Microsoft

AZURE

Azure Automation

Automatización de la operación de servicios, máquinas virtuales y recursos de Azure.

TEMAS PRINCIPALES

- RUNBOOKS
- ACTUALIZACIONES
- INVENTARIO
- CONFIGURACIÓN

Microsoft Azure

Documentación técnica · Diseño original de portada

© Microsoft Corporation.

DSC de PowerShell

INICIO RÁPIDO

Creación de una configuración de DSC a partir de la directiva de grupo

TUTORIAL

Configurar servidores y administrar el desfase con State Configuration

Trabajar con entornos híbridos

INFORMACIÓN GENERAL

Introducción a Hybrid Runbook Worker

IMPLEMENTAR

Desplegar Windows o Linux Worker basado en extensiones

Implementación de Hybrid Runbook Worker de Windows

Implementación de Hybrid Runbook Worker de Linux

Inicio de runbooks en Hybrid Runbook Worker

Automatización entre servicios

TUTORIAL

Envío de un correo electrónico desde un Runbook mediante SendGrid

COMENZAR

Inicio de un runbook desde alertas de Azure

IMPLEMENTAR

Implementación de una plantilla de Administrador de Recursos en un cuaderno de ejecución de PowerShell de Automation

Arquitectura

ARQUITECTURA

Recomendaciones para implementar Automatización

Solución de problemas de Azure Automation

GUÍA PASO A PASO

Solución de problemas de runbooks

Solución de problemas de Hybrid Runbook Worker basado en extensiones

Solución de problemas de identidad administrada

Datos que se van a recopilar al abrir un caso para Azure Automation

¿Qué es Azure Automation?

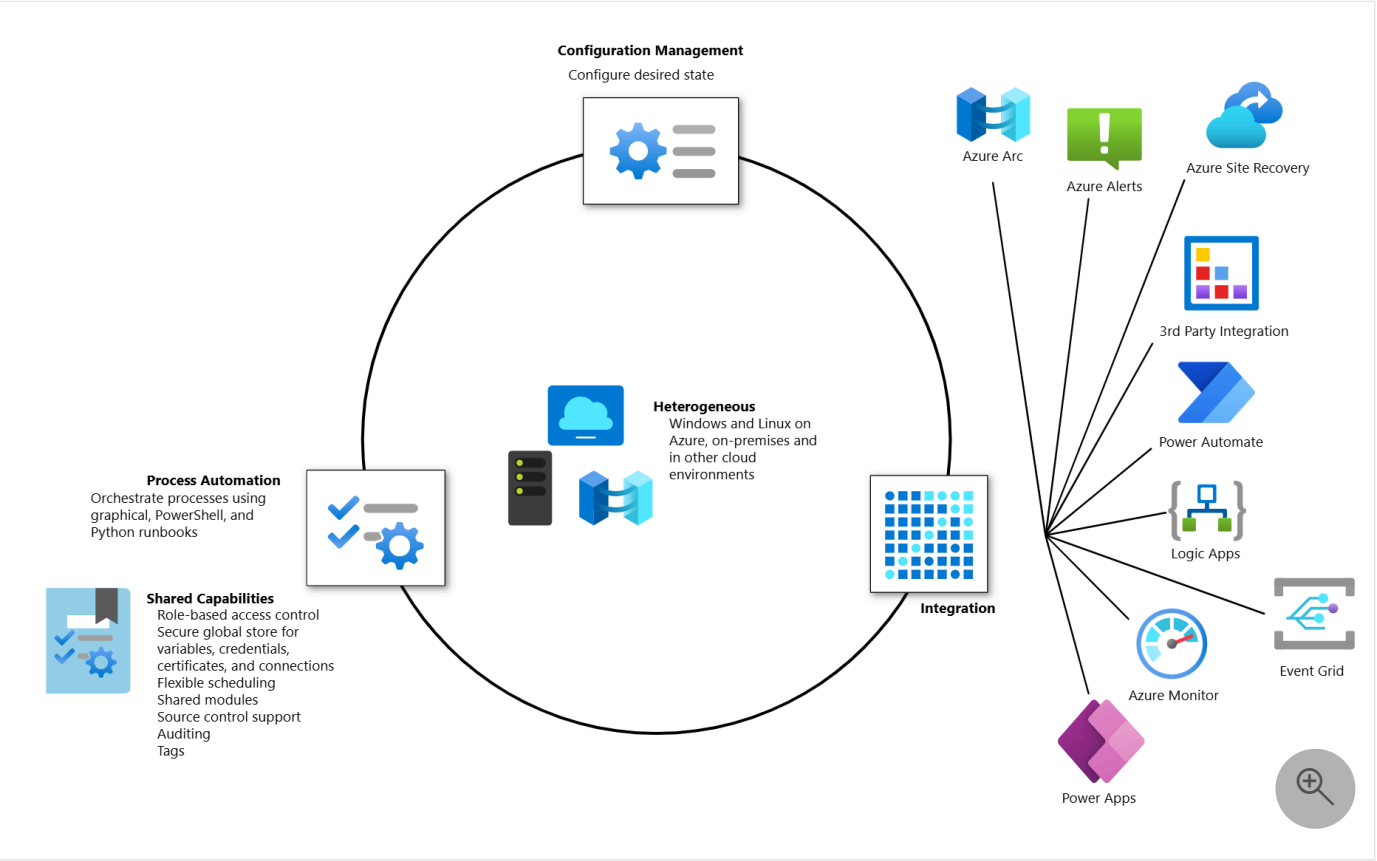
Azure Automation es un servicio de automatización basado en la nube que te ayuda a automatizar procesos y soporta una gestión consistente tanto en entornos Azure como no Azure. Incluye capacidades como automatización de procesos, gestión de configuración y capacidades compartidas para entornos heterogéneos.

Al utilizar Azure Automation, puedes reducir el esfuerzo manual, mejorar la eficiencia operativa y garantizar la ejecución coherente de las tareas en todo tu entorno. Puedes aplicar automatización en tres áreas clave de las operaciones en la nube:

- Implementación y administración: proporcione una infraestructura repetible y coherente en forma de código.
- Respuesta: cree una automatización basada en eventos para diagnosticar y resolver problemas.
- Organización: organice e integre la automatización con otros servicios y productos de Azure o de terceros.

Cómo se integra Azure Automation con otros servicios de Azure

Varios servicios de Azure se integran con Azure Automation para soportar el despliegue y la gestión, la respuesta a eventos y la orquestación de flujos de trabajo. Cada servicio proporciona un conjunto de capacidades y sirve como plataforma programable para construir soluciones en la nube. Por ejemplo, Azure Bicep y Resource Manager proporcionan un lenguaje para desarrollar plantillas de implementación repetibles y coherentes dirigidas a recursos de Azure. Azure Automation ejecuta estas plantillas para desplegar un recurso de Azure y luego realiza tareas de configuración posteriores al despliegue.



Dependiendo de tus necesidades, Azure Automation puede integrarse con uno o más de los siguientes servicios de Azure:

- Los servidores habilitados para Azure Arc permiten la incorporación simplificada de máquinas híbridas a Change Tracking e Inventario mediante AMA y el rol Hybrid Runbook Worker.
- Los grupos de acciones de Alertas de Azure pueden iniciar un runbook de Automation cuando se genera una alerta.
- Azure Monitor recopila métricas y datos de registro de tu cuenta de Automation para análisis posteriores y actúa sobre la telemetría.
- Azure Policy incluye definiciones de iniciativa para ayudar a establecer y mantener el cumplimiento con diferentes estándares de seguridad en relación con su cuenta de Automation.
- Azure Site Recovery puede usar runbooks de Azure Automation para automatizar planes de recuperación.

Estos servicios de Azure pueden funcionar con recursos de runbook y trabajo de Automation mediante un webhook HTTP o un método de API:

- Azure Logic Apps
- azure Power Apps
- Azure Event Grid

- Azure Power Automate

> **Nota:**
>
> Este servicio admite **Azure Lighthouse**, que permite a los proveedores de servicios iniciar sesión en su propio inquilino para administrar las suscripciones y los grupos de recursos que los clientes hayan delegado.

Automation proporciona control total durante la implementación, las operaciones y la retirada de las cargas de trabajo y los recursos empresariales.

Azure Automation ofrece las siguientes capacidades principales:

Automatización de procesos

La automatización de procesos en Azure Automation te ayuda a automatizar tareas de gestión repetitivas, que consumen mucho tiempo y son propensas a errores. Este servicio le ayuda a centrarse en el trabajo que agrega valor empresarial. Al reducir errores y aumentar la eficiencia, también ayuda a disminuir los costes operativos. Para saber más sobre el entorno operativo de automatización de procesos, consulta Runbook execution in Azure Automation.

La automatización de procesos permite la integración de los servicios de Azure y otros sistemas de terceros necesarios para desplegar, configurar y gestionar tus procesos de extremo a extremo. Esta capacidad te permite crear runbooks gráficos, de PowerShell y de Python. Para ejecutar runbooks directamente en máquinas Windows o Linux o sobre recursos en entornos locales u otros entornos cloud para gestionar esos recursos locales, puedes desplegar un Hybrid Runbook Worker.

Webhooks te permiten atender solicitudes y garantizar la entrega y operaciones continuas activando automatizaciones desde Azure Logic Apps, Azure Functions, productos o servicios ITSM, DevOps y sistemas de monitorización.

Administración de configuración

La gestión de configuración en Azure Automation ayuda a garantizar que los sistemas se mantengan consistentes y conformes mediante la aplicación y mantenimiento automático de las configuraciones deseadas. Azure Automation soporta la gestión de configuración a través de la capacidad State Configuration.

Configuración de estado de Azure Automation

Configuración de estado de Azure Automation es una característica en la nube para la configuración de estado deseado (DSC) de PowerShell que proporciona servicios para entornos empresariales. Utilizando esta función, puedes gestionar tus recursos DSC en Azure Automation y aplicar configuraciones a máquinas virtuales o físicas desde un servidor DSC pull en la nube de Azure.

Funcionalidades compartidas

Azure Automation ofrece varias capacidades compartidas, incluyendo recursos compartidos, control de acceso basado en roles (Azure RBAC), programación flexible, integración de control de versiones, auditoría y etiquetado.

Recursos compartidos

Azure Automation incluye un conjunto de recursos compartidos que te ayudan a automatizar y configurar tus entornos a gran escala.

- **Programas** : Activar operaciones de automatización en tiempos predefinidos.
- **Módulos** : administre Azure y otros sistemas. Importe módulos a la cuenta de Automation para Microsoft, recursos de DSC y cmdlets definidos personalizados, de la comunidad o de terceros.
- **Galerie de módulos** - Soporta integración nativa con la Galería de PowerShell. Puedes ver los runbooks e importarlos a la cuenta de Automation. También te ayuda a empezar rápidamente a integrar y crear tus procesos desde PowerShell gallery y Microsoft Script Center.
- **Paquetes de Python 2 y 3** - Compatibilidad con runbooks de Python 2 y 3 para su cuenta de Automation.
- **Credenciales** : almacene de forma segura la información confidencial que los runbooks y las configuraciones pueden usar en tiempo de ejecución.
- **Conexiones** - Almacena pares de nombre y valor de información común para las conexiones con sistemas. El autor del módulo define las conexiones en runbooks y configuraciones para usarlas en tiempo de ejecución.
- **Certificados** - Definir información para usar en la autenticación y seguridad de los recursos desplegados cuando se accede mediante runbooks o configuraciones DSC en tiempo de ejecución.

- **Variables**: almacenan contenido que puede usarse en runbooks y configuraciones. Puedes cambiar los valores de las variables sin modificar ninguno de los libros de ejecuciones ni configuraciones que los referencian.

Control de acceso basado en rol

Azure Automation soporta el control de acceso basado en roles de Azure para regular el acceso a la cuenta de Automation y sus recursos. Para más información sobre la configuración de Azure RBAC en su cuenta de Automation, runbooks y trabajos, consulte Control de acceso basado en roles en Azure Automation.

Programación flexible

Azure Automation ofrece una programación flexible para automatizar los runbooks en momentos concretos y con intervalos periódicos, como cada hora, diariamente, semanalmente o mensualmente. Puedes configurar horarios por días específicos de la semana, días del mes o patrones de fechas personalizados. Un solo runbook puede vincularse a varias programaciones, y viceversa. Se encarga automáticamente de los ajustes del horario de verano, soporta la configuración de zonas horarias para operaciones globales y permite la creación y gestión de horarios a través del portal de Azure o PowerShell. Para saber más, consulta Gestionar horarios en Azure Automation.

Integración de control de código fuente

Azure Automation soporta integración con control de versiones. Esta característica promueve la configuración como código donde los runbooks o las configuraciones pueden insertarse en un sistema de control de código fuente.

Auditing

Azure Automation soporta auditorías mediante registros de auditoría que capturan operaciones de creación, actualización y eliminación entre cuentas de Automation, libros de operaciones y otros activos. Puedes enviar estos registros a un espacio de trabajo de Azure Monitor para monitorización de seguridad, revisiones de cumplimiento e investigación de cambios operativos. Para obtener más información, consulta Reenviar los datos de trabajos de Azure Automation a los logs de Azure Monitor y la retención de datos en Azure Automation.

Etiquetado

Azure Automation soporta etiquetas de Azure Resource Manager para categorizar y organizar recursos de Automation. Al crear una cuenta de Automation desde el portal, puedes especificar etiquetas de Resource Manager para ayudar a organizar tus recursos de Azure. Para saber más, consulta Recursos de etiquetas, grupos de recursos y suscripciones para la organización lógica.

Automatización híbrida

Automatiza cargas de trabajo en entornos Azure y locales, incluyendo servicios como SQL Server, Active Directory y SharePoint Server mediante Hybrid Runbook Worker.

Compatibilidad heterogénea (Windows y Linux)

La automatización está diseñada para funcionar en servidores físicos y máquinas virtuales de Windows y Linux en entornos no Azure, locales y otros proveedores de nube. Proporciona un enfoque coherente para automatizar y configurar las cargas de trabajo desplegadas y los sistemas operativos que las ejecutan. La característica Hybrid Runbook Worker le permite ejecutar runbooks directamente en el servidor físico o la máquina virtual que no sean de Azure y que hospedan el rol, así como en recursos del entorno para administrar esos recursos locales.

A través de servidores habilitados para Arc, proporciona una experiencia consistente de despliegue y gestión para tus equipos que no son Azure. Además, posibilita la integración con el servicio Automation mediante el marco de extensión de máquina virtual, para implementar el rol de Hybrid Runbook Worker y simplificar la incorporación en Seguimiento de cambios e inventario con el agente de Azure Monitor.

Escenarios frecuentes

Azure Automation permite la administración de la infraestructura y las aplicaciones a lo largo de su ciclo de vida. Entre los escenarios habituales se incluyen los siguientes:

- **Programar tareas** - Detener máquinas virtuales o servicios por la noche e iniciarlos durante el día, así como programar flujos de trabajo de mantenimiento periódicos semanales o mensuales.
- **Compilar y desplegar recursos** - Desplegar recursos en un entorno híbrido usando runbooks y plantillas de Azure Resource Manager. Integra en herramientas de desarrollo como Jenkins y Azure DevOps.
- **Mantenimiento periódico** - Ejecutar tareas que deben realizarse en intervalos definidos, como purgar datos antiguos o obsoletos, o reindexar una base de datos SQL.

- **Responder a alertas** - Orquestar una respuesta cuando se generen alertas basadas en costes, en sistemas, en servicios y en utilización de recursos.
- **Gestión del ciclo de vida de recursos de Azure** - Gestionar recursos IaaS y PaaS a lo largo de su ciclo de vida, incluyendo:
  - Aprovisionamiento y desaprovisionamiento de recursos.
  - Agregue las etiquetas, los bloqueos, los NSG y las UDR correctos según las reglas de negocio.
  - Creación, eliminación y actualización de grupos de recursos.
  - Inicio de grupo de contenedores.
  - Registro del registro DNS.
  - Cifrar máquinas virtuales.
  - Configuración de disco (instantánea de disco, eliminación de instantáneas antiguas).
  - Administración de suscripciones.
  - Inicie y detenga recursos para ahorrar costes.
- **Monitorización e integración** - Integrar con sistemas externos de primera mano (a través de Azure Monitor) o de terceros, incluyendo:
  - Asegurarse de que las operaciones de creación y eliminación de recursos se capturen en SQL.
  - Envío de datos de uso de recursos a la API web.
  - Envía los datos de monitorización a ServiceNow, Event Hubs, New Relic, y así sucesivamente.
  - Recopile y almacene información sobre los recursos de Azure.
  - Realizar comprobaciones de supervisión de SQL y elaboración de informes.
  - Comprobación de disponibilidad del sitio web.
- **Escenarios de automatización de desarrollo/pruebas** : parar, arrancar y escalar recursos.
- **Automatización relacionada con la gobernanza** - Aplicar o actualizar automáticamente etiquetas y bloqueos.
- **Azure Site Recovery:** organice scripts previos o posteriores definidos en un flujo de trabajo de recuperación ante desastres de Site Recovery.
- **Azure Virtual Desktop** - Orquestar escalabilidad de VMs o iniciar/detener VMs según su utilización.
- **Configurar máquinas virtuales**: evalúe y configure máquinas Windows y Linux con configuraciones para la infraestructura y la aplicación.
- **Recuperar el inventario** - Obtenga un inventario completo de los recursos implementados para la selección de objetivos, la generación de informes y el cumplimiento.
- **Encontrar cambios** - Identificar y aislar cambios en las máquinas que pueden causar una mala configuración y mejorar el cumplimiento operativo. Corríjalos o escálelos a sistemas

de administración.

Precios de Azure Automation

La automatización de procesos incluye trabajos de runbook y monitores. La facturación de los trabajos se basa en el número de minutos de tiempo de ejecución del trabajo usados en el mes, mientras que para los observadores se basa en el número de horas gastadas en un mes. Se incurre en cargos por la automatización de procesos cada vez que se ejecuta un trabajo o monitor. Creas cuentas de automatización con un SKU básico, donde los primeros 500 minutos de ejecución de trabajo son gratuitos por suscripción. Solo pagas por minutos u horas que superen las 500 unidades gratuitas incluidas.

Para saber más sobre los precios de Azure Automation, visita la página de precios.

Pasos siguientes

Creación de una cuenta de Automation

recomendaciones bien diseñadas para implementar la automatización

---

Last updated on 23/07/2026

Elija los servicios de Automation en Azure

En este artículo se explican varios servicios de Automation que se ofrecen en el entorno de Azure. Estos servicios pueden automatizar procesos empresariales y operativos y resolver problemas de integración entre varios servicios, sistemas y procesos. Los servicios de automatización pueden definir entradas, acción, actividad a realizar, condiciones, manejo de errores, y generación de salidas. Con estos servicios, puede ejecutar varias actividades según una programación o realizar una ejecución manual basada en la demanda. Cada servicio tiene sus ventajas únicas y su público objetivo. Con estos servicios, puede desplazar el esfuerzo de la realización de tareas operativas de forma manual a la creación de automatización para esas tareas. Con esto se logra lo siguiente:

- Reducción del tiempo para realizar una acción
- Reducción del riesgo en la realización de la acción
- Mayor capacidad humana para fomentar la innovación
- Normalización de operaciones

Categorías en operaciones de automatización

La automatización se requiere en tres categorías amplias de operaciones:

- **Implementación y administración de recursos**: cree y configure mediante programación y usando automatización o infraestructura como herramientas de código para ofrecer una implementación y administración repetibles y coherentes de los recursos en la nube. Por ejemplo, con una plantilla de Azure Resource Manager o un script de automatización se puede implementar un grupo de seguridad de red de Azure y crear las reglas de este.

- **Respuesta a eventos externos**: en función de un evento externo crítico, como responder a los cambios de la base de datos, actuar según las entradas dadas a una página web, entre otros, puede diagnosticar y resolver problemas.

- **Orquestación compleja**: mediante la integración con productos propios o de terceros, puede definir flujos de trabajo de automatización de un extremo a otro.

Servicios de Azure para Automatización

Hay varios servicios de Azure que pueden cumplir los requisitos anteriores. Cada servicio tiene sus ventajas y limitaciones; los clientes pueden usar varios servicios para satisfacer sus

requisitos de automatización.

Implementación y administración de recursos

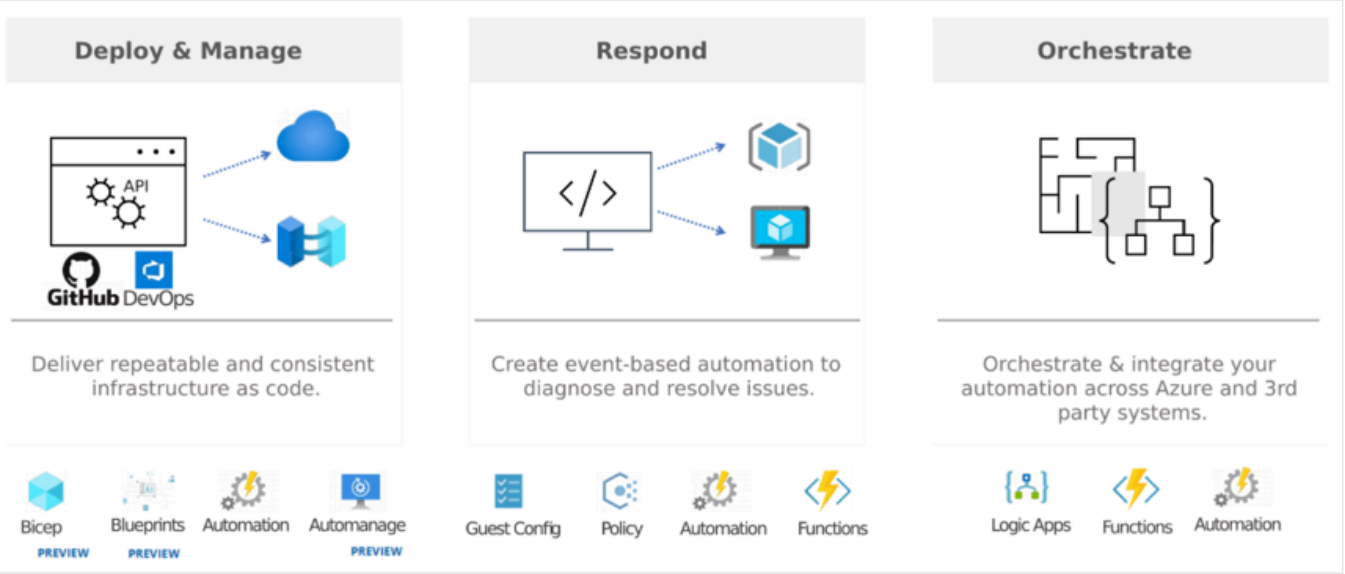
- Plantillas de Azure Resource Manager (ARM) con Bicep
- Azure Blueprints
- Azure Automation
- Azure Automanage (para la configuración y administración de la máquina)

Respuesta a eventos externos

- Azure Functions
- Azure Automation
- Configuración de invitado de Azure Policy (para realizar una acción cuando haya un cambio en el estado de cumplimiento del recurso)

Orquestación e integración complejas con productos de primeras o terceras partes

- Azure Logic Apps
- Azure Functions o Azure Automation (la aplicación lógica de Azure tiene más de 400 conectores a otros servicios, incluidos Azure Automation y Azure Functions, que podrían usarse para satisfacer escenarios de automatización complejos)



Implementación y administración de servicios de Automation

Plantilla de Azure Resource Manager (ARM)

Azure Resource Manager proporcionan un lenguaje para desarrollar plantillas de implementación repetibles y coherentes dirigidas a recursos de Azure. La plantilla es un archivo de notación de objetos JavaScript (JSON) que contiene la infraestructura y la configuración del proyecto. Usa sintaxis declarativa, lo que permite establecer lo que pretende implementar sin tener que escribir la secuencia de comandos de programación para crearla. En la plantilla se especifican los recursos que se van a implementar y las propiedades de esos recursos. Más información.

Bicep

Hemos presentado un nuevo lenguaje denominado Bícep que ofrece las mismas capacidades que las plantillas de ARM pero con una sintaxis más fácil de usar. Cada archivo de Bicep se convierte automáticamente en una plantilla de ARM durante la implementación. Si está pensando en usar infraestructura como opciones de código, le recomendamos Bicep. Para obtener más información, consulte ¿Qué es Bicep?

En la tabla siguiente se describen los escenarios y los usuarios de la plantilla de ARM y Bicep:

Expandir tabla

Escenarios	Usuarios
Cree, administre y actualice recursos de infraestructura, como máquinas virtuales, redes, cuentas de almacenamiento y contenedores, entre otros. Implemente aplicaciones, agregue etiquetas, asigne directivas y controle el acceso basado en roles, todo de forma declarativa como código e integrado con sus herramientas de CI\CD. Administre varios entornos, como producción, no producción y recuperación ante desastres. Implemente los recursos de forma coherente y confiable a escala.	Desarrolladores de aplicaciones, administradores de infraestructura, ingenieros de DevOps que usan Azure por primera vez o que usan Azure como su nube principal. Ingeniero de TI\Arquitecto en la nube responsable de la implementación de la infraestructura en la nube.

Azure Blueprints (versión preliminar)

Nota:

El 11 de julio de 2026, Azure Blueprints (versión preliminar) quedará en desuso. Más información

Azure Blueprints (versión preliminar) define un conjunto repetible de recursos de Azure que implementa y cumple los estándares, patrones y requisitos de la organización. Blueprints son una manera declarativa de orquestar la implementación de varias plantillas de recursos y otros artefactos, como asignaciones de roles, asignaciones de Policy, plantillas de ARM y grupos de recursos. Más información.

Expandir tabla

Escenarios	Usuarios
Cree, administre y actualice los recursos de infraestructura para asegurarse de que la infraestructura implementada respeta los estándares de cumplimiento de la organización. Audite y realice un seguimiento de las implementaciones de Azure.	Auditores y grupos de tecnologías de la información centrales, responsables de garantizar que la infraestructura de Azure implementada respeta los estándares de cumplimiento de la organización.

Azure Automation

Azure Automation orquesta procesos repetitivos mediante runbooks gráficos, de PowerShell y Python en el entorno híbrido o en la nube. Proporciona recursos compartidos persistentes que incluyen variables, conexiones y objetos que permiten la orquestación de trabajos complejos. Más información.

Hay más de 3000 módulos en la Galería de PowerShell y la comunidad de PowerShell sigue creciendo. Los módulos de Azure Automation basados en PowerShell pueden trabajar con varias aplicaciones y proveedores, tanto propios como de terceros. A medida que más proveedores de aplicaciones publiquen módulos de PowerShell para tareas de integración, extensibilidad y automatización, podrá usar un script de PowerShell existente tal cual para ejecutarlo como un runbook de PowerShell en una cuenta de Automation sin aplicar ningún cambio.

Expandir tabla

Escenarios	Usuarios
Permite que se escriba un runbook de Automation PowerShell que implemente un recurso de Azure	Administradores de TI, administradores del sistema, administradores de operaciones de TI

Escenarios	Usuarios
mediante una plantilla de Azure Resource Manager. Programe tareas, por ejemplo, detener máquinas virtuales o servicios de desarrollo/pruebas por la noche y activarlas durante el día. Respuesta a alertas como alertas del sistema, alertas de servicio, alertas de CPU/memoria elevadas, creación de vales de ServiceNow, entre otros. Automatización híbrida donde se puede gestionar la automatización de servidores locales, como SQL Server y Active Directory, entre otros. La administración y la gobernanza del ciclo de vida de los recursos de Azure incluyen el aprovisionamiento de recursos, el desaprovisionamiento y la adición de etiquetas, bloqueos y grupos de seguridad de red correctos, entre otros.	que son expertos en el uso de scripts basados en PowerShell o Python. Los administradores de infraestructura administran la infraestructura local mediante scripts o la ejecución de trabajos de larga duración, como operaciones de fin de mes en servidores que se ejecutan en el entorno local.

Azure Automation basada en la administración de invitados

Administración de configuración: recopila el inventario y realiza un seguimiento de los cambios en el entorno. Más información. Puede configurar el estado deseado de las máquinas para detectar y corregir el desfase de configuración. Más información.

Administración de actualizaciones: evalúe el cumplimiento de los servidores y programe la instalación de actualizaciones en las máquinas. Más información.

Expandir tabla

Escenarios	Usuarios
Detecte y alerte sobre los cambios en software, servicios, archivos y registros en sus máquinas, manteniéndose atento a todo lo instalado en sus servidores. Evalúe e instale actualizaciones en los servidores mediante Azure Update Management. Configure el estado deseado de los servidores y asegúrese de que se mantienen compatibles.	TI central\Administradores de infraestructura\Auditores que buscan requisitos normativos a escala y garantizan que el estado final de los servidores sea el deseado, con revisiones y auditados.

Azure Automanage (versión preliminar)

Reemplaza las tareas operativas diarias repetitivas por un modelo de administración de solo excepciones, donde un estado correcto y estable de la máquina virtual es igual a la administración de manos libres. Más información.

Compatibilidad con Linux y Windows

- Le permite incorporar máquinas virtuales de forma inteligente a servicios de Azure de procedimientos recomendados
- Le permite configurar automáticamente cada servicio según los procedimientos recomendados de Azure
- Admite la personalización de los servicios de procedimientos recomendados a través de la plantilla de procedimientos recomendados de máquina virtual para la carga de trabajo de desarrollo/pruebas y producción
- Le permite supervisar los desfases y corregirlos cuando se detecten
- Proporciona una experiencia sencilla (apuntar, elegir, configurar y olvidar)

Expandir tabla

Escenarios	Usuarios
Configura automáticamente el sistema operativo invitado según la configuración de línea de base de Microsoft. Detecta automáticamente los desfases y los corrige en todo el ciclo de vida de una máquina virtual. Tiene como objetivo una administración manos libres de las máquinas.	Los administradores de TI, administradores de infraestructura y administradores de operaciones de TI son responsables de administrar la carga de trabajo del servidor, las tareas de administración diarias, como la creación de copias de seguridad, la recuperación ante desastres, las actualizaciones de seguridad y la respuesta a las amenazas de seguridad en Azure y en el entorno local. Desarrolladores que no desean administrar servidores o dedican tiempo tareas menos prioritarias.

Respuesta a eventos en el flujo de trabajo de Automation

Configuración de invitados basada en Azure Policy

La Configuración de invitado basada en Azure Policy es la siguiente iteración de la Configuración de estado de Azure Automation. Más información.

Puede comprobar lo que está instalado en:

- La siguiente iteración de Azure Automation State Configuration.
- Para aplicaciones maliciosas conocidas, certificados de protocolo, privilegios de administrador y estado de los agentes.
- Para el contenido escrito por el cliente.

Expandir tabla

Escenarios	Usuarios
Obtenga datos de cumplimiento que pueden incluir: Configuración del sistema operativo: archivos, registro y servicios, Configuración o presencia de la aplicación, Comprobación de la configuración del entorno. Audite o implemente la configuración en todas las máquinas (Establecer) del ámbito de forma reactiva en las máquinas existentes o de forma proactiva en las máquinas nuevas a medida que se implementa. Responda a eventos de directiva para proporcionar corrección a petición o corrección continua.	TI central, los administradores de infraestructura y los auditores (custodios de la nube) están trabajando para cumplir los requisitos normativos a escala y garantizar que el estado final de los servidores sea el deseado. Los equipos de la aplicación validan el cumplimiento antes de publicar el cambio.

Azure Automation: automatización de procesos

Orquestar procesos repetitivos utilizando runbooks gráficos, de PowerShell y de Python en la nube o en un entorno híbrido. Más información.

- Proporciona recursos compartidos persistentes que incluyen variables, conexiones y objetos que permiten la orquestación de trabajos complejos
- Le permite invocar un runbook en función de una alerta de Azure Monitor o a través de un webhook

Expandir tabla

Escenarios	Usuarios
Respuesta a alertas del sistema, alertas de servicio o alertas de CPU/memoria elevadas de herramientas de supervisión propias o de terceros, como Splunk o ServiceNow, creación de alertas base de vales de ServiceNow, etc. Escenarios de automatización híbrida en los que puede administrar la automatización en servidores locales, como SQL Server y Active Directory, entre otros, en función de un evento externo. Administración y gobernanza del ciclo de vida de los recursos de Azure, que incluye el aprovisionamiento de recursos, el desaprovisionamiento y la adición de etiquetas, bloqueos y NSGs correctos, etcétera, en función de las alertas de Azure Monitor.	Administradores de TI, administradores del sistema, administradores de operaciones de TI que son expertos en el uso de scripts basados en PowerShell o Python.

funciones de Azure

Proporciona una plataforma de proceso controlada por eventos sin servidor para la automatización que permite escribir código para reaccionar a eventos críticos de diversos orígenes, servicios de terceros y sistemas locales. Por ejemplo, un desencadenador HTTP sin preocuparse por la plataforma subyacente. Más información.

- Puede usar una variedad de lenguajes para escribir funciones en un lenguaje de su elección, como C#, Java, JavaScript, PowerShell o Python, y centrarse en fragmentos de código específicos. El tiempo de ejecución de Functions es de código abierto.
- Puede elegir el plan de hospedaje según los requisitos de escalado de la aplicación de funciones, la funcionalidad y los recursos necesarios.
- Puede orquestar flujos de trabajo complejos a través de Durable Functions.
- Es recomendable que evite funciones grandes y de larga duración que puedan provocar problemas inesperados de tiempo de espera. Más información.
- Al escribir scripts de PowerShell en aplicaciones de funciones, debe ajustar los scripts para definir cómo se comporta la función; por ejemplo, cómo se desencadena y sus parámetros de entrada y salida. Más información.

Expandir tabla

Escenarios	Usuarios
Responder a eventos en recursos: como agregar etiquetas al centro de costos base del grupo de	Los desarrolladores de aplicaciones que tienen conocimientos de lenguajes de codificación

Escenarios	Usuarios
recursos, cuando se elimina la máquina virtual, etc. Establezca tareas programadas, como establecer un patrón para detener e iniciar una máquina virtual en un momento específico, leer el contenido de Blob Storage a intervalos regulares, etc. Gestionar alertas de Azure para notificar al equipo cuando la actividad de la CPU se dispara al 90 %. Orquestación con sistemas externos como Microsoft 365. Responder a cambios en una base de datos.	como C#, F#, PHP, Java, JavaScript, PowerShell o Python. Arquitectos en la nube que crean aplicaciones sin servidor en las que Azure Functions podría formar parte de un flujo de trabajo de aplicaciones más grande.

Orquestación de trabajos complejos en Azure Automation

Azure Logic Apps

Logic Apps es una plataforma para crear y ejecutar flujos de trabajo de orquestación compleja que integren sus aplicaciones, datos, servicios y sistemas. Más información.

- Permite crear integraciones inteligentes entre aplicaciones, servicios y sistemas de primeras y terceras partes que se ejecutan en entornos locales, híbridos y nativos en la nube.
- Permite usar conectores administrados de entre más de 450 ecosistemas de conectores de Azure (y creciendo) para usarlos en los flujos de trabajo.
- Proporciona soporte de primera clase para escenarios B2B y de integración empresarial.
- Permite flexibilidad para crear y editar flujos de trabajo visualmente: enfoque de poco código\sin código
- Solo se ejecuta en la nube.
- Proporciona una gran colección de desencadenadores y acciones listos para su uso.

Expandir tabla

Escenarios	Usuarios
Programación y envío de notificaciones por correo electrónico mediante Office 365 cuando se produce un evento específico. Por ejemplo, se carga un archivo nuevo. Procesamiento y enrutamiento de pedidos entre sistemas locales y servicios en la nube. Traslado de archivos cargados de un servidor SFTP o FTP a Azure Storage. Supervisión de tuits, análisis de opiniones y creación de alertas o tareas para los elementos que se deben revisar.	Los integradores y desarrolladores de Pro, profesionales de TI que desean usar una opción de poco código o sin código para escenarios de integración avanzada en sistemas o API externos.

Azure Automation: automatización de procesos

Orquestar procesos repetitivos utilizando runbooks gráficos, de PowerShell y de Python en la nube o en un entorno híbrido. Proporciona recursos compartidos persistentes que incluyen variables, conexiones y objetos que permiten la orquestación de trabajos complejos. Más información.

Expandir tabla

Escenarios	Usuarios
Administración y gobernanza del ciclo de vida de los recursos de Azure, que incluye el aprovisionamiento y desaprovisionamiento de recursos, así como la adición de las etiquetas, bloqueos y grupos de seguridad de red (NSG) correctos, entre otros, mediante runbooks que se desencadenan a partir de alertas de ITSM. Use Hybrid Worker como puente entre la nube y el entorno local, lo que permite la administración local de recursos o usuarios. Ejecute flujos de trabajo complejos de recuperación ante desastres a través de runbooks de Automation. Ejecute runbooks de automatización como parte del flujo de trabajo de Logic Apps mediante Azure Automation Connector.	Administradores de TI, administradores del sistema, administradores de operaciones de TI que son expertos en el uso de scripts basados en PowerShell o Python. Los administradores de infraestructura administran la infraestructura local mediante scripts o la ejecución de trabajos de larga duración, como operaciones de fin de mes en servidores que se ejecutan en el entorno local.

funciones de Azure

Proporciona una plataforma de proceso controlada por eventos sin servidor para la automatización que permite escribir código para reaccionar a eventos críticos de diversos orígenes, servicios de terceros y sistemas locales. Por ejemplo, un desencadenador HTTP sin preocuparse por la plataforma subyacente. Más información.

- Puede usar una variedad de lenguajes para escribir funciones en un lenguaje de su elección, como C#, Java, JavaScript, PowerShell o Python, y centrarse en fragmentos de código específicos. El tiempo de ejecución de Functions es de código abierto.
- Puede elegir el plan de hospedaje según los requisitos de escalado de la aplicación de funciones, la funcionalidad y los recursos necesarios.
- Puede orquestar flujos de trabajo complejos a través de Durable Functions.
- Es recomendable que evite funciones grandes y de larga duración que puedan provocar problemas inesperados de tiempo de espera. Más información.
- Al escribir scripts de PowerShell en aplicaciones de funciones, debe ajustar los scripts para definir cómo se comporta la función; por ejemplo, cómo se desencadena y sus parámetros de entrada y salida. Más información.

Expandir tabla

Escenarios	Usuarios
Responder a eventos en recursos: como agregar etiquetas al centro de costos base del grupo de recursos, cuando se elimina la máquina virtual, etc. Establezca tareas programadas, como establecer un patrón para detener e iniciar una máquina virtual en un momento específico, leer el contenido de Blob Storage a intervalos regulares, etc. Procesar alertas de Azure en las que puede enviar el evento del equipo cuando la actividad de CPU aumente al 90 %. Orquestación con sistemas externos como Microsoft 365. Ejecuta Azure Function como parte del flujo de trabajo de Logic Apps a través de Azure Function Connector.	Desarrolladores de aplicaciones que tienen conocimientos de lenguajes de codificación como C#, F#, PHP, Java, JavaScript, PowerShell o Python. Arquitectos en la nube que crean aplicaciones sin servidor en las que una o varias instancias de Azure Functions podrían formar parte de un flujo de trabajo de aplicaciones más grande.

Pasos siguientes

Para obtener información sobre cómo ejecutar de forma segura los trabajos de automatización, consulte Procedimientos recomendados sobre seguridad en Azure Automation.

Last updated on 07/04/2026

Preguntas más frecuentes sobre Azure Automation

Artículo • 20/08/2024

En este artículo de preguntas más frecuentes de Microsoft, se presenta una lista con las preguntas frecuentes sobre Azure Automation. Si tiene alguna otra pregunta sobre sus funcionalidades, vaya al foro de discusión y publíquela. Si una pregunta es frecuente, se agrega a este artículo para que se encuentre de forma rápida y sencilla.

¿Por qué no puedo crear un trabajo de automatización en la región Oeste de Europa?

Es posible que experimente un retraso o un error en la creación de trabajos debido a problemas de escalabilidad en la región Oeste de Europa. Para obtener más información, vea Creación de un trabajo de Automation en la región Oeste de Europa.

¿Update Management puede evitar actualizaciones inesperadas de nivel de sistema operativo?

Sí. Para obtener más información, consulte Exclusión de actualizaciones.

¿Por qué no se aplican actualizaciones críticas o de seguridad a una máquina Linux con Update Management?

Al implementar actualizaciones en una máquina Linux, puede seleccionar clasificaciones de actualizaciones. Esta opción filtra las actualizaciones que cumplen los criterios especificados. Para más información, consulte Clasificaciones de actualizaciones de Linux.

¿Update Management puede implementar actualizaciones en los inquilinos de Azure?

Sí. Para obtener instrucciones, consulte Implementación de actualizaciones en los inquilinos de Azure.

¿Qué versión de Python 3 se admite en Azure Automation?

En el caso de los trabajos en la nube, se admite Python 3.8. Los scripts y paquetes de cualquier versión 3.x podrían funcionar si el código es compatible con distintas versiones.

En el caso de los trabajos híbridos de Hybrid Runbook Worker para Windows, puede optar por instalar cualquier versión 3.x que desee usar. En el caso de los trabajos híbridos de Hybrid Runbook Worker para Linux, dependemos de la versión de Python 3 instalada en el equipo para ejecutar DSC OMSConfig y Hybrid Worker para Linux. Se recomienda instalar la versión 3.6; sin embargo, las distintas versiones también deben funcionar si no hay cambios importantes en los contratos o las firmas de método entre las versiones de Python 3.

¿Pueden los runbooks de Python 2 y Python 3 ejecutarse en la misma cuenta de Automation?

Sí, no hay ninguna limitación para usar runbooks de Python 2 y Python 3 en la misma cuenta de Automation.

¿Cuál es el plan para migrar los paquetes y runbooks existentes de Python 2 a Python 3?

Azure Automation no pretende migrar los runbooks y paquetes de Python 2 a Python 3. Tendrá que realizar esta migración usted mismo. Los runbooks y paquetes nuevos y existentes de Python 2 seguirán funcionando.

¿Cuáles son los paquetes admitidos de manera predeterminada en el entorno de Python 3?

El paquete 4.0.0 de Azure. Para más información, consulte Administración de paquetes de Python 3.

¿Qué ocurre si ejecuto un runbook de Python 3 que haga referencia a un paquete de Python 2 o viceversa?

Python 2 y Python 3 tienen entornos de ejecución diferentes. Mientras se ejecuta un runbook de Python 2, solo se pueden importar paquetes de Python 2 y similares para Python 3.

¿Cómo se diferencian los paquetes y runbooks de Python 2 y Python 3?

Python 3 es una nueva definición de runbook, que distingue entre los runbooks de Python 2 y Python 3. Del mismo modo, se introduce otro tipo de paquete para los paquetes de Python 3.

¿Cómo sabe Hybrid Runbook Worker la versión de Python que se debe ejecutar cuando se instalan Python 2 y Python 3?

En el caso de Runbook Worker de Windows, al ejecutar un runbook de Python 2, este busca primero la variable de entorno `PYTHON_2_PATH` y valida si apunta a un archivo ejecutable válido. Por ejemplo, si la carpeta de instalación es `C:\Python2`, comprobaría si `C:\Python2\python.exe` es una ruta de acceso válida. Si no se encuentra, busca la variable de entorno `PATH` para realizar una comprobación similar.

Para Python 3, primero busca la variable env `PYTHON_3_PATH` y, a continuación, recurre a la variable de entorno `PATH`.

Consulte Varias versiones de Python.

¿Cómo localiza una instancia de Hybrid Runbook Worker al intérprete de Python?

La búsqueda del módulo de Python se controla mediante variables de entorno, como se explicó anteriormente.

¿Se admite Python 3 en el control de código fuente?

No. Actualmente, el control de código fuente no se admite para Python 3. De forma predeterminada, los runbooks de Python se sincronizan como runbooks de Python 2.

¿Cómo puede un autor de runbook saber qué paquetes de Python están disponibles en un espacio aislado de Azure?

Consulte Identificación de los paquetes disponibles en el espacio aislado.

¿Cómo puede establecer un autor de runbook qué versión de un módulo de paquetes se va a usar si hay varios módulos?

Consulte Administración de paquetes de Python 3.

Pasos siguientes

Si su pregunta no se ha respondido aquí, puede consultar las fuentes siguientes para preguntas y respuestas adicionales.

- Azure Automation
- Foro de comentarios

Comentarios

¿Le ha resultado útil esta página? Sí No

Proporcionar comentarios sobre el producto | Obtener ayuda en Microsoft Q&A

Novedades de Azure Automation

Azure Automation recibe mejoras de forma continua. Para mantenerse al día de los avances más recientes, este artículo proporciona información acerca de los elementos siguientes:

- Versiones más recientes
- Nuevas características
- Mejoras de las características existentes
- Problemas conocidos
- Corrección de errores

Esta página se actualiza mensualmente, por lo que se recomienda visitarla con frecuencia. Si busca elementos que tengan más de seis meses, puede encontrarlos en el Archivo de novedades de Azure Automation.

Julio de 2026

Disponibilidad general: Azure Automation soporta PowerShell 7.6

Azure Automation anuncia la disponibilidad general de PowerShell 7.6 en todas las regiones públicas. Las versiones de PowerShell 7.1 y 7.2 y Python 2.7 y 3.8 se anuncian retiradas por productos primarios de PowerShell y Python, respectivamente, y, por tanto, no se recomienda. Más información.

Aviso de retirada - El soporte para Python-2.7 y 3.8 y PowerShell 7.1 y 7.2 se retirará el 30 de septiembre de 2026.

Las versiones en tiempo de ejecución Python 2.7, Python 3.8 y PowerShell 7.1 y 7.2 dejarán de ser soportadas por Automation después del 30 de septiembre de 2026. Los libros de ejecución que usan estas versiones pueden seguir funcionando, pero no recibirán actualizaciones de seguridad, corrección de errores ni soporte oficial. La actualización es necesaria para mantener la elegibilidad para el soporte y garantizar el acceso continuo a las actualizaciones de seguridad y a las funciones de la plataforma. Más información

Marzo de 2026

Actualización de seguridad: extensión de Windows Hybrid Worker de Azure Automation (1.3.74)

La extensión de Windows Hybrid Worker versión 1.3.74 incluye una mejora de seguridad que refuerza los controles de acceso para la comunicación con hybrid Instance Metadata Service (HIMDS).

En condiciones específicas de tiempo de inicio, la extensión de Windows Hybrid Worker intenta conectarse a HIMDS antes de que el servicio se inicialice por completo. En esta ventana estrecha, un proceso local sin privilegios puede suplantar el punto de conexión del servicio de metadatos y obtener acceso de lectura no autorizado a metadatos protegidos o información de configuración.

Esta versión soluciona el problema agregando una validación adicional para asegurarse de que solo se conecta a puntos de conexión de metadatos de confianza propiedad del sistema. Estos cambios mejoran la protección de los metadatos del sistema y la información de configuración al garantizar que el acceso está limitado a los componentes del sistema de confianza.

Agosto de 2025

Reanudación de la implementación: límites de suscripción y servicio revisados de Azure Automation

Después de una breve pausa, Azure Automation reanuda las implementaciones de los límites de servicio y suscripción revisados desde el 11 de agosto de 2025. Esta actualización garantizará una distribución equitativa de los recursos de la nube en todos los clientes y mejorará aún más la fiabilidad y el rendimiento del servicio. Dado que los requisitos de recursos varían según las organizaciones y evolucionan con el tiempo, estamos permitiendo a los clientes configurar sus cuotas en función del uso real. Puede encontrar la lista completa de límites de Azure Automation aquí. Puede comprobar el uso y los límites actuales de las cuentas de Automation y los trabajos que se ejecutan simultáneamente, y solicitar cambios creando una solicitud de soporte técnico en la categoría Límites de servicio y suscripción (cuotas) -> Azure Automation. Más información.

Junio de 2025

Disponibilidad general: Azure Automation admite runbooks de PowerShell 7.4 y Python 3.10

Azure Automation anuncia la disponibilidad general de runbooks de PowerShell 7.4 y Python 3.10 en todas las regiones públicas, excepto en el sudeste de Brasil y las nubes gubernamentales. Las

versiones de PowerShell 7.1 y 7.2 y Python 2.7 y 3.8 se anuncian retiradas por productos primarios de PowerShell y Python, respectivamente, y, por tanto, no se recomienda. Más información.

Disponibilidad general: entorno en tiempo de ejecución de Azure Automation

Azure Automation anuncia la disponibilidad general del entorno de ejecución en todas las regiones públicas, excepto las nubes del sudeste de Brasil y las nubes gubernamentales. Puede actualizar sin problemas los runbooks obsoletos de PowerShell 7.1, PowerShell 7.2, Python 2.7 y Python 3.8 a las versiones más recientes del entorno de ejecución, lo que mejora la seguridad. Esta característica proporciona control completo para configurar el entorno de ejecución de scripts, sin preocuparse por las versiones de módulo en conflicto en una sola cuenta de Automation. Todos los runbooks existentes están disponibles automáticamente en la nueva experiencia del entorno de runtime sin ningún esfuerzo manual. Más información.

Disponibilidad general: Azure Automation admite comandos de la CLI de Azure en runbooks de PowerShell 7.4

Azure Automation anuncia la disponibilidad general de comandos de CLI de Azure en los runbooks de PowerShell 7.4 en todas las regiones públicas, excepto el sudeste de Brasil y las nubes gubernamentales. Ahora, aproveche las ventajas combinadas del amplio conjunto de comandos de la CLI de Azure en runbooks de PowerShell de Azure Automation para simplificar la administración de los recursos de Azure.

Diciembre de 2024

Aviso de retirada final: Módulos de AzureRM a partir del 1 de febrero de 2025

Azure Automation **interrumpirá** la ejecución de todos los runbooks que usan módulos de AzureRM. El módulo AzureRM PowerShell se retiró el **29 de febrero de 2024**. Consulte recursos de migración para obtener instrucciones sobre la transición de AzureRM al módulo Az de PowerShell.

Aviso de retirada final: protocolos TLS 1.0/1.1 para Azure Automation

A partir del **01 de marzo de 2025** todos los nodos híbridos de runbook híbridos de usuario, webhooks y DSC basados en agente mediante protocolos de seguridad de la capa de transporte (TLS) 1.0 y 1.1 ya no podrían conectarse a Azure Automation. Se producirá un error en todos los trabajos que se ejecutan o programan en instancias de Hybrid Worker que usan los protocolos TLS 1.0 y 1.1. Más información.

Aviso de retirada final: Hybrid Runbook Worker basado en agente

A partir del **1 de abril de 2025**, se detendrán todos los trabajos basados en agente que se ejecutan en Hybrid Runbook Worker (Windows y Linux). Más información.

Julio de 2024

Disponibilidad general: Azure Automation admite runbooks de PowerShell 7.2 en nubes gubernamentales

Azure Automation ahora admite runbooks de PowerShell 7.2 en nubes gubernamentales. Más información.

Abril de 2024

Cambios en los límites y cuotas de suscripción y servicio de Automatización de procesos

Busque los cambios en los límites y cuotas de Azure Automation aquí. Estos cambios están dirigidos a mejorar la confiabilidad y el rendimiento del servicio al garantizar un acceso justo a los recursos en la nube para todos los usuarios. Se recomienda usar otras regiones u otras suscripciones dentro de la misma geografía de Azure para crear más cuentas de Automation.

Febrero de 2024

Nueva versión de Start/Stop VMs

Start/Stop VM durante las horas fuera de servicio, la versión 1 está en desuso y no está disponible en Marketplace ahora y se recomienda empezar a usar la versión 2, que ahora está disponible con carácter general. La nueva versión de Start/Stop VMs v2 proporciona una opción de automatización descentralizada de bajo costo para los clientes que desean optimizar sus

costos de máquina virtual. Ofrece todas las mismas funcionalidades que la versión original que estaba disponible con Azure Automation. Más información.

Enero de 2024

Versión preliminar pública: Entorno de Azure Automation Runtime y compatibilidad con los comandos de la CLI de Azure en los runbooks

Azure Automation presenta un entorno de runtime (versión preliminar) que proporciona una experiencia sencilla y sin complicaciones para actualizar los scripts a las últimas versiones del lenguaje. También proporciona un control total para configurar el entorno de ejecución del script, sin preocuparse por versiones conflictivas de módulos en una única cuenta de Automation. Todos los runbooks existentes están disponibles automáticamente en la nueva experiencia del entorno de runtime sin ningún esfuerzo manual. Puede navegar sin problemas entre la experiencia antigua y la nueva con un solo clic. Más información.

Además, esta característica permite la compatibilidad con Comandos de la CLI de Azure (versión preliminar) en runbooks de PowerShell 7.2. Aproveche ahora los beneficios combinados del rico conjunto de comandos de la CLI de Azure en los runbooks de Azure Automation para agilizar la administración de los recursos de Azure.

Diciembre de 2023

Restauración de los runbooks de Automation eliminados

Tipo: Nuevo cambio

Ahora, puede restaurar los runbooks eliminados en los últimos 29 días. Más información

Retirada de Azure Update Management y Change Tracking mediante Log Analytics

Tipo: retirada

En **31 de agosto de 2024**, Azure Automation retirará Automation Update Management y Change Tracking mediante Log Analytics. Debe migrar a Administrador de actualizaciones de Azure y seguimiento de cambios e inventario mediante "Azure Monitoring Agent" respectivamente antes de la fecha de desuso.

Noviembre de 2023

Disponibilidad general: Azure Automation admite runbooks de PowerShell 7.2

Azure Automation anuncia la disponibilidad general de runbooks de PowerShell 7.2. Esto le permite crear runbooks en la versión compatible a largo plazo de PowerShell mediante la extensión de Azure Automation para VS Code y ejecutarlos en una plataforma segura y confiable. Más información.

Octubre de 2023

Disponibilidad general: extensión de Automation para Visual Studio Code

Azure Automation ahora proporciona una experiencia de edición avanzada para scripts de PowerShell y Python junto con operaciones de gestión de runbooks. Para obtener más información, vea Características clave y limitaciones.

Disponibilidad general: Seguimiento de Cambios mediante el Agente de Supervisión de Azure

Azure Automation anuncia la disponibilidad general de Change Tracking mediante el agente de Azure Monitoring. Más información.

Retirada de cuentas de ejecución

Tipo: retirada

Las cuentas de ejecución de Azure Automation, incluyendo las cuentas de ejecución clásicas, se retiraron el **30 de septiembre de 2023** y se han reemplazado por identidades administradas. Ya no se podrán crear o renovar cuentas de ejecución a través de Azure Portal. Para obtener más información, consulta la migración de cuentas de Run As existentes a identidades administradas.

Mayo de 2023

Disponibilidad general: runbook de Python 3.8

Azure Automation anuncia la disponibilidad general de los runbooks de Python 3.8. Más información sobre Azure Automation Runbooks y Paquetes Python.

abril de 2023

Actualización de versión preliminar pública: Azure Automation admite runbooks de PowerShell 7.2 y Python 3.10

Azure Automation ha ampliado la compatibilidad con la vista previa pública para los runbooks de PowerShell 7.2 y Python 3.10 en casi todas las regiones públicas excepto Australia Central2, Sur de Corea del Sur, Sur de Suecia, Jio del Centro de la India, Sudeste de Brasil, India Central, India Occidental, Centro de Emiratos Árabes Unidos y nubes gubernamentales. Estos nuevos entornos de ejecución ahora se admiten para trabajos híbridos y en la nube. Obtén más información sobre los runbooks de PowerShell 7.2 y Python 3.10 y la ejecución de trabajos híbridos en nuevas versiones de runtime.

Marzo de 2023

Retirada del usuario Hybrid Runbook Worker basado en agente de Azure Automation

Tipo: Plan de cambio

El **31 de agosto de 2024**, Azure Automation retirará Hybrid Runbook Worker basado en agente (Windows y Linux). Deberá migrar todas las instancias de Hybrid Worker de usuario basado en agente a Hybrid Runbook Worker basado en extensiones (Windows y Linux) antes de la fecha de desuso. Además, a partir del **1 de noviembre de 2023**, no se podrán crear **nuevas** instancias de Hybrid Runbook Worker basado en agente. Más información.

Enero de 2023

Versión preliminar pública de la extensión de Automation para Visual Studio Code

Azure Automation ahora proporciona una extensión de VS Code que permite crear y administrar runbooks. Para obtener más información, consulte Características clave y limitaciones y operaciones de administración de runbooks.

Noviembre de 2022

Disponibilidad general: extensión de Hybrid Runbook Worker de usuario para Azure Automation

Hybrid Worker de usuario permite la ejecución de los scripts directamente en las máquinas para administrar cargas de trabajo de invitado o como puerta de enlace a entornos a los que no se puede acceder desde Azure. Azure Automation anuncia la **disponibilidad general de la extensión de Hybrid Worker de usuario**, que se basa en el marco de extensiones de máquina virtual y proporciona una experiencia de instalación **perfecta e integrada**. Es compatible con máquinas virtuales Azure de Windows y Linux y con servidores habilitados para Azure Arc. También está disponible para máquinas virtuales de VMware vSphere habilitadas para Azure Arc en versión preliminar.

Octubre de 2022

Versión preliminar pública de PowerShell 7.2 y Python 3.10

Azure Automation ahora admite runbooks en las versiones más recientes del entorno de ejecución: PowerShell 7.2 y Python 3.10 en versión preliminar pública. Esto permite la creación y ejecución de runbooks para la orquestación de tareas de administración. Actualmente, estos nuevos entornos de ejecución solo se admiten para trabajos en la nube en cinco regiones: Centro-oeste de EE. UU., Este de EE. UU., Norte de Sudáfrica, Norte de Europa, Australia y Sudeste. Más información.

Guía para la recuperación ante desastres de una cuenta de Azure Automation

Configure la recuperación ante desastres para las cuentas de Automation para controlar un error en toda la región o en toda la zona. Más información.

Septiembre de 2022

Compatibilidad con zonas de disponibilidad para Azure Automation

Azure Automation ahora admite zonas de disponibilidad de Azure para proporcionar una alta disponibilidad mejorada de resistencia y fiabilidad al servicio, runbooks y otros recursos de automatización. Más información.

Agosto de 2022

La extensión Azure Automation Hybrid Worker (versión preliminar) ahora admite máquinas virtuales de VMware habilitadas para Arc

Tipo: Mejora de una característica existente

Además de la compatibilidad con máquinas virtuales de Azure y servidores habilitados para Arc, la extensión Azure Automation Hybrid Worker (versión preliminar) ahora admite máquinas virtuales de VMware habilitadas para Arc como destino. Ahora puede orquestar tareas de administración mediante PowerShell y runbooks de Python en máquinas virtuales de Azure, servidores habilitados para Arc y máquinas virtuales de VMware habilitadas para Arc con una experiencia idéntica. Aquí puede encontrar más información al respecto.

Julio de 2022

Compatibilidad con cuentas de ejecución

Tipo: Plan de cambio

La cuenta Run As de Azure Automation se retirará el 30 de septiembre de 2023 y será reemplazada por identidades administradas. Antes de esa fecha, deberá empezar a migrar los runbooks para usar identidades administradas. Para obtener más información, vea Migración desde una cuenta de ejecución existente a una identidad administrada a fin de empezar a migrar los runbooks de la cuenta de ejecución a identidades administradas antes del 30 de septiembre de 2023.

Marzo de 2022

Reenviar los datos de auditoría de diagnóstico a los registros de Azure Monitor

Tipo: Nueva característica

Azure Automation puede enviar registros de auditoría de diagnóstico, además del estado de un trabajo del runbook y de flujos de trabajos, al área de trabajo de Log Analytics. Aquí puede encontrar más información al respecto.

Febrero de 2022

Los permisos cambian en el rol de Lector integrado para la cuenta de Automation.

Tipo: Nuevo cambio

Para reforzar la posición de seguridad general de Azure Automation, el rol de Lector de RBAC integrado no tendría acceso a las claves de cuenta de Automation a través de la llamada API: `GET /automationAccounts/agentRegistrationInformation`. Aquí puede encontrar más información al respecto.

Restauración de las cuentas de Automation eliminadas

Tipo: Nuevo cambio

Los usuarios ya pueden restaurar cuentas de Automation eliminadas en un plazo máximo de 30 días. Aquí puede encontrar más información al respecto.

Diciembre de 2021

Nuevos scripts agregados para la administración de máquinas virtuales de Azure en función de la alerta de Azure Monitor

Tipo: Nueva característica

Se agregan nuevos scripts a la organización de GitHub de Azure Automation para abordar uno de los escenarios clave de Azure Automation de administración de máquinas virtuales, en función de la alerta de Azure Monitor. Para obtener más información, consulte Desencadenamiento de un runbook desde una alerta de Azure.

- Stop-Azure-VM-On-Alert
- Restart-Azure-VM-On-Alert
- Delete-Azure-VM-On-Alert
- ScaleDown-Azure-VM-On-Alert
- ScaleUp-Azure-VM-On-Alert

Noviembre de 2021

Disponibilidad general de la identidad administrada para Azure Automation

Tipo: Nueva característica

Azure Automation ahora es compatible con Identidades administradas en Azure public, Azure Gov y Microsoft Azure operado por 21Vianet. Las identidades administradas asignadas por el sistema se admiten tanto en la nube como en los trabajos híbridos, mientras que las identidades administradas asignadas por el usuario solo se admiten para los trabajos en la nube. Para más información, lea este anuncio.

Compatibilidad con la versión preliminar de PowerShell 7.1

Tipo: Nueva característica

La compatibilidad de Azure Automation con los runbooks de PowerShell 7.1 está disponible como versión preliminar pública en las nubes públicas de Azure, Azure Gov y Azure operada por 21Vianet. Para obtener más información, consulte los tipos de runbook de Azure Automation y el anuncio.

Octubre de 2021

Compatibilidad en versión preliminar con la extensión Hybrid Runbook Worker para máquinas virtuales de Azure y servidores habilitados para Arc

Tipo: Nueva característica

Azure Automation ha liberado la integración nativa de User Hybrid Runbook Worker no solo para máquinas virtuales de Azure, sino también para máquinas que no son de Azure, a través de servidores habilitados para Arc. Para más información, consulte Información general de Automation Hybrid Runbook Worker y anuncio.

Compatibilidad en versión preliminar con la autenticación de Azure Active Directory

Tipo: Nueva característica

Azure Automation agregó una característica de seguridad crítica con la compatibilidad de la autenticación de Azure AD con todos los puntos de conexión públicos del servicio Automation.

La característica se ha implementado a través de la compatibilidad con la extensión Hybrid Runbook Worker para máquinas virtuales de Azure y servidores habilitados para Arc. Más información

Así se elimina la dependencia de los certificados y se permite cumplir los estrictos requisitos de cumplimiento y auditoría, ya que no se usan métodos de autenticación locales. Para más información, lea este anuncio .

Control de código fuente habilitado para usar identidades administradas

Tipo: Nueva característica

La integración del control de código Azure Automation ahora puede usar identidades administradas, en lugar de una cuenta de ejecución. Para más información, consulte los requisitos previos de la integración del control de código fuente.

Septiembre de 2021

Compatibilidad predeterminada con los módulos de Az

Tipo: Nueva característica

Azure Automation ahora admite los módulos Az de forma predeterminada. Las nuevas cuentas de Automation creadas incluyen la versión más reciente de los módulos Az: 6.4.0 de forma predeterminada. Automation también incluye una opción en Azure Portal, Actualizar módulos Az, que le permite actualizar módulos Az en las cuentas de Automation existentes. Para más información, lea este anuncio .

Agosto de 2021

Configuración de invitado de Azure Policy

Tipo: Plan de cambio

Los clientes deben evaluar y planear la migración de State Configuration de Azure Automation a la configuración de invitado de Azure Policy. Para más información, consulte Configuración de invitado de Azure Policy.

Julio de 2021

Compatibilidad en versión preliminar con la identidad administrada asignada por el usuario

Tipo: Nueva característica

Azure Automation ahora admite Identidades administradas asignadas por el usuario para trabajos en la nube en las regiones Azure global, Azure Government y Azure operada por 21Vianet. Para más información, lea este anuncio.

Disponibilidad general de claves administradas por el cliente para Azure Automation

Tipo: Nueva característica

Los clientes pueden administrar y proteger el cifrado de los recursos de Azure Automation mediante sus propias claves administradas. Con la introducción de las claves administradas por el cliente puede complementar el cifrado predeterminado con una capa extraordinaria de cifrado mediante una clave que se crea y administra en Azure Key Vault. Este nivel adicional de cifrado debería ayudarle a satisfacer las necesidades normativas o de cumplimiento de su organización.

Para más información, consulte Uso de claves administradas por el cliente.

Junio de 2021

Actualización de seguridad para la función de colaborador de Log Analytics

Tipo: Plan de cambio

Microsoft quiere quitar los derechos de la cuenta de Automation del rol Colaborador de Log Analytics. Actualmente, el rol integrado Colaborador de Log Analytics puede escalar los privilegios al rol Colaborador de la suscripción. Como las cuentas Run As de Automation se configuran inicialmente con derechos de Colaborador en la suscripción, un atacante puede usarlas para crear nuevos runbooks y ejecutar código con permisos de Colaborador en la suscripción.

Debido a este riesgo de seguridad, se recomienda no utilizar el rol Colaborador de Log Analytics para ejecutar trabajos de Automation. En su lugar, cree el rol personalizado Colaborador de Azure Automation y utilícelo para las acciones relacionadas con la cuenta de Automation.

Compatibilidad con Automation y State Configuration disponible en Oeste de EE. UU. 3

Tipo: Nueva característica

Para obtener información sobre la disponibilidad regional, consulte Residencia de datos en Azure y seleccione su geografía en la lista desplegable. Más información.

Mayo de 2021

Iniciar/Detener VMs durante horas no laborales (v1)

Tipo: Plan de cambio

Start/Stop VMs during off-hours (v1) caerá en desuso el 21 de mayo de 2022. Los clientes deben evaluar y planificar la migración a Start/Stop VMs v2 (versión preliminar). Para más información, consulte Información general sobre Start/Stop v2 (versión preliminar).

Abril de 2021

Compatibilidad con Update Management y Change Tracking

Tipo: Nueva característica

La asignación de regiones se ha actualizado para admitir Update Management y Change Tracking en el Este de Noruega, el Norte de Emiratos Árabes Unidos, el Centro-norte de EE. UU., el Sur de Brasil y el Centro de Corea. Para obtener más información, consulte Asignaciones admitidas.

Compatibilidad con identidades administradas asignadas por el sistema

Tipo: Nueva característica

Ahora, Azure Automation admite identidades administradas asignadas por el sistema para trabajos híbridos y en la nube en las regiones Azure global y Azure Government. Para más información, lea este anuncio .

Pasos siguientes

Para colaborar en la documentación de Azure Automation, consulte nuestra guía del colaborador.

Last updated on 27/07/2026

Archivo de novedades de Azure Automation

> **Precaución**
>
> Este artículo hace referencia a CentOS, una distribución de Linux que ha alcanzado el estado de finalización del servicio (EOL). Tenga en cuenta su uso y planificación en consecuencia. Para más información, consulte la Guía de fin de ciclo de vida de CentOS.

El artículo principal ¿Qué hay de nuevo en Azure Automation? contiene actualizaciones de los últimos seis meses, mientras que este artículo contiene toda la información más antigua.

¿Qué hay nuevo en Azure Automation? le proporciona información sobre:

- Versiones más recientes
- Problemas conocidos
- Corrección de errores

Marzo de 2021

Nuevas definiciones de directivas integradas de Azure Automation

Tipo: Nueva característica

Azure Automation ha agregado cinco nuevas definiciones de directiva integradas:

- Las cuentas de Automation deben deshabilitar el acceso a la red pública.
- Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en descanso.
- Configuración de cuentas de Azure Automation para deshabilitar el acceso a la red pública
- Configuración de conexiones de punto de conexión privado en cuentas de Azure Automation
- Las conexiones de punto de conexión privado en cuentas de Automation deben estar habilitadas.

Para obtener más información, vea la referencia de Azure Policy.

Compatibilidad con la disponibilidad general declarada de Automation y State Configuration en Sur de la India

Tipo: Nueva característica

Utilice la característica Automatización de procesos y Configuración de estado en el sur de la India. Para más información, lea este anuncio.

Compatibilidad con la disponibilidad general declarada de Automation y State Configuration en Oeste de Reino Unido

Tipo: Nueva característica

Use la característica Automatización de procesos y Configuración de Estado en Reino Unido Oeste. Para más información, lea el anuncio.

Compatibilidad con la disponibilidad general declarada de Automation y State Configuration en Centro de Emiratos Árabes Unidos

Tipo: Nueva característica

Utiliza la característica Automatización de procesos y Configuración de estado en el centro de Emiratos Árabes Unidos. Para más información, lea este anuncio.

Compatibilidad con Automation y State Configuration disponible en Centro de Australia 2, Oeste de Noruega y Sur de Francia

Tipo: Nueva característica

Para más información sobre la página Residencia de datos, seleccione la ubicación geográfica de cada región.

Nuevos scripts agregados para instalar Hybrid Worker en Windows y Linux

Tipo: Nueva característica

Se han agregado dos nuevos scripts al repositorio Azure Automation GitHub abordando uno de los escenarios clave de Azure Automation de configuración de Hybrid Runbook Worker en una Windows o en una máquina Linux. El script crea una nueva máquina virtual o usa una existente, crea un área de trabajo de Log Analytics si es necesario, instala el agente de Log Analytics para Windows o Log Analytics agente para Linux y registra la máquina en el área de trabajo de Log Analytics. El script de Windows se denomina **Create Automation Windows HybridWorker** y el script de Linux es **Create Automation Linux HybridWorker**.

Invocación de runbook mediante un webhook de una plantilla de Azure Resource Manager

Tipo: Nueva característica

Para más información, consulte Uso de un webhook de una plantilla de ARM.

Azure Update Management ahora admite CentOS 8.x, Red Hat Enterprise Linux Server 8.x y SUSE Linux Enterprise Server 15

Tipo: Nueva característica

Consulte la lista completa de sistemas operativos Linux compatibles para más información.

Compatibilidad con la residencia de datos en la región para las regiones Sur de Brasil y Sudeste de Asia

Tipo: Nueva característica

En todas las regiones excepto Sur de Brasil y Sudeste de Asia, los datos de Azure Automation se almacenan en otra región (Azure región emparejada) para proporcionar continuidad empresarial y recuperación ante desastres (BCDR). Solo para las regiones de Brasil y Sudeste de Asia, ahora almacenamos Azure Automation datos en la misma región para dar cabida a los requisitos de residencia de datos para estas regiones. Para obtener más información, consulte Residencia de datos.

Febrero de 2021

Compatibilidad con la disponibilidad general declarada de Automation y State Configuration en Japón Occidental

Tipo: Nueva característica

Disponibilidad de la cuenta de Automation y de State Configuration en la región Japón Occidental. Para más información, lea el anuncio.

Se ha presentado el cumplimiento de Azure Policy personalizado para aplicar la ejecución de runbooks en Hybrid Worker

Tipo: nueva característica

Puede usar la nueva regla de cumplimiento de Azure Policy para permitir la creación de trabajos, webhooks y programaciones de trabajos para que solo se ejecuten en grupos de Hybrid Worker. Lea los detalles en Utilizar la política de Azure para imponer la ejecución de trabajos en Hybrid Runbook Worker

Disponibilidad de Update Management en las regiones Este de EE. UU., Centro de Francia y Norte de Europa

Tipo: Nueva característica

La característica Update Management de Automation está disponible en las regiones Este de EE. UU., Centro de Francia y Norte de Europa. Vea Asignación de regiones admitidas para obtener actualizaciones de la documentación en la que se refleja este cambio.

Enero de 2021

Compatibilidad con la disponibilidad general declarada de Automation y State Configuration en Oeste de Suiza

Tipo: Nueva característica

Disponibilidad de la cuenta de Automation y de State Configuration en la región Oeste de Suiza. Para más información, lea el anuncio.

Se ha agregado Python 3 script para importar el módulo con varias dependencias.

Tipo: Nueva característica

El script está disponible para su descarga desde nuestro repositorio GitHub.

Compatibilidad con roles de Hybrid Runbook Worker para CentOS 8.x, RHEL 8.x y SLES 15

Tipo Nueva característica

La característica Hybrid Runbook Worker admite las distribuciones CentOS 8.x, REHL 8.x y SLES 15 solo para la automatización de procesos en instancias de Hybrid Runbook Worker.

Disponibilidad de Update Management y Change Tracking en las regiones Este de Australia, Este de Asia, Oeste de EE. UU. y Centro de EE. UU.

Tipo: Nueva característica

La cuenta de automatización, Seguimiento de Cambios e Inventario, y Gestión de Actualizaciones están disponibles en las regiones de Este de Australia, Este de Asia, Oeste de EE. UU. y Centro de EE. UU.

Se introdujo una versión preliminar pública de runbooks de Python 3 en la nube de la Administración Pública de EE. UU.

Tipo: Nueva característica: Azure Automation introduce soporte para la vista previa pública de la ejecución de runbooks en Python 3 en la nube y en entornos híbridos en las regiones en la nube del gobierno de EE. UU. Para obtener más información, consulte el anuncio.

Runbooks de Azure Automation migrados desde el Centro de scripts de TechNet a GitHub

Tipo: Plan de cambio

El Centro de scripts de TechNet se está retirando y todos los runbooks hospedados en la Galería de runbooks se han migrado a la organización GitHub de Automation. Para obtener más información, lea Azure Automation Runbooks migrando a GitHub.

Diciembre de 2020

Disponibilidad general de Azure Automation y Administración de actualizaciones Private Link

Tipo: Nueva característica

La compatibilidad de Azure Automation y Administración de actualizaciones se ha anunciado como GA para las nubes globales de Azure y del gobierno. Azure Automation ha habilitado la compatibilidad con Private Link para proteger la ejecución de un runbook en un rol de trabajo híbrido, mediante el uso de Administración de actualizaciones para aplicar revisiones a las máquinas, la invocación de un runbook a través de un webhook y el uso del servicio State Configuration para mantener las máquinas conformes. Para obtener más información, lea soporte de Azure Automation Private Link

Azure Automation clasificado como certificado Grade-C en accesibilidad

Tipo: Nueva característica

Las características de accesibilidad de los productos de Microsoft ayudan a las agencias a abordar los requisitos de accesibilidad globales. En la página blog announcement, busque **Azure Automation** para leer el informe de conformidad de accesibilidad del servicio Automation.

Compatibilidad con GA de Automation y State Configuration en Norte de Emiratos Árabes Unidos

Tipo: Nueva característica

Disponibilidad de la cuenta de Automation y de State Configuration en la región de Norte de Emiratos Árabes Unidos. Para más información, lea el anuncio.

Compatibilidad con GA de Automation y State Configuration en Centro-oeste de Alemania

Tipo: Nueva característica

Disponibilidad de la cuenta de Automation y de State Configuration en la región de Oeste de Alemania. Para más información, lea el anuncio.

Compatibilidad de DSC con Oracle 6 y 7

Tipo: Nueva característica

Administre máquinas Oracle Linux 6 y 7 con Configuración de Estado de Automatización. Consulte Distros de Linux compatibles para ver las actualizaciones de la documentación para reflejar estos cambios.

Versión preliminar pública de runbooks de Python3 en Automation

Tipo: Nueva característica

Azure Automation ahora admite la ejecución de runbooks en la nube y en entornos híbridos con Python 3 en la versión preliminar pública en todas las regiones de la nube global de Azure. Para obtener más información, consulte el anuncio.

Noviembre de 2020

Compatibilidad de DSC con Ubuntu 18.04.

Tipo: Nueva característica

Consulte Supported Linux Distros para ver las actualizaciones de la documentación que refleja estos cambios.

Octubre de 2020

Compatibilidad con GA de Automation y State Configuration en Norte de Suiza

Tipo: Nueva característica

Disponibilidad de la cuenta de Automation y de State Configuration en la región de Norte de Suiza. Para más información, lea el anuncio.

Compatibilidad con GA de Automation y State Configuration en Sureste de Brasil

Tipo: Nueva característica

Disponibilidad de la cuenta de Automation y de State Configuration en la región de Sureste de Brasil. Para más información, lea el anuncio.

Disponibilidad de Update Management en Centro-sur de EE. UU.

Tipo: Nueva característica

Se ha actualizado la asignación de regiones de Azure Automation para que la característica Update Management se pueda usar en la región Centro-sur de EE. UU. Consulte el apartado sobre la asignación de regiones admitidas para conocer las actualizaciones de la documentación para reflejar este cambio.

Septiembre de 2020

Se han actualizado los runbooks con Start/Stop VMs during off-hours para usar módulos de Azure Az

Tipo: Nueva característica

Los runbooks de Máquina Virtual Start/Stop se han actualizado para usar los módulos de Az en lugar de los módulos de Azure Resource Manager.

Agosto de 2020

Se publicó la extensión DSC para admitir Azure Arc

Tipo: Nueva característica

Use Azure Automation State Configuration para almacenar de forma centralizada las configuraciones y mantener el estado deseado de las máquinas híbridas conectadas habilitadas mediante la extensión de VM DSC en servidores gestionados por Azure Arc. Para más información, lea Descripción general de las extensiones VM para servidores habilitados con Arc.

Julio de 2020

Se introdujo la versión preliminar pública de la compatibilidad con Private Link en Automation

Tipo: Nueva característica

Use Azure Private Link para conectar redes virtuales de forma segura a Azure Automation mediante puntos de conexión privados. Para más información, lea el anuncio.

Compatibilidad de Hybrid Runbook Worker con Windows Server 2008 R2

> **Nota:**
>
> Windows Server 2008 y Windows Server 2008 R2 han alcanzado la finalización del soporte (EOS). Para obtener más información, vea **Fin del soporte técnico para Windows Server 2008 y Windows Server 2008 R2** y **Realizar una actualización local a Windows Server 2016, 2019, 2022 o 2025**. Revise el uso y planee las actualizaciones y migraciones del sistema operativo en consecuencia.

Tipo: Nueva característica

Automation Hybrid Runbook Worker admite el sistema operativo Windows Server 2008 R2.

Compatibilidad de Update Management con Windows Server 2008 R2

Tipo: Nueva característica

Update Management admite la evaluación y aplicación de revisiones del sistema operativo Windows Server 2008 R2. Consulte el apartado sobre sistemas operativos compatibles para conocer las actualizaciones de la documentación que refleja estos cambios.

Actualización del esquema de registros de diagnóstico de Automation

Tipo: Nueva característica

Se cambió el esquema de los datos de registro de Azure Automation en el servicio de Log Analytics. Para más información, consulte Reenviar datos de trabajo de Azure Automation a registros de Azure Monitor.

Azure Lighthouse es compatible con Automation Update Management

Tipo: Nueva característica

Azure Lighthouse habilita la administración de recursos delegados con Update Management para proveedores de servicios y clientes. Obtenga más información aquí.

Junio de 2020

Disponibilidad de Automation y Update Management en la región de US Gov Arizona

Tipo: Nueva característica

La cuenta de Automatización y Gestión de Actualizaciones está disponible en US Gov Arizona. Para obtener más información, consulte el anuncio.

Script de incorporación del Hybrid Runbook Worker actualizado para usar módulos Az

Tipo: Nueva característica

El runbook New-OnPremiseHybridWorker se ha actualizado para que admita módulos Az. Para obtener más información, consulte el paquete en el PowerShell Gallery.

Disponibilidad de Update Management en Este de China 2

Tipo: Nueva característica

Se ha actualizado la asignación de regiones de Azure Automation para que la característica Administración de actualizaciones se pueda usar en la región Este de China 2. Consulte el apartado sobre la asignación de regiones admitidas para conocer las actualizaciones de la documentación para reflejar este cambio.

Mayo de 2020

Registros DNS del servicio Automation actualizados desde direcciones URL específicas de la región a específicas de la cuenta de Automation

Tipo: Nueva característica

Los registros DNS de Azure Automation han sido actualizados para admitir Private Links. Para más información, lea el anuncio.

Funcionalidad agregada para mantener los runbooks de Automation y los scripts de DSC cifrados de forma predeterminada

Tipo: Nueva característica

Además de mejorar la seguridad de los recursos, los runbooks y los scripts de DSC son también cifrados para fortalecer la seguridad de Azure Automation.

Abril de 2020

Retirada de la tarea de monitor de Automation

Tipo: Plan de cambio

Azure Logic Apps ahora es la manera recomendada y compatible de supervisar eventos, programar tareas periódicas y desencadenar acciones. No habrá más inversiones en la funcionalidad de la tarea de Watcher. Para más información, consulte Programación y ejecución de tareas, procesos y flujos de trabajo automatizados y periódicos con Azure Logic Apps.

Marzo de 2020

Compatibilidad con el aislamiento de cómputo de nivel de impacto 5 (IL5) en la nube comercial y gubernamental de Azure

Tipo:

Hybrid Runbook Worker de Azure Automation se puede usar en Azure Government para admitir cargas de trabajo de nivel 5 de impacto. Para más información, consulte la documentación.

Febrero de 2020

Se introdujo compatibilidad con etiquetas de servicio de red virtual de Azure

Tipo: Nueva característica

La compatibilidad de Automation con las etiquetas de servicio permite o deniega el tráfico para el servicio Automation en un subconjunto de escenarios. Para obtener más información, consulte la documentación.

Habilitación del soporte para TLS 1.2 en el servicio de Azure Automation

Tipo: Plan de cambio

Azure Automation admite totalmente TLS 1.2 o posterior y todas las llamadas de cliente (a través de webhooks, nodos DSC y hybrid Worker). TLS 1.1 y TLS 1.0 aún se admiten para la compatibilidad con versiones anteriores con clientes anteriores hasta que los clientes se estandaricen y migren por completo a TLS 1.2.

Enero de 2020

Se introdujo la versión preliminar pública de las claves administradas por el cliente para Azure Automation

Tipo: Nueva característica

Los clientes pueden administrar y proteger el cifrado de Azure Automation recursos mediante sus propias claves administradas. Para más información, consulte Uso de claves administradas por el cliente.

Retirada de las API REST de administración de servicios de Azure (ASM) para Azure Automation

Tipo: Retirada

Las API REST de Azure Service Management (ASM) para Azure Automation se retirarán y dejarán de tener soporte técnico el 30 de enero de 2020. Para más información, vea el anuncio.

Pasos siguientes

Si desea contribuir a la documentación de Azure Automation, consulte nuestra guía para colaboradores.

Last updated on 24/04/2026

Inicio rápido: Creación de una cuenta de Automation mediante Azure Portal

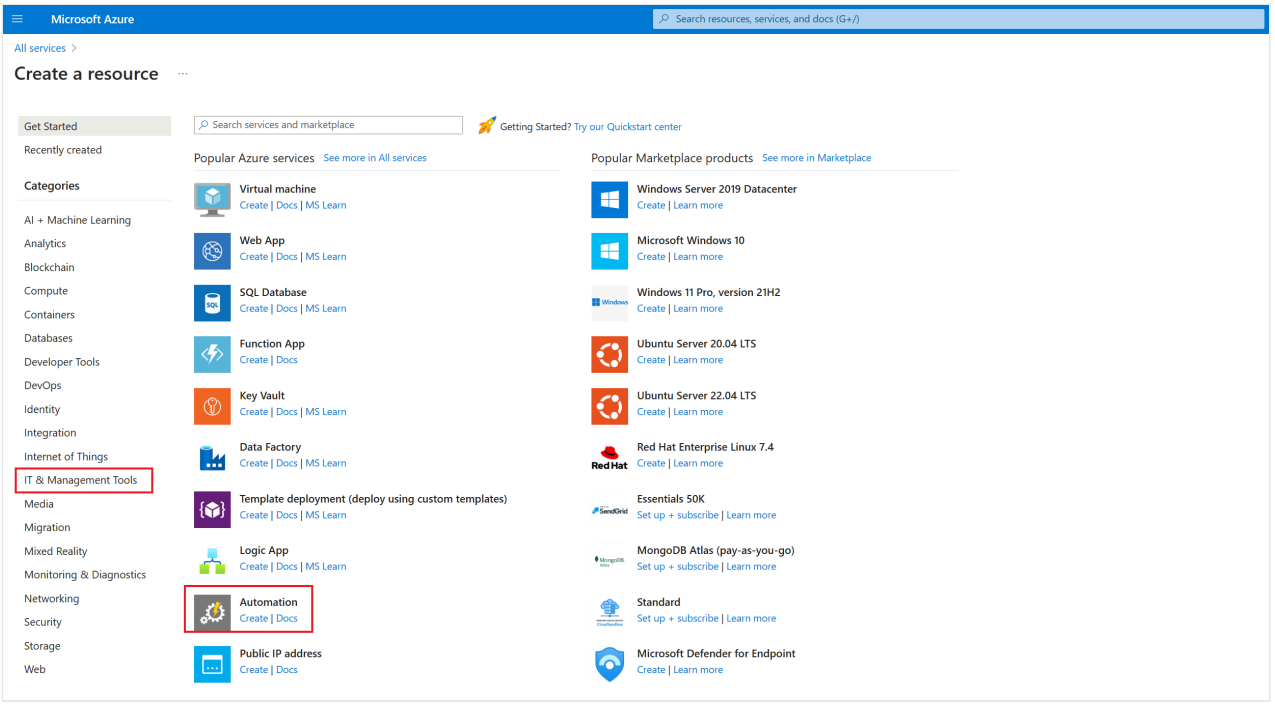
Puede crear una cuenta de Azure Automation mediante Azure Portal, una interfaz de usuario basada en explorador que permita el acceso a varios recursos. Una cuenta de Automation puede administrar recursos en todas las regiones y suscripciones de un inquilino determinado. Este inicio rápido le guía en la creación de una cuenta de Automation.

Prerrequisitos

Una cuenta de Azure con una suscripción activa. Cree una cuenta gratuita .

Creación de una cuenta de Automation

1. Inicie sesión en Azure Portal .
2. En el menú superior, seleccione + **Crear un recurso**.
3. En **Categorías**, seleccione **HERRAMIENTAS de ADMINISTRACIÓN y TI** y, a continuación, seleccione **Automatización**.



Las opciones de la nueva cuenta de Automation se organizan en pestañas en la página **Crear una cuenta de Automation** . En las secciones siguientes se describe cada una de las pestañas y sus opciones.

Aspectos básicos

En la pestaña **Aspectos básicos** , proporcione la información esencial de la cuenta de Automation. Después de completar la pestaña **Aspectos básicos** , puede elegir personalizar aún más la nueva cuenta de Automation estableciendo opciones en las otras pestañas, o bien puede seleccionar **Revisar y crear** para aceptar las opciones predeterminadas y continuar validando y crear la cuenta.

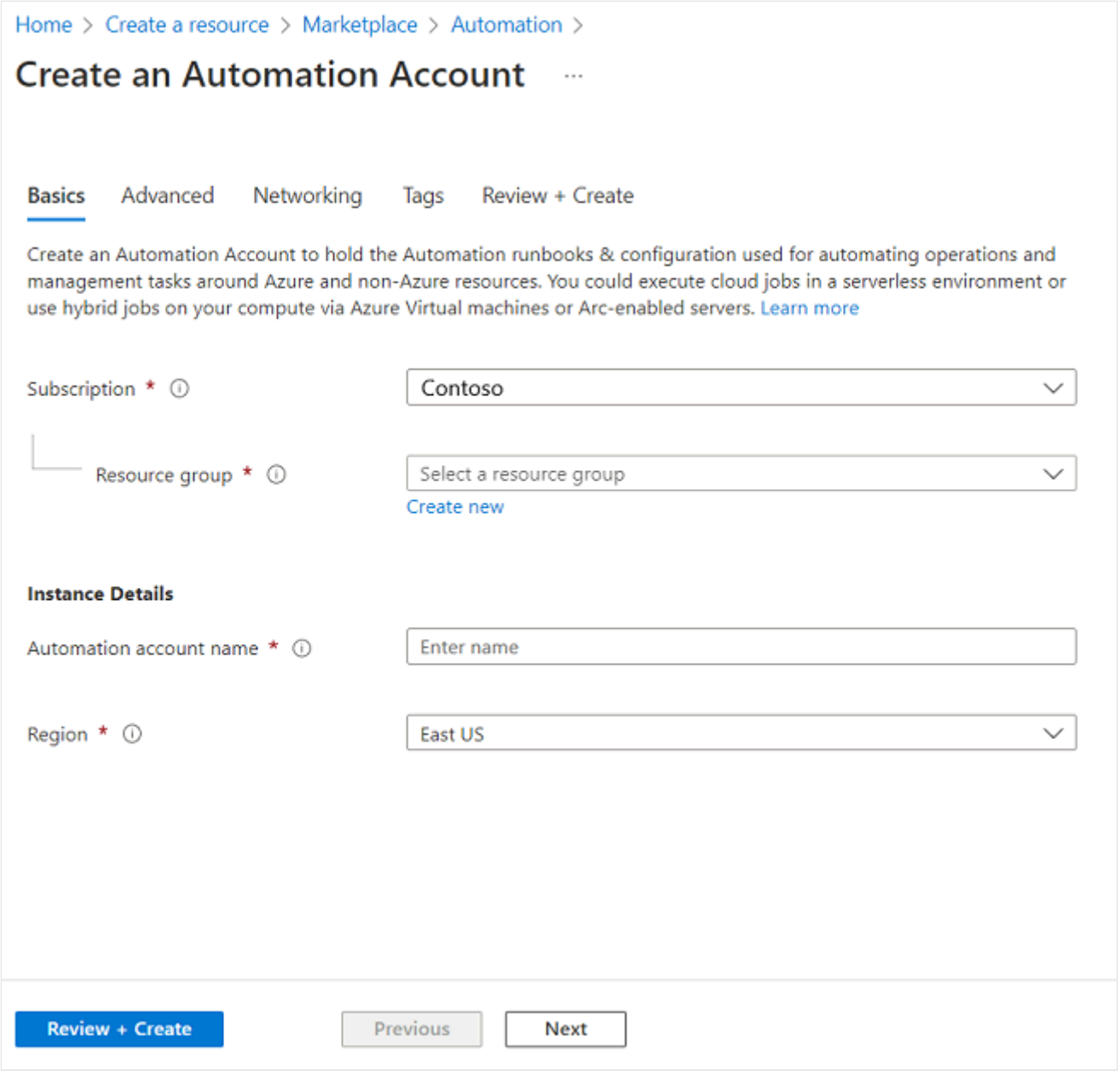
> **Nota:**
>
> De forma predeterminada, se habilita una identidad administrada asignada por el sistema para la cuenta de Automation.

En la siguiente tabla se describen los campos de la pestaña **Aspectos básicos**.

Expandir tabla

Campo	Required o opcional	Descripción
Subscription	Obligatorio	En la lista desplegable, seleccione la suscripción de Azure para la cuenta.
Grupo de recursos	Obligatorio	En la lista desplegable, seleccione el grupo de recursos existente o seleccione **Crear nuevo**.
Nombre de la cuenta de Automation	Obligatorio	Escriba un nombre único para su ubicación y grupo de recursos. Es posible que los nombres de las cuentas de Automation que se hayan eliminado no estén disponibles inmediatamente. No puede cambiar el nombre de la cuenta una vez que se haya escrito en la interfaz de usuario.
Región	Obligatorio	En la lista desplegable, seleccione una región para la cuenta. Para obtener una lista actualizada de ubicaciones en las que puede implementar una cuenta de Automation, consulte Productos disponibles por región .

En la imagen siguiente se muestra una configuración estándar para una nueva cuenta de Automation.



Advanced

En la pestaña **Avanzadas** , puede configurar la opción identidad administrada para la nueva cuenta de Automation. La opción identidad administrada asignada por el usuario también se puede configurar después de crear la cuenta de Automation.

Para obtener instrucciones sobre cómo crear una identidad administrada asignada por el usuario, consulte Creación de una identidad administrada asignada por el usuario.

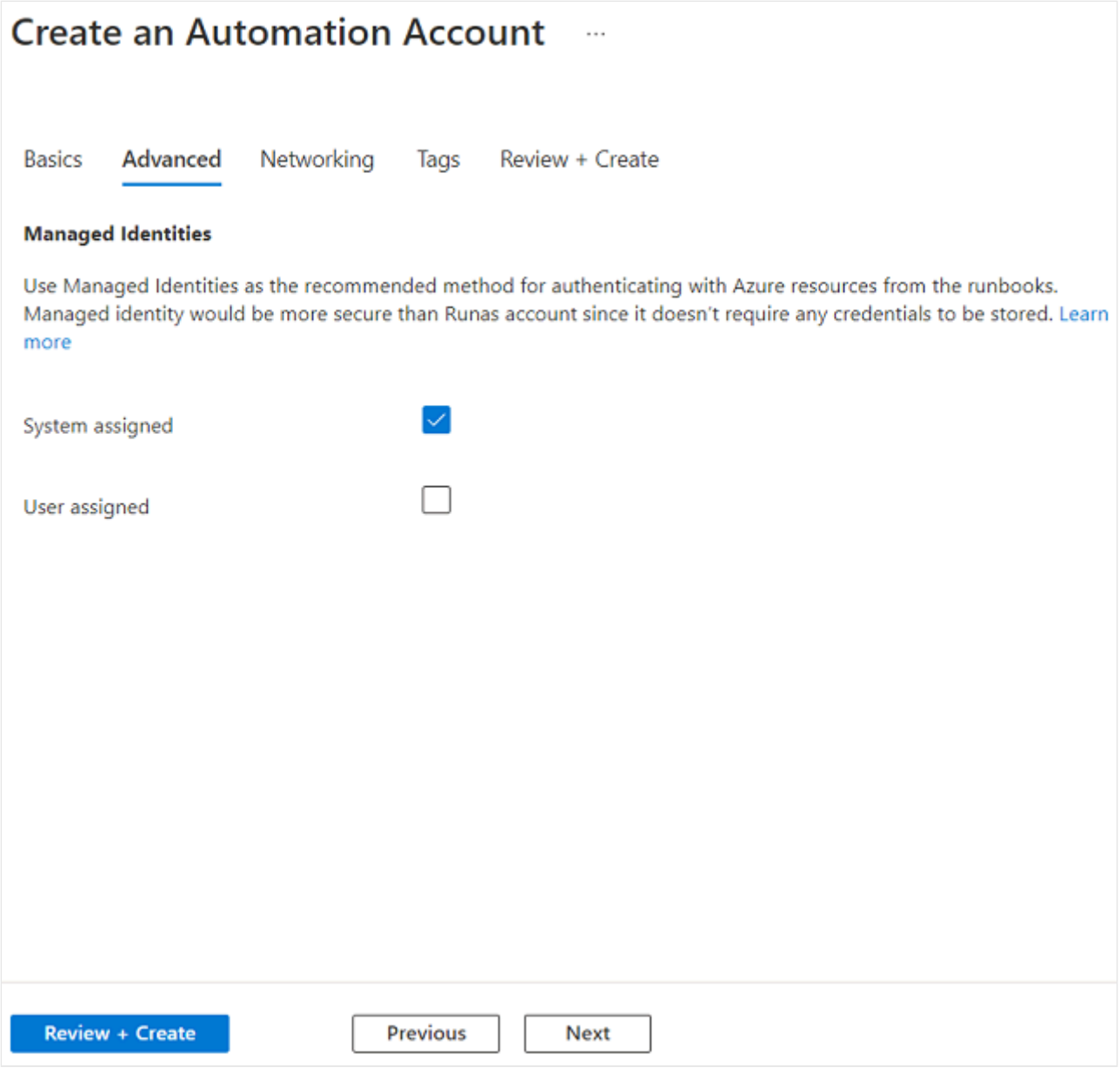
En la siguiente tabla se describen los campos de la ficha **Opciones avanzadas**.

Expandir tabla

Campo	Required o opcional	Descripción
Asignado por el sistema	Opcional	Una identidad de Microsoft Entra que está vinculada al ciclo de vida de la cuenta de Automation.
Asignada por el usuario	Opcional	Una identidad administrada representada como un recurso de Azure independiente que se administra por separado de los recursos que lo usan.

Puede optar por habilitar las identidades administradas más adelante y la cuenta de Automation se crea sin una. Para habilitar una identidad administrada después de crear la cuenta, consulte Habilitación de la identidad administrada. Si selecciona ambas opciones, para la identidad asignada por el usuario, seleccione la opción **Agregar identidades asignadas por el usuario** . En la página **Seleccionar identidad administrada asignada por el usuario** , seleccione una suscripción y agregue una o varias identidades asignadas por el usuario creadas en esa suscripción para asignarlas a la cuenta de Automation.

En la imagen siguiente se muestra una configuración estándar para una nueva cuenta de Automation.



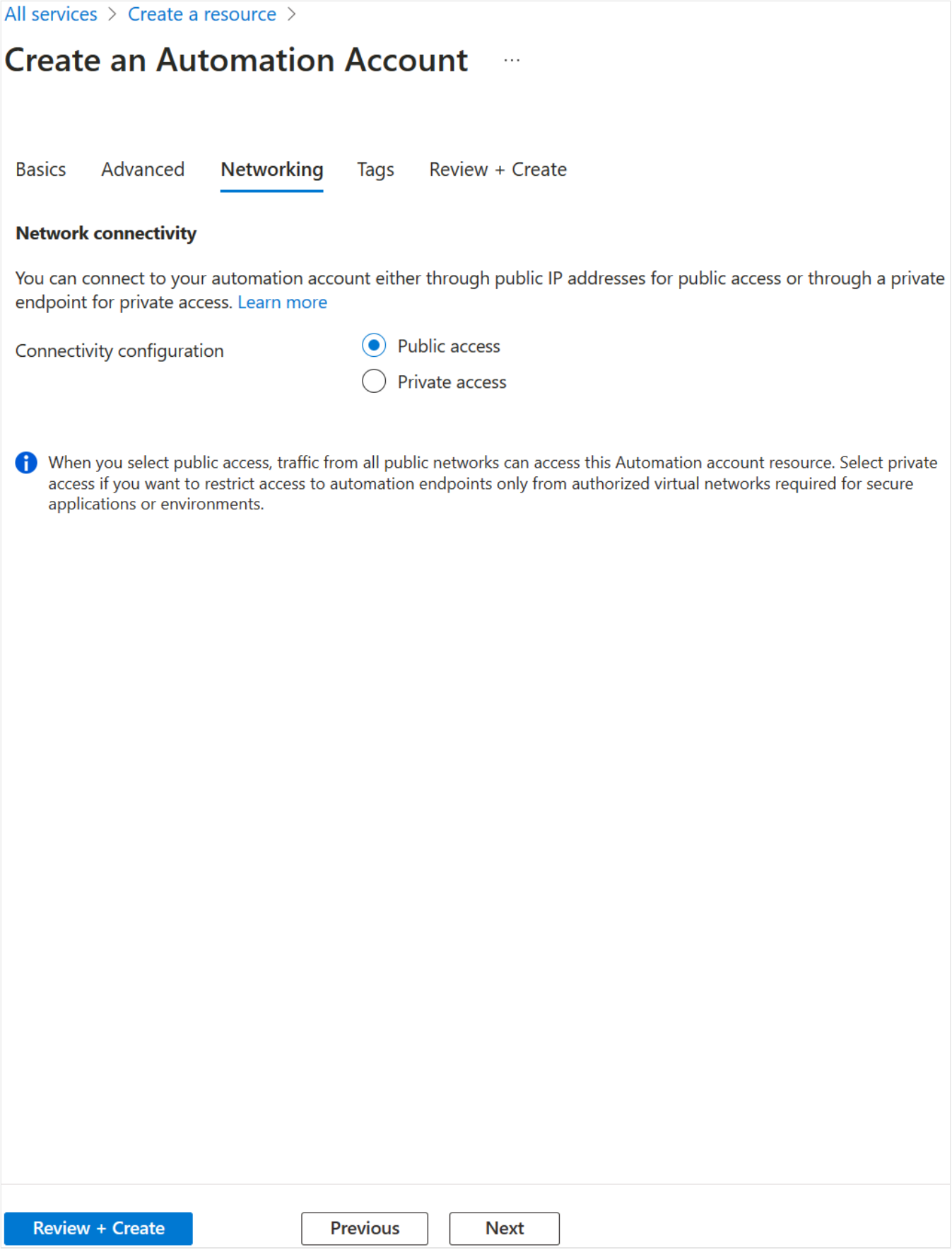
Networking

En la pestaña **Redes** , puede conectarse a la cuenta de Automation públicamente (a través de direcciones IP públicas) o de forma privada mediante un punto de conexión privado. En la imagen siguiente se muestra la configuración de conectividad que puede definir para una nueva cuenta de Automation.

- **Acceso** público: esta opción predeterminada proporciona un punto de conexión público para la cuenta de Automation que puede recibir tráfico a través de Internet y no requiere ninguna configuración adicional. Sin embargo, no se recomienda para aplicaciones privadas ni entornos seguros. En su lugar, la segunda opción **Acceso privado** se puede utilizar para aprovechar un Enlace Privado (mencionado a continuación) con el fin de restringir el acceso a los endpoints de automatización solo desde redes virtuales autorizadas. El acceso público puede coexistir simultáneamente con el punto de conexión privado habilitado en la cuenta de Automation. Si selecciona acceso público al crear la

cuenta de Automation, puede agregar un punto de conexión privado más adelante en la hoja Redes de la cuenta de Automation.

- **Acceso** privado: esta opción proporciona un punto de conexión privado para la cuenta de Automation que usa una dirección IP privada de la red virtual. Esta interfaz de red le conecta de forma privada y segura a la cuenta de Automation. Para incorporar el servicio a la red virtual, habilite un punto de conexión privado. Esta es la configuración recomendada desde un punto de vista de seguridad; Sin embargo, esto requiere que configure Hybrid Runbook Worker conectado a una red virtual de Azure y actualmente no admite trabajos en la nube.



Etiquetas

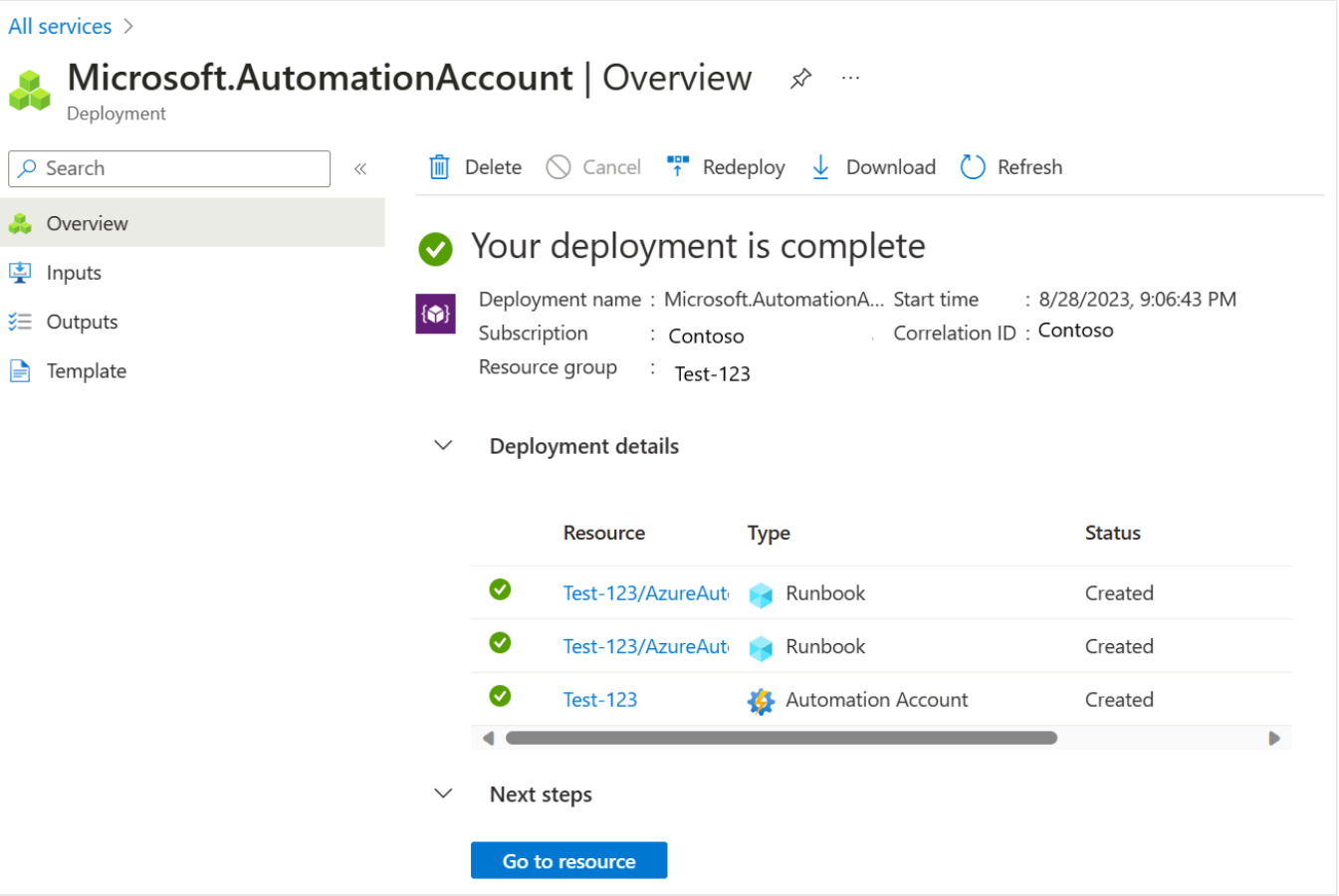
En la pestaña **Etiquetas**, puede especificar etiquetas de Resource Manager para ayudar a organizar los recursos de Azure. Para obtener más información, consulte Etiquetado de recursos, grupos de recursos y suscripciones para una organización lógica.

Pestaña Revisar y crear

Al navegar a la pestaña **Revisar y crear** , Azure ejecuta la validación en la configuración de la cuenta de Automation que ha elegido. Si se supera la validación, puede continuar con la creación de la cuenta de Automation.

En cambio, si se produce un error en la validación, el portal indica qué configuración debe modificarse.

Revise la nueva cuenta de Automation.



Limpieza de recursos

Si no va a seguir usando la cuenta de Automation, seleccione **Eliminar** en la página **Información general** y, a continuación, seleccione **Sí** cuando se le solicite.

Pasos siguientes

En este inicio rápido, ha creado una cuenta de Automation. Para usar identidades administradas con su cuenta de Automation, continúe con el siguiente inicio rápido:

Tutorial: Crear un runbook de PowerShell de Automation usando identidad administrada

Last updated on 17/11/2025

Inicio rápido: Uso de Terraform para crear una cuenta de Azure Automation

Artículo • 17/01/2025

En este inicio rápido, creará una cuenta de Azure Automation y usará Terraform para asignar un rol de "Lector" a la cuenta. Una cuenta de Automation es un servicio basado en la nube que proporciona un entorno seguro para ejecutar runbooks, que son scripts que automatizan procesos. La cuenta puede automatizar tareas frecuentes, lentas y propensas a errores que se administran en la nube. Esta cuenta de Automation se crea dentro de un grupo de recursos de Azure, es un contenedor que almacena los recursos relacionados para una solución de Azure. Además, a la cuenta de Automation se le asigna un rol "Lector" y se le conceden permiso de suscripción para ver todos los recursos de una cuenta de Automation, pero no para realizar ningún cambio.

Terraform habilita la definición, vista previa e implementación de la infraestructura en la nube. Con Terraform, se crean archivos de configuración mediante la sintaxis de HCL. La sintaxis de HCL permite especificar el proveedor de la nube, como Azure, y los elementos que componen la infraestructura de la nube. Después de crear los archivos de configuración, se crea un *plan de ejecución* que permite obtener una vista previa de los cambios de infraestructura antes de implementarlos. Una vez que compruebe los cambios, aplique el plan de ejecución para implementar la infraestructura.

En este artículo aprenderá a:

- ✓ Cree un grupo de recursos de Azure con un nombre único.
- ✓ Genere una cadena aleatoria para la nomenclatura única de los recursos de Azure.
- ✓ Cree una cuenta de Automation y habilite el acceso a la red pública.
- ✓ Recupere la suscripción actual de Azure.
- ✓ Recupere la definición de rol para "Lector".
- ✓ Asigne el rol "Lector" a la cuenta de Automation.
- ✓ Genere los nombres del grupo de recursos y la cuenta de Automation creados.

Requisitos previos

- Cree una cuenta de Azure con una suscripción activa. También puede crear una cuenta de forma gratuita. Las opciones de la nueva cuenta de Automation se organizan en pestañas en la página **Crear una cuenta de Automation** de Azure Portal.

- Instale y configure Terraform.

Implementación del código de Terraform

> **Nota**
>
> El código de ejemplo de este artículo se encuentra en el repositorio de GitHub de Azure Terraform. Puedes ver el archivo de registro que contiene los resultados de las pruebas de las versiones actuales y anteriores de Terraform.
>
> Consulte más artículos y código de ejemplo sobre el uso Terraform para administrar recursos de Azure.

1. Cree un directorio en el que probar y ejecutar el código de ejemplo de Terraform y conviértalo en el directorio actual.

2. Cree un archivo llamado `main.tf` e inserte el siguiente código:

Terraform

```
resource "random_pet" "rg_name" {
  prefix = var.resource_group_name_prefix
}

resource "azurerm_resource_group" "rg" {
  location = var.resource_group_location
  name     = random_pet.rg_name.id
}

resource "random_string" "azurerm_automation_account_name" {
  length  = 13
  lower   = true
  numeric = false
  special = false
  upper   = false
}

resource "azurerm_automation_account" "example" {
  name                = coalesce(var.automation_account_name,
"autoacc-${random_string.azurerm_automation_account_name.result}")
  resource_group_name = azurerm_resource_group.rg.name
  location            = azurerm_resource_group.rg.location
  sku_name            = "Basic"
  identity {
    type         = "SystemAssigned"
  }

  public_network_access_enabled = true
}

data "azurerm_subscription" "current" {}
```

```
data "azurerm_role_definition" "contributor" {
  name = "Contributor"
}

resource "azurerm_role_assignment" "example" {
  scope              = data.azurerm_subscription.current.id
  role_definition_name = "Contributor"
  principal_id       =
azurerm_automation_account.example.identity[0].principal_id
}
```

3. Cree un archivo llamado `outputs.tf` e inserte el siguiente código:

Terraform

```
output "resource_group_name" {
  value = azurerm_resource_group.rg.name
}

output "automation_account_name" {
  value = azurerm_automation_account.example.name
}
```

4. Cree un archivo llamado `providers.tf` e inserte el siguiente código:

Terraform

```
terraform {
  required_version = ">=1.0"

  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~>3.0"
    }
    random = {
      source  = "hashicorp/random"
      version = "~>3.0"
    }
  }
}

provider "azurerm" {
  features {}
}
```

5. Cree un archivo llamado `variables.tf` e inserte el siguiente código:

Terraform

```
variable "resource_group_name_prefix" {
  type        = string
  default     = "rg"
  description = "Prefix of the resource group name that's combined with
a random ID so name is unique in your Azure subscription."
}

variable "resource_group_location" {
  type        = string
  default     = "eastus"
  description = "Location of the resource group."
}

variable "automation_account_name" {
  type        = string
  description = "The name of the Automation Account resource. The value
will be randomly generated if blank."
  default     = ""
}
```

Inicialización de Terraform

Para inicializar la implementación de Terraform, ejecute terraform init. Este comando descarga el proveedor de Azure necesario para administrar los recursos de Azure.

Consola

```
terraform init -upgrade
```

Puntos clave:

- El parámetro `-upgrade` actualiza los complementos de proveedor necesarios a la versión más reciente que cumpla con las restricciones de versión de la configuración.

Creación de un plan de ejecución de Terraform

Ejecute terraform plan para crear un plan de ejecución.

Consola

```
terraform plan -out main.tfplan
```

Puntos clave:

- El comando `terraform plan` crea un plan de ejecución, pero no lo ejecuta. En su lugar, determina qué acciones son necesarias para crear la configuración especificada en los archivos de configuración. Este patrón le permite comprobar si el plan de ejecución coincide con sus expectativas antes de realizar cambios en los recursos reales.
- El parámetro `-out` opcional permite especificar un archivo de salida para el plan. El uso del parámetro `-out` garantiza que el plan que ha revisado es exactamente lo que se aplica.

Aplicación de un plan de ejecución de Terraform

Ejecute terraform apply para aplicar el plan de ejecución a su infraestructura en la nube.

Consola

```
terraform apply main.tfplan
```

Puntos clave:

- El comando `terraform apply` de ejemplo asume que ejecutó `terraform plan -out main.tfplan` previamente.
- Si especificó un nombre de archivo diferente para el parámetro `-out`, use ese mismo nombre de archivo en la llamada a `terraform apply`.
- Si no ha utilizado el parámetro `-out`, llame a `terraform apply` sin ningún parámetro.

Verificación de los resultados

CLI de Azure

1. Obtenga el nombre del grupo de recursos de Azure.

   Consola

   ```
   resource_group_name=$(terraform output -raw resource_group_name)
   ```

2. Obtenga el nombre de la cuenta de Automation.

Consola

```
automation_account_name=$(terraform output -raw
automation_account_name)
```

3. Ejecute az automation account show para ver la cuenta de Automation.

Azure CLI

```
az automation account show --name $automation_account_name --
resource-group $resource_group_name
```

Limpieza de recursos

Cuando ya no necesite los recursos creados a través de Terraform, realice los pasos siguientes:

1. Ejecute el comando terraform plan y especifique la marca `destroy`.

Consola

```
terraform plan -destroy -out main.destroy.tfplan
```

Puntos clave:

- El comando `terraform plan` crea un plan de ejecución, pero no lo ejecuta. En su lugar, determina qué acciones son necesarias para crear la configuración especificada en los archivos de configuración. Este patrón le permite comprobar si el plan de ejecución coincide con sus expectativas antes de realizar cambios en los recursos reales.
- El parámetro `-out` opcional permite especificar un archivo de salida para el plan. El uso del parámetro `-out` garantiza que el plan que ha revisado es exactamente lo que se aplica.

2. Ejecute terraform apply para aplicar el plan de ejecución.

Consola

```
terraform apply main.destroy.tfplan
```

Solución de problemas de Terraform en Azure

Solución de problemas comunes al usar Terraform en Azure.

Pasos siguientes

En este inicio rápido, ha creado una cuenta de Automation. Explore artículos sobre las cuentas de Automation para más información.

Para usar identidades administradas con la cuenta de Automation, continúe con lo siguiente:

Tutorial: Creación de un runbook de PowerShell de Automation mediante una identidad administrada

---

Comentarios

¿Le ha resultado útil esta página? Sí No

Proporcionar comentarios sobre el producto | Obtener ayuda en Microsoft Q&A

Actualización del runbook de PowerShell 7.1 a PowerShell 7.4

24/07/2025

Use el entorno en tiempo de ejecución para actualizar un runbook de una versión en tiempo de ejecución a otra mediante la creación de un entorno en tiempo de ejecución diferente y, a continuación, la vinculación del runbook a este. Puede seguir pasos similares para actualizar runbooks de Python.

> **Nota**
>
> Las versiones 7.1 y 7.2 y Python 2.7 y 3.8 de PowerShell ya no son compatibles con sus respectivos productos primarios powerShell y Python. Se recomienda actualizar los runbooks obsoletos a las versiones más recientes compatibles mediante el entorno en tiempo de ejecución.

Requisitos previos

- Una cuenta de Azure Automation (en cualquier región pública admitida, excepto las regiones Sureste de Brasil y las nubes Gov).
- Un runbook de PowerShell 7.1 en Azure Automation.

Creación de entornos de ejecución

Para crear un entorno en tiempo de ejecución, siga estos pasos:

1. Inicie sesión en Azure Portal y seleccione su cuenta de Automation.

2. En **Automatización de procesos**, seleccione **Entornos de tiempo de ejecución**. Si no encuentra entornos en tiempo de ejecución en la lista, seleccione **Probar experiencia de entorno** en tiempo de ejecución en la página **Información general** para cambiar a la nueva interfaz del portal.

3. Seleccione **Crear** para crear un nuevo entorno de ejecución de PowerShell 7.4.

4. En la pestaña **Aspectos básicos** , proporcione los detalles siguientes:
   a. Escriba **Nombre** para el entorno en tiempo de ejecución. Debe comenzar con una letra y solo puede contener letras, números, caracteres de subrayado y guiones.
   b. En la lista desplegable **Lenguaje**, seleccione **PowerShell**.

c. En **Versión de tiempo de ejecución** para el lenguaje de secuencias de comandos, seleccione 7.4.
d. Escriba la **descripción** adecuada.

5. En la pestaña **Paquetes** , en la lista desplegable **Versión del paquete** , verá que los paquetes **Az** y la **CLI de Azure** predeterminados ya están presentes.

6. Seleccione **+Agregar desde la galería** para agregar más paquetes desde la galería y seleccione **Siguiente**.

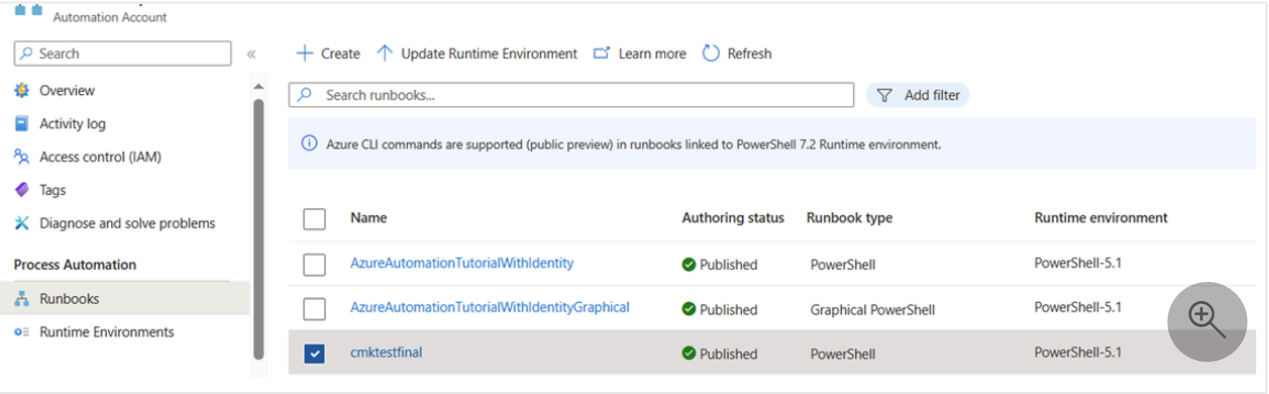
7. En la pestaña **Revisar y crear** , revise las entradas y seleccione **Crear**.

   Aparece una notificación para confirmar que se ha creado correctamente un entorno en tiempo de ejecución.

Actualización del entorno en tiempo de ejecución del runbook

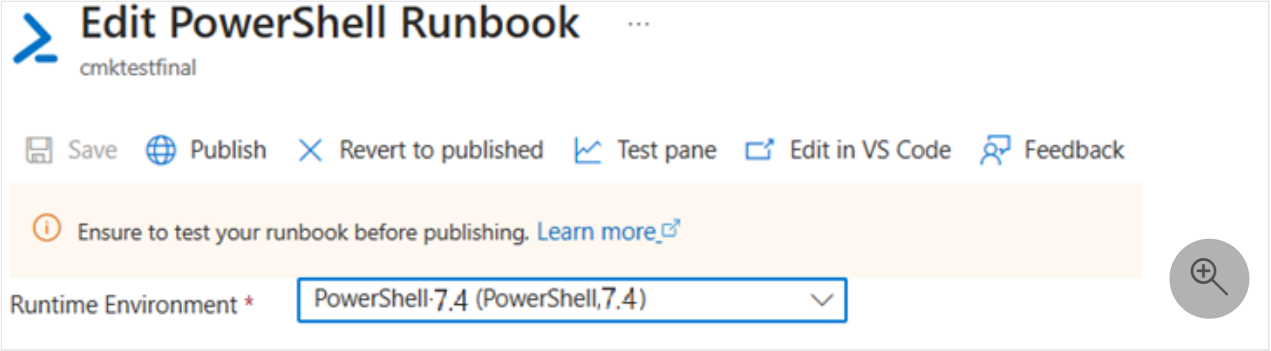
Para actualizar el entorno en tiempo de ejecución del runbook, siga estos pasos:

1. En la página **Cuenta de Automation | Runbooks**, seleccione el runbook vinculado al entorno en tiempo de ejecución de PowerShell 7.1 que desea actualizar.



2. Seleccione **Editar en el portal**.

3. Seleccione **Entorno en tiempo de ejecución** en la lista desplegable para ver la lista de entornos compatibles que puede vincular al runbook. Aquí, seleccione el entorno en tiempo de ejecución que creó en entorno en tiempo de ejecución para PowerShell 7.4.



4. Realice cambios en el código del runbook para garantizar la compatibilidad con PowerShell 7.4.

5. Seleccione **Panel de prueba** para probar el runbook actualizado antes de publicarlo.

6. Después de confirmar que los resultados de la prueba son los esperados, seleccione **Publicar** para publicar el runbook en producción. En la página de lista **Runbooks**, puede ver el mismo runbook vinculado al entorno en tiempo de ejecución de PowerShell 7.4.

Pasos siguientes

- Consulte Administración del entorno en tiempo de ejecución para ver las distintas operaciones a través del portal y la API REST.

Ejecución de comandos de la CLI de Azure en runbooks de PowerShell 7.4

21/07/2025

Puede ejecutar comandos de la CLI de Azure en runbooks vinculados con el entorno en tiempo de ejecución de PowerShell 7.4.

> **Nota**
>
> Los comandos de la CLI de Azure versión 2.64.0 están disponibles como un paquete predeterminado en el entorno en tiempo de ejecución de PowerShell 7.4.

Requisitos previos

- Una cuenta de Azure Automation (en cualquier región pública admitida, excepto las regiones Sureste de Brasil y las nubes Gov).

Creación de entornos de ejecución

Azure Portal

Para crear un entorno en tiempo de ejecución en Azure Portal, siga estos pasos:

1. Inicie sesión en Azure Portal y seleccione su cuenta de Automation.
2. En la página **Información general** , seleccione Probar experiencia del entorno en tiempo de ejecución, si aún no está en la nueva experiencia.
3. En **Automatización de procesos**, seleccione **Entornos en tiempo de ejecución (versión preliminar)** y, a continuación, **Crear**.
4. En la pestaña **Aspectos básicos** , proporcione los detalles siguientes:
   a. Escriba **Nombre** para el entorno en tiempo de ejecución. Debe comenzar con una letra y solo puede contener letras, números, caracteres de subrayado y guiones.
   b. En la lista desplegable **Lenguaje**, seleccione **PowerShell**.
   c. En **Versión de tiempo de ejecución** para el lenguaje de secuencias de comandos, seleccione 7.4.
   d. Escriba la **descripción** adecuada.

5. En la pestaña **Paquetes** , en la lista desplegable **Versión del paquete** , verá que **Az versión 12.3.0** y **la CLI de Azure versión 2.64.0** ya están presentes.

6. Seleccione **+Agregar desde la galería** para agregar más paquetes desde la galería y seleccione **Siguiente**.

7. En la pestaña **Revisar y crear** , revise las entradas y seleccione **Crear**.

   Aparece una notificación para confirmar que se ha creado correctamente un entorno en tiempo de ejecución.

Crear Runbook

Azure Portal

Puede crear un nuevo runbook de PowerShell que admita comandos de la CLI de Azure y esté asociado con el entorno en tiempo de ejecución de PowerShell 7.4.

Para crear un runbook, siga estos pasos:

1. En la cuenta de Automation, en **Automatización de procesos**, seleccione **Runbooks**.

2. Seleccione **Crear**.

3. En la pestaña **Aspectos básicos**, puede crear un nuevo runbook o cargar un archivo desde el equipo local o desde la Galería de PowerShell.
   a. Escriba un **Nombre** para el runbook. El nombre debe empezar por una letra, y debe contener solo letras, números, caracteres de subrayado y guiones.
   b. En la lista desplegable **Tipo de runbook**, seleccione el tipo de runbook que desea crear.
   c. Seleccione Entorno de ejecución de PowerShell 7.4 creado anteriormente.
   d. Escriba la **descripción** adecuada.

4. Agregue código de runbook en la página **Editar runbook** y seleccione **Guardar**.

5. **Pruebe** la ejecución del runbook en el panel de pruebas. Después de confirmar los resultados, seleccione **Publicar** para publicar el runbook y ejecutarlo.

Pasos siguientes

- Consulte Administración del entorno en tiempo de ejecución para ver las distintas operaciones a través del portal y la API REST.

Inicio rápido: Instalación de la extensión de Hybrid Worker en máquinas virtuales mediante Azure Portal

Artículo • 07/03/2024

Azure Automation User Hybrid Worker permite la ejecución de scripts de PowerShell y Python directamente en máquinas para administrar cargas de trabajo invitadas o como puerta de enlace a entornos que no son accesibles desde Azure. Puede configurar máquinas virtuales de Azure Windows y Linux. Servidor habilitado para Azure Arc, máquina virtual VMware vSphere habilitada para Arcy SCVMM habilitado para Azure Arc como Hybrid Worker del usuario mediante la instalación de la extensión Hybrid Worker.

En este inicio rápido se muestra cómo instalar la extensión de Hybrid Worker de Azure Automation en una máquina virtual de Azure a través de la hoja Extensiones de Azure Portal.

Puede seguir pasos similares para instalar la extensión de Hybrid Worker en todos los demás tipos de máquina.

Requisitos previos

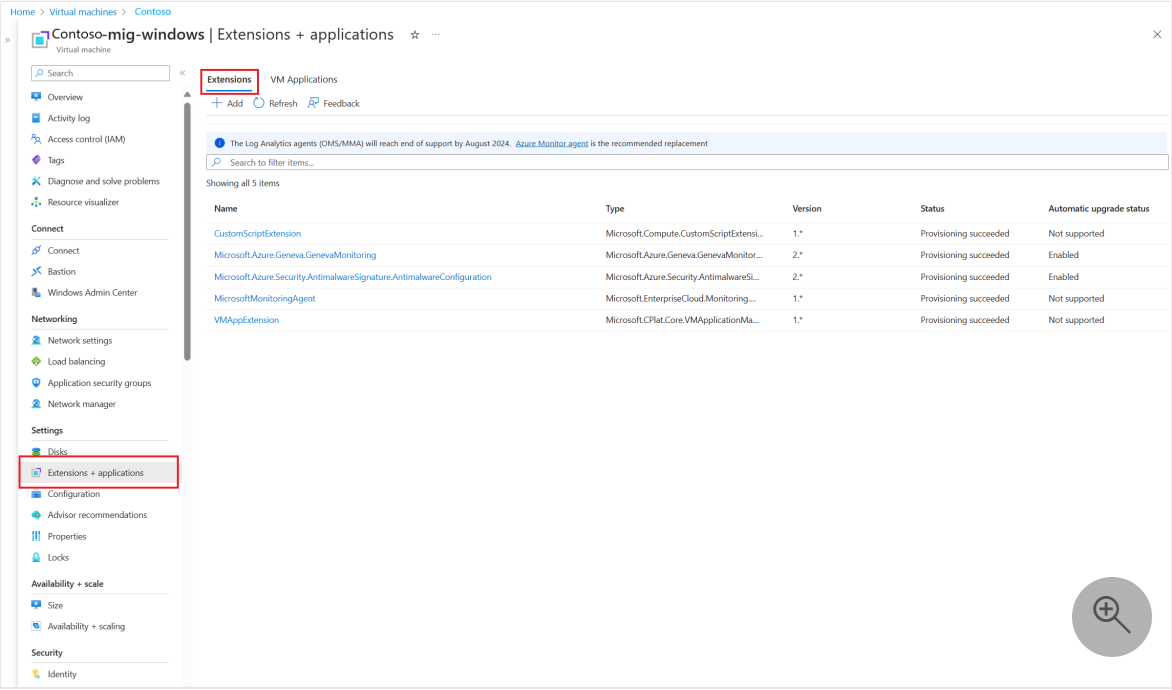
- Una cuenta de Azure Automation en una región admitida.
- Una máquina virtual de Windows Azure en la que se instalaría la extensión Hybrid Worker.
- Habilite la identidad administrada asignada por el sistema en la máquina virtual de Azure. Si la identidad administrada asignada por el sistema no está habilitada, se habilitará como parte del proceso de adición. Para obtener más información, consulte requisitos previos detallados.

Instale la extensión de Hybrid Worker:

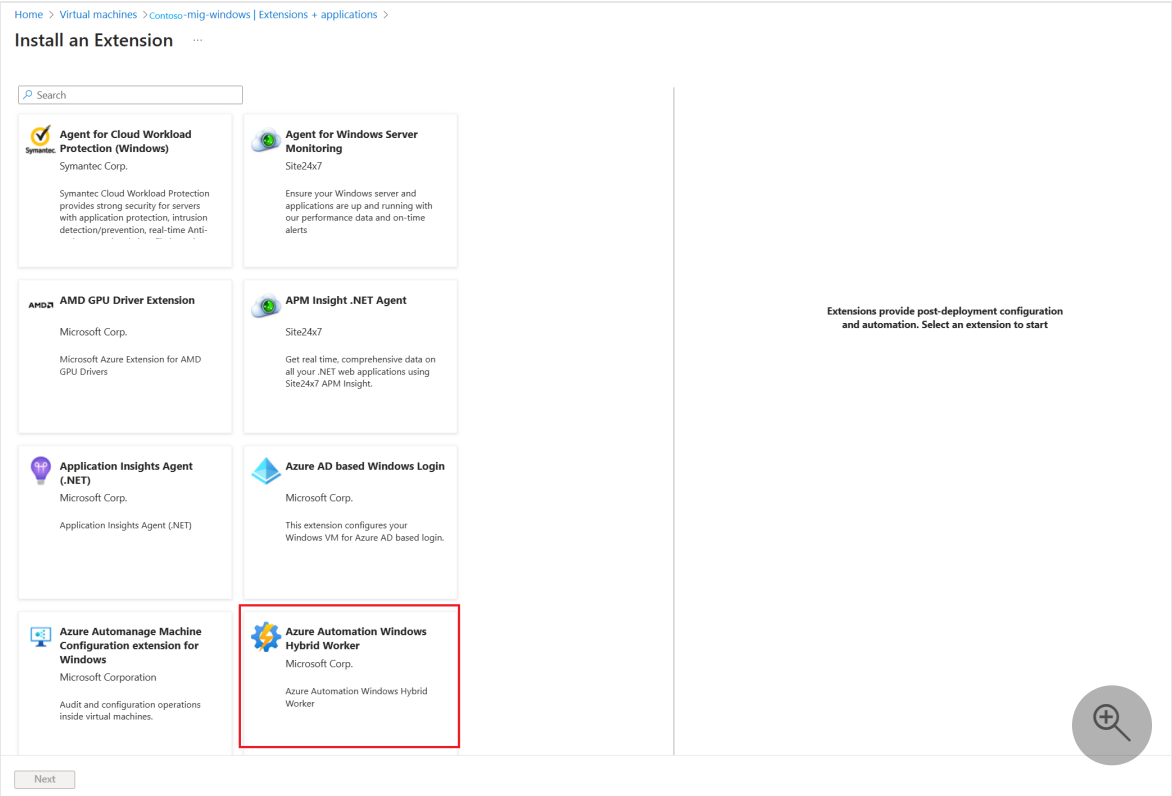
1. Acceda en el Azure Portal y busque Máquinas virtuales.

2. En la página **Máquinas virtuales**, en la lista, seleccione la máquina en la que desea instalar la extensión Hybrid Worker.

3. En **Configuración**, seleccione **Extensiones y aplicaciones** para máquinas virtuales de Azure o seleccione **Extensiones**.



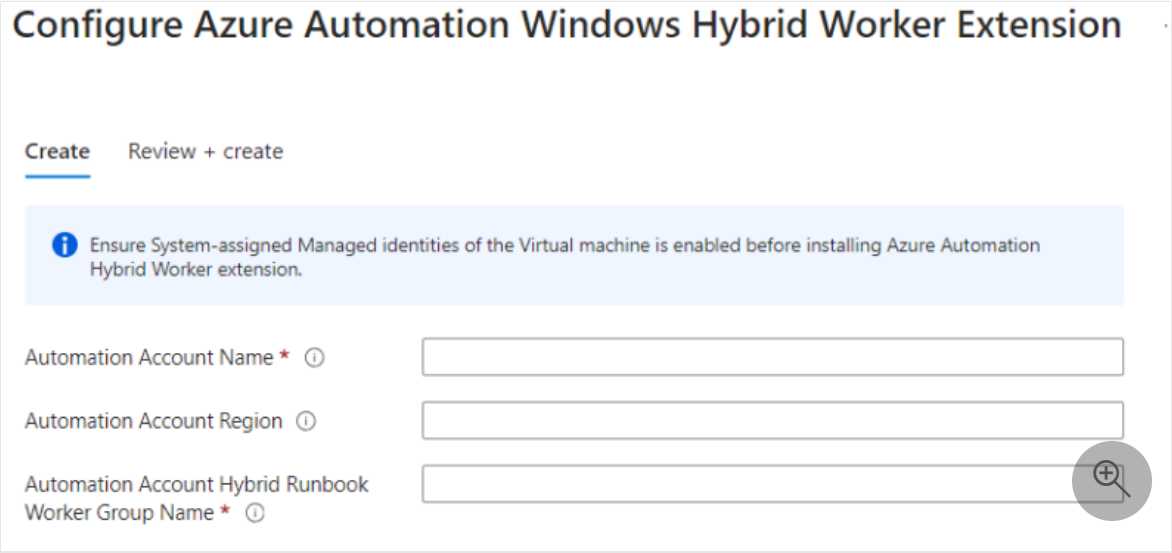
4. Seleccione **+Agregar**, y en **Instalar una extensión**, busque y seleccione **Azure Automation Windows Hybrid Worker** de la lista, y a continuación seleccione **Siguiente**.



5. Complete la instalación proporcionando el nombre de la cuenta de Automation, la región de la cuenta de Automation y el nombre del grupo de Hybrid Runbook Worker de la cuenta de Automation, donde desea agregar la máquina seleccionada en el paso 2.

> ⓘ **Nota**

Si no tiene una cuenta de Automation existente o un grupo de Hybrid Runbook Worker, proporcione los nuevos nombres y los recursos se crearán automáticamente en la suscripción.



6. Después de confirmar la información necesaria, seleccione **Revisar y crear**.

   Se muestra un resumen de la implementación y puede revisar el estado de esta. La extensión de Hybrid Worker se instala en la máquina y la instancia de Hybrid Worker se registra en el grupo de Hybrid Worker. La máquina ahora se puede usar para ejecutar trabajos de Azure Automation.

Pasos siguientes

- Para obtener información sobre Hybrid Runbook Worker, las ventajas y las limitaciones, consulte Introducción a Hybrid Runbook Worker de Automation
- Para aprender a desplegar un Trabajador de Runbook Híbrido de Usuario en máquinas Windows y Linux, consulte Ejecutar runbooks de Automation en un Hybrid Runbook Worker
- Para obtener información sobre las ventajas del runbook híbrido de usuario basado en extensiones y cómo migrar un Hybrid Worker basado en agente existente a Hybrid Worker basado en extensiones, consulte Migración de los trabajos híbridos basados en agente existentes a los trabajos híbridos basados en extensiones

Inicio rápido: Habilitación de identidades administradas para la cuenta de Automation mediante Azure Portal

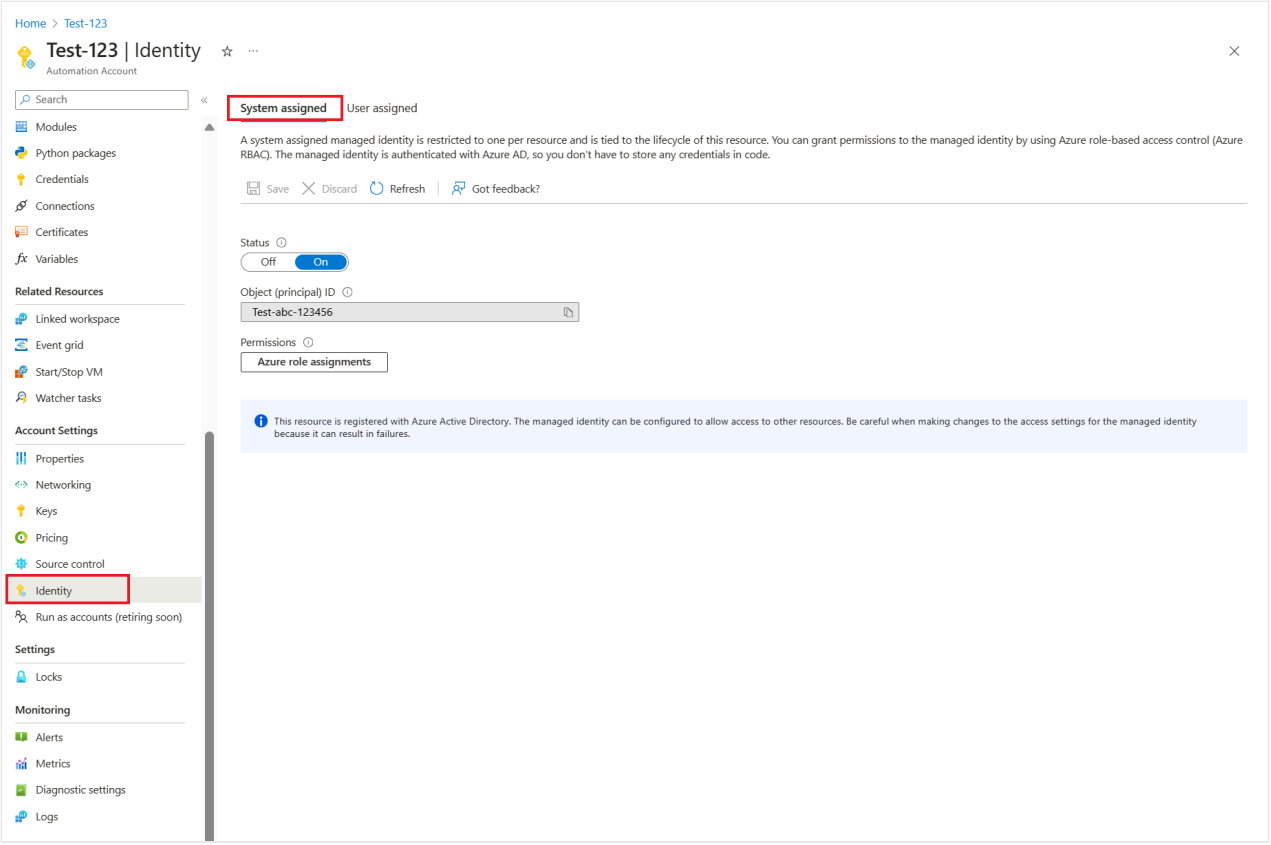
En este inicio rápido se muestra cómo habilitar identidades administradas para una cuenta de Azure Automation. Para más información sobre cómo funcionan las identidades administradas con Azure Automation, consulte Identidades administradas.

Prerrequisitos

- Una cuenta de Azure con una suscripción activa. Cree una cuenta gratuita.
- Una cuenta de Azure Automation Para obtener instrucciones, consulte Creación de una cuenta de Automation.
- Una identidad administrada asignada por el usuario. Para obtener instrucciones, consulte Creación de una identidad administrada asignada por el usuario. La identidad administrada asignada por el usuario y los recursos de Azure de destino que administra el runbook mediante esa identidad deben estar en la misma suscripción de Azure.

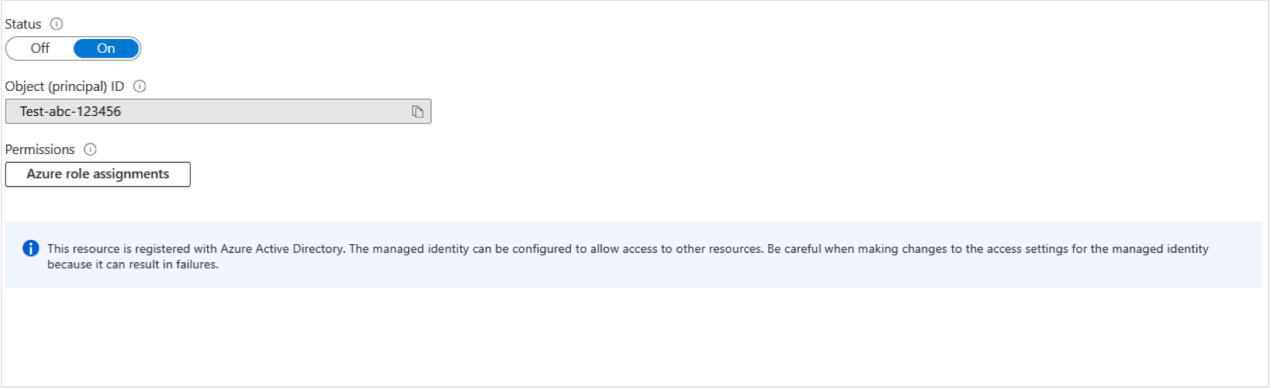
Habilitación de identidad administrada asignada por el sistema

1. Inicie sesión en Azure Portal y vaya a su cuenta de Automation.
2. En **Configuración de la cuenta**, seleccione **Identidad (versión preliminar)** .



3. Establezca la opción **Estado** asignado por el sistema en **Activado** y presione **Guardar**. Cuando se le solicite confirmación, seleccione **Sí**.

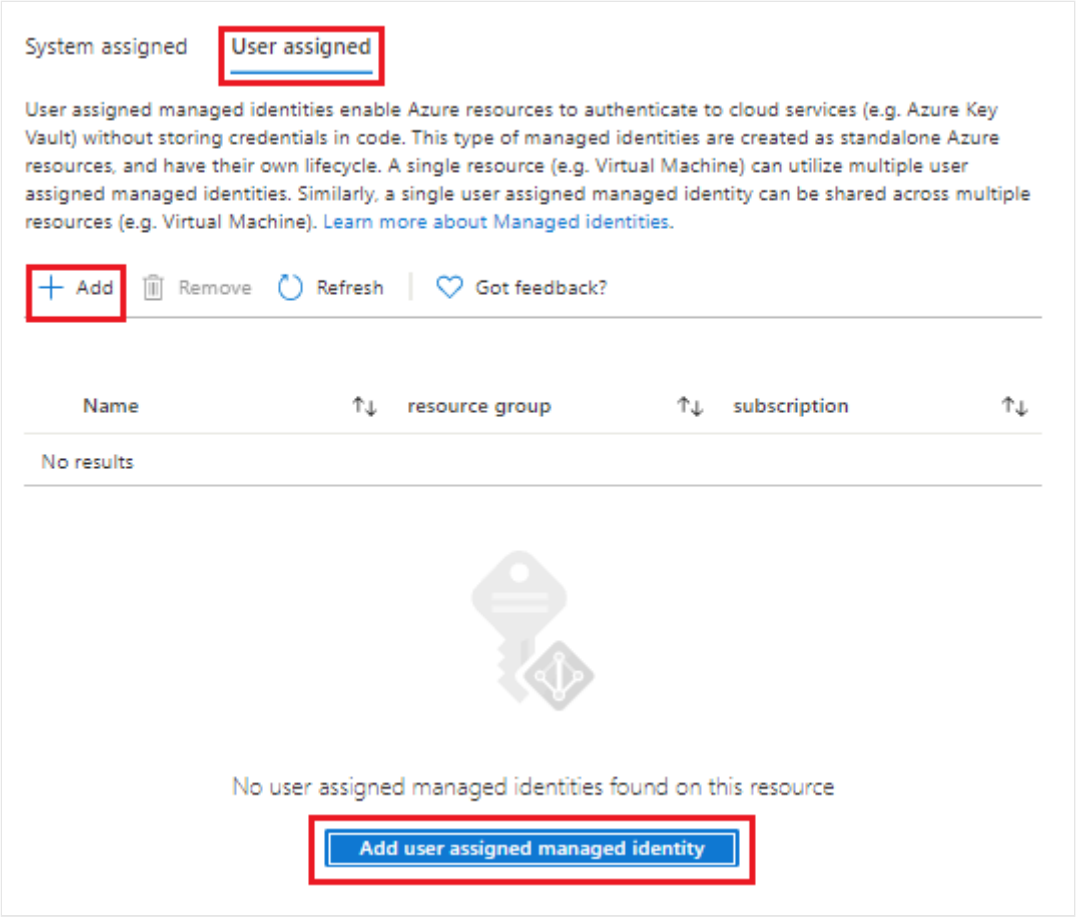
   La cuenta de Automation ahora podrá usar la identidad asignada por el sistema, registrada con Microsoft Entra ID y representada mediante un identificador de objeto.



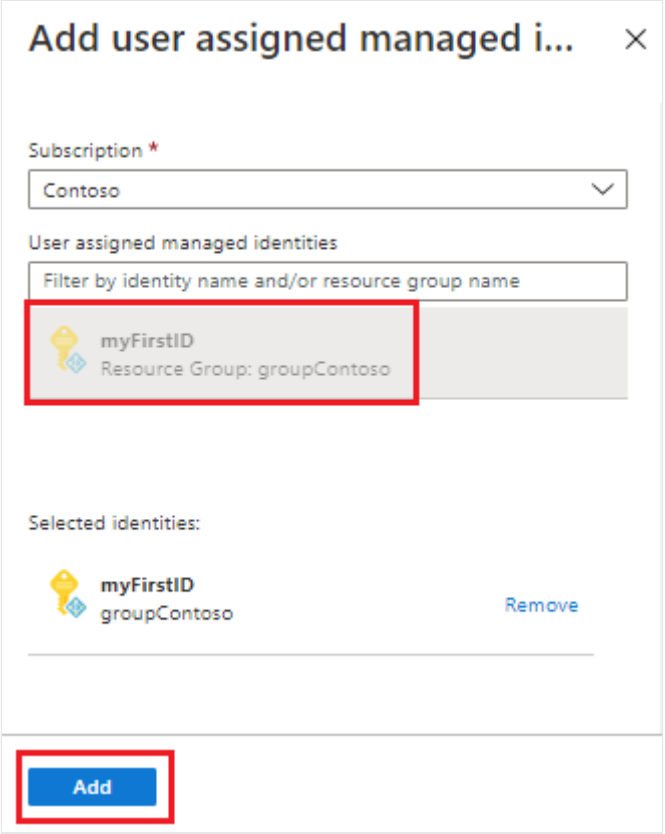
Adición de una identidad administrada asignada por el usuario

Esta sección continúa desde donde finalizó la última sección.

1. Seleccione la pestaña **Usuario asignado** y luego elija **+ Agregar** o **Agregar identidad administrada asignada por el usuario** para abrir la página **Agregar identidad administrada asignada por el usuario**.

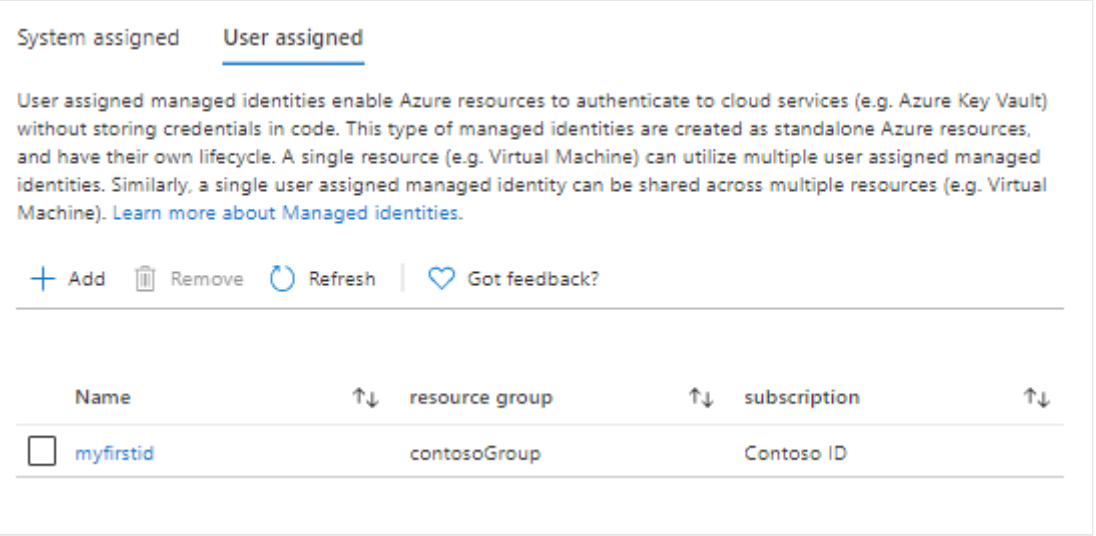


2. En la lista desplegable **Suscripción** , seleccione la suscripción de la identidad administrada asignada por el usuario.



3. En **Identidades administradas asignadas por** el usuario, seleccione la identidad administrada asignada por el usuario existente y, a continuación, seleccione **Agregar**. A

continuación, volverá a la pestaña **Usuario asignado**.



Limpieza de recursos

Si ya no necesita la identidad administrada asignada por el usuario asociada a la cuenta de Automation, siga estos pasos:

1. En la pestaña **Usuario asignado** , seleccione la identidad administrada asignada por el usuario.

2. En el menú superior, seleccione **Quitar** y, a continuación, seleccione **Sí** cuando se le pida confirmación.

Si ya no necesita la identidad administrada asignada por el sistema habilitada para la cuenta de Automation, siga estos pasos:

1. En la pestaña **Asignado por el sistema** , en **Estado**, seleccione **Desactivado**.

2. En el menú superior, seleccione **Guardar** y, a continuación, seleccione **Sí** cuando se le pida confirmación.

Pasos siguientes

En este inicio rápido, ha habilitado identidades administradas para una cuenta de Azure Automation. Para usar la cuenta de Automation con identidades administradas para ejecutar un runbook, consulte.

Tutorial: Creación de un runbook de PowerShell de Automation mediante una identidad administrada

Last updated on 17/11/2025

Configuración de una máquina virtual con Desired State Configuration

> **Nota:**
>
> State Configuration de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration para esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los **vínculos de navegación Agregar**, **Redactar**y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> DSC de Azure Automation para Linux se ha retirado. Para obtener más información, consulte el anuncio.

Al habilitar State Configuration de Azure Automation, puede administrar y supervisar las configuraciones de los servidores Windows mediante Desired State Configuration (DSC). Las configuraciones que se desvían de una configuración deseada se pueden identificar o corregir automáticamente. En este inicio rápido se explica cómo habilitar una máquina virtual de Azure e implementar una pila LAMP mediante State Configuration de Azure Automation.

Requisitos previos

Para completar este inicio rápido necesita instalar:

- Suscripción a Azure. Si no tiene una suscripción a Azure, cree una cuenta gratuita.
- Una máquina virtual de Azure Resource Manager.

Inicio de sesión en Azure

Inicie sesión en Azure Portal.

Habilitación de una máquina virtual

Hay muchas formas diferentes de habilitar una máquina para el servicio State Configuration de Automation. En este inicio rápido se explica cómo habilitar la característica para una máquina virtual mediante una cuenta de Automation. Puede leer Habilitación de State Configuration de Azure Automation en una máquina para aprender más sobre los distintos métodos para realizar esta operación.

1. En Azure Portal, vaya a las **cuentas de Automation**.
2. En la lista de cuentas de Automation, seleccione una opción.
3. En el panel izquierdo de la cuenta de Automation, seleccione **State Configuration (DSC)** (Configuración de estado [DSC]).
4. Haga clic en **Agregar** para abrir la página de **selección de máquinas virtuales**.
5. Busque la máquina virtual en la que va a habilitar DSC. Puede usar el campo de búsqueda y las opciones de filtro para buscar una máquina virtual específica.
6. Haga clic en la máquina virtual y, después, haga clic en **Conectar**.
7. Seleccione la configuración de DSC adecuada para la máquina virtual. Si ya tiene una configuración preparada, la puede especificar como `Node Configuration Name`. Puede establecer el modo de configuración para controlar el comportamiento de la configuración de la máquina.
8. Haga clic en **OK**. Aunque la extensión DSC se implementa en la máquina virtual, el estado notificado es `Connecting`.

Importación de módulos

Los módulos contienen recursos de DSC y muchos de ellos se pueden encontrar en la Galería de PowerShell. Todos los recursos que se utilizan en las configuraciones se deben importar en la cuenta de Automation antes de la compilación. En este tutorial, se necesita el módulo **nx**.

1. En el panel izquierdo de la cuenta de Automation, seleccione **Galería de módulos** en **Recursos compartidos**.
2. Para buscar el módulo que desea importar, escriba parte de su nombre: `nx`.
3. Haga clic en el módulo que se va a importar.
4. Haga clic en **Import**.

Importar la configuración

Esta guía de inicio rápido usa una configuración de DSC que configura el servidor HTTP Apache, MySQL y PHP en la máquina. Consulte Configuraciones de DSC.

En un editor de texto, escriba lo siguiente y guárdelo localmente como **AMPServer.ps1**.

PowerShell

```
configuration 'LAMPServer' {
   Import-DSCResource -module "nx"

   Node localhost {

        $requiredPackages = @("httpd","mod_ssl","php","php-
mysql","mariadb","mariadb-server")
        $enabledServices = @("httpd","mariadb")

        #Ensure packages are installed
        ForEach ($package in $requiredPackages){
            nxPackage $Package{
                Ensure = "Present"
                Name = $Package
                PackageManager = "yum"
            }
        }

        #Ensure daemons are enabled
        ForEach ($service in $enabledServices){
            nxService $service{
                Enabled = $true
                Name = $service
                Controller = "SystemD"
                State = "running"
            }
        }
   }
}
```

Para importar la configuración:

1. En el panel izquierdo de la cuenta de Automation, seleccione **State Configuration (DSC)** (Configuración de estado [DSC]) y, luego, haga clic en la pestaña **Configuraciones**.
2. Haga clic en **+ Agregar**.
3. Seleccione el archivo de configuración que guardó en el paso anterior.
4. Haga clic en **OK**.

Compilar una configuración

Debe compilar una configuración DSC en la configuración de un nodo (documento de Managed Object Format) antes de que se pueda asignar a un nodo. La compilación valida la

configuración y permite la entrada de valores de parámetro. Para más información sobre la compilación de una configuración, consulte: Compilación de configuraciones en State Configuration.

1. En el panel izquierdo de la cuenta de Automation, seleccione **State Configuration (DSC)** (Configuración de estado [DSC]) y, luego, haga clic en la pestaña **Configuraciones**.
2. Seleccione la configuración `LAMPServer`.
3. En las opciones del menú, seleccione **Compilar** y, después, haga clic en **Sí**.
4. En la vista Configuración aparecerá un nuevo trabajo de compilación en cola. Cuando el trabajo se complete correctamente, podrá avanzar al paso siguiente. Si se produce algún error, puede hacer clic en el trabajo de compilación para obtener más información.

Asignación de una configuración de nodo

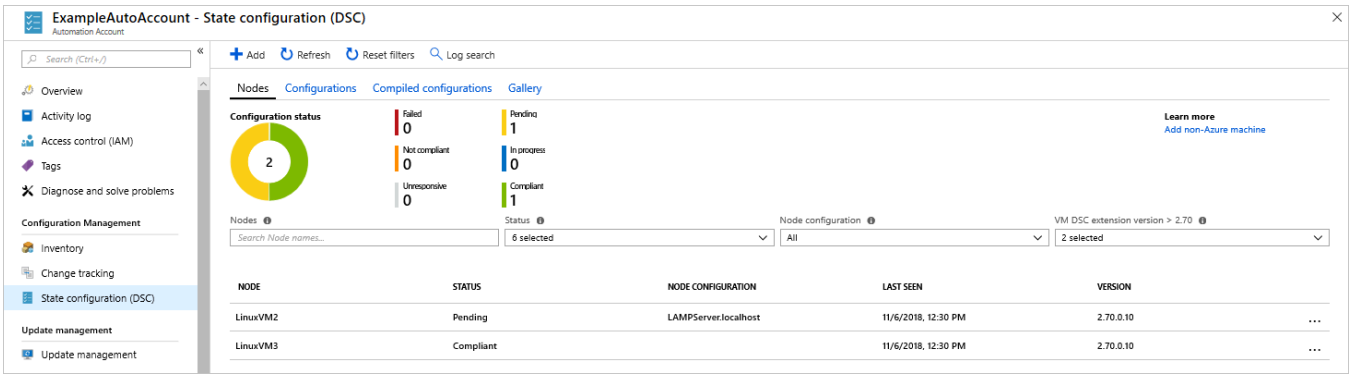
Puede asignar una configuración de nodo compilada a un nodo de DSC. La asignación aplica la configuración a la máquina y supervisa, o corrige automáticamente, cualquier desvío de esa configuración.

1. En el panel izquierdo de la cuenta de Automation, seleccione **State Configuration (DSC)** y, luego, haga clic en la pestaña **Nodos**.
2. Seleccione el nodo al que va asignar una configuración.
3. Haga clic en **Asignar configuración de nodo**.
4. Seleccione la configuración del nodo `LAMPServer.localhost` y haga clic en **Aceptar**. State Configuration ahora asigna la configuración compilada al nodo y el estado del nodo cambia a `Pending`. En la siguiente comprobación periódica, el nodo recupera la configuración, la aplica y genera un informe del estado.

El nodo puede tardar hasta 30 minutos en recuperar la configuración en función de su configuración.

Visualización del estado del nodo

Puede ver el estado de todos los nodos administrados por State Configuration de la cuenta de Automation. Para mostrar la información hay que elegir **State Configuration (DSC)** y hacer clic en la pestaña **Nodos**. Puede filtrar la visualización por el estado, la configuración del nodo o la búsqueda de nombre.



Pasos siguientes

En este inicio rápido, habilitó una máquina virtual de Azure para State Configuration, ha creado una configuración para una pila LAMP e implementó la configuración en la máquina virtual. Para aprender a usar Azure Automation State Configuration para habilitar la implementación continua, consulte este artículo:

Configuración de la implementación continua con Chocolatey

Last updated on 17/11/2025

Tutorial: Creación de un runbook de PowerShell de Automation mediante una identidad administrada

Este tutorial le guía por la creación de un runbook de PowerShell en Azure Automation que usa una identidad administrada para interactuar con los recursos. Los runbooks de PowerShell se basan en Windows PowerShell. Una identidad administrada de Microsoft Entra ID permite que el runbook acceda fácilmente a otros recursos protegidos por Microsoft Entra.

En este tutorial, aprenderá a:

- ✓ Asignación de permisos a las identidades administradas
- ✓ Creación de un runbook de PowerShell

Si no tiene una suscripción de Azure, cree una cuenta free antes de comenzar.

Prerrequisitos

Antes de asignar permisos a identidades administradas, asegúrese de cumplir estos requisitos previos:

- Una cuenta Azure Automation con al menos una identidad administrada asignada por el usuario. Para obtener más información, consulte Uso de una identidad administrada asignada por el usuario para una cuenta de Azure Automation.
- Los módulos de az `Az.Accounts`, `Az.Automation`, `Az.ManagedServiceIdentity` y `Az.Compute` importados en la cuenta de Automation. Para más información, consulte Importar módulos Az.
- El módulo Azure Az PowerShell instalado en el equipo. Para instalar o actualizar, consulte Cómo instalar el módulo Az de PowerShell de Azure. `Az.ManagedServiceIdentity` es un módulo en versión preliminar y no se instala como parte del módulo Az. Para instalarlo, ejecute `Install-Module -Name Az.ManagedServiceIdentity`.
- Una máquina virtual Azure. Dado que va a detener e iniciar esta máquina, no debería ser una máquina virtual de producción.
- Familiaridad general con los runbooks de Automation.

Asignación de permisos a las identidades administradas

Asigne permisos a las identidades administradas para permitirles detener e iniciar una máquina virtual.

Para asignar permisos a identidades administradas, siga estos pasos:

1. Inicie sesión en Azure de forma interactiva con el cmdlet Connect-AzAccount y siga las instrucciones:

   PowerShell

   ```
   # Sign in to your Azure subscription
   $sub = Get-AzSubscription -ErrorAction SilentlyContinue
   if(-not ($sub)) {
       Connect-AzAccount
   }

   # If you have multiple subscriptions, set the one to use
   # Select-AzSubscription -SubscriptionId <SUBSCRIPTIONID>
   ```

2. Escriba un valor adecuado para las variables siguientes y, a continuación, ejecute el script.

   PowerShell

   ```
   $resourceGroup = "resourceGroupName"

   # These values are used in this tutorial
   $automationAccount = "xAutomationAccount"
   $userAssignedManagedIdentity = "xUAMI"
   ```

3. Use el cmdlet de PowerShell New-AzRoleAssignment para asignar un rol a la identidad administrada asignada por el sistema.

   PowerShell

   ```
   $role1 = "DevTest Labs User"

   $SAMI = (Get-AzAutomationAccount -ResourceGroupName $resourceGroup -Name
   $automationAccount).Identity.PrincipalId
   New-AzRoleAssignment `
       -ObjectId $SAMI `
       -ResourceGroupName $resourceGroup `
       -RoleDefinitionName $role1
   ```

4. Se necesita la misma asignación de roles para la identidad administrada asignada por el usuario.

PowerShell

```
$UAMI = (Get-AzUserAssignedIdentity -ResourceGroupName $resourceGroup -Name
$userAssignedManagedIdentity).PrincipalId
New-AzRoleAssignment `
    -ObjectId $UAMI `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName $role1
```

5. La identidad administrada asignada por el sistema necesita permisos adicionales para ejecutar los cmdlets `Get-AzUserAssignedIdentity` y `Get-AzAutomationAccount` tal como se usan en este tutorial.

PowerShell

```
$role2 = "Reader"
New-AzRoleAssignment `
    -ObjectId $SAMI `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName $role2
```

Creación de runbook de PowerShell

Cree un runbook que permita la ejecución por cualquiera de las identidades administradas. El runbook iniciará una máquina virtual detenida o detendrá una máquina virtual en ejecución.

Para crear un runbook de PowerShell, siga estos pasos:

1. Inicie sesión en el portal Azure y vaya a la cuenta de Automation.

2. En la página **Información general**, seleccione **Probar la experiencia del entorno de ejecución**, si aún no está en la nueva experiencia.

3. En **Automatización de procesos**, seleccione **Runbooks**.

4. Seleccione **Crear un runbook** y haga lo siguiente:
   a. Asigne al runbook el nombre `miTesting`.
   b. En la lista desplegable **Tipo de Runbook**, seleccione **PowerShell**.
   c. En la lista desplegable de **Entorno de tiempo de ejecución**, **seleccione un entorno de tiempo de ejecución existente** o **cree uno nuevo** con PowerShell como entorno de ejecución y versión 7.4.
   d. Escriba una **descripción** correspondiente.

5. Seleccione **Crear** para crear el runbook.

6. En el editor de runbooks, pegue el código siguiente:

PowerShell

```
Param(
    [string]$ResourceGroup,
    [string]$VMName,
    [string]$Method,
    [string]$UAMI
)

$automationAccount = "xAutomationAccount"

# Ensures you do not inherit an AzContext in your runbook
$null = Disable-AzContextAutosave -Scope Process

# Connect using a Managed Service Identity
try {
    $AzureConnection = (Connect-AzAccount -Identity).context
}
catch {
    Write-Output "There is no system-assigned user identity. Aborting."
    exit
}

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureConnection.Subscription -
DefaultProfile $AzureConnection

if ($Method -eq "SA") {
    Write-Output "Using system-assigned managed identity"
}
elseif ($Method -eq "UA") {
    Write-Output "Using user-assigned managed identity"

    # Connects using the Managed Service Identity of the named user-assigned
managed identity
    $identity = Get-AzUserAssignedIdentity -ResourceGroupName $ResourceGroup -
Name $UAMI -DefaultProfile $AzureContext

    # validates assignment only, not perms
    $AzAutomationAccount = Get-AzAutomationAccount -ResourceGroupName
$ResourceGroup -Name $automationAccount -DefaultProfile $AzureContext
    if
($AzAutomationAccount.Identity.UserAssignedIdentities.Values.PrincipalId.Contains
($identity.PrincipalId)) {
        $AzureConnection = (Connect-AzAccount -Identity -AccountId
$identity.ClientId).context

        # set and store context
        $AzureContext = Set-AzContext -SubscriptionName
$AzureConnection.Subscription -DefaultProfile $AzureConnection
    }
    else {
```

```
        Write-Output "Invalid or unassigned user-assigned managed identity"
        exit
    }
}
else {
    Write-Output "Invalid method. Choose UA or SA."
    exit
}

# Get current state of VM
$status = (Get-AzVM -ResourceGroupName $ResourceGroup -Name $VMName -Status -
DefaultProfile $AzureContext).Statuses[1].Code

Write-Output "`r`n Beginning VM status: $status `r`n"

# Start or stop VM based on current state
if ($status -eq "Powerstate/deallocated") {
    Start-AzVM -Name $VMName -ResourceGroupName $ResourceGroup -DefaultProfile
$AzureContext
}
elseif ($status -eq "Powerstate/running") {
    Stop-AzVM -Name $VMName -ResourceGroupName $ResourceGroup -DefaultProfile
$AzureContext -Force
}

# Get new state of VM
$status = (Get-AzVM -ResourceGroupName $ResourceGroup -Name $VMName -Status -
DefaultProfile $AzureContext).Statuses[1].Code

Write-Output "`r`n Ending VM status: $status `r`n `r`n"

Write-Output "Account ID of current context: " $AzureContext.Account.Id
```

7. En el editor, en la línea 8, modifique el valor de la variable `$automationAccount` según sea necesario.

8. Seleccione **Guardar** y, luego, **Panel de prueba**.

9. Rellene los parámetros `RESOURCEGROUP` y `VMNAME` con los valores adecuados. Escriba `SA` en el parámetro `METHOD` y `xUAMI` en el parámetro `UAMI`. El runbook intentará cambiar el estado de energía de la máquina virtual mediante la identidad administrada asignada por el sistema.

10. Seleccione **Inicio**. Una vez completado el runbook, la salida debería ser similar a la siguiente:

Resultados

```
 Beginning VM status: PowerState/deallocated

OperationId : 5b707401-f415-4268-9b43-be1f73ddc54b
```

```
Status      : Succeeded
StartTime   : 8/3/2021 10:52:09 PM
EndTime     : 8/3/2021 10:52:50 PM
Error       :
Name        :

 Ending VM status: PowerState/running

Account ID of current context:
MSI@50342
```

11. Cambie el valor del parámetro `METHOD` a `UA`.

12. Seleccione **Inicio**. El runbook intentará cambiar el estado de energía de la máquina virtual mediante la identidad administrada asignada por el usuario con nombre. Una vez completado el runbook, la salida debería ser similar a la siguiente:

Resultados

```
Using user-assigned managed identity

 Beginning VM status: PowerState/running

OperationId : 679fcadf-d0b9-406a-9282-66bc211a9fbf
Status      : Succeeded
StartTime   : 8/3/2021 11:06:03 PM
EndTime     : 8/3/2021 11:06:49 PM
Error       :
Name        :

 Ending VM status: PowerState/deallocated

Account ID of current context:
9034f5d3-c46d-44d4-afd6-c78aeab837ea
```

Limpieza de recursos

Para quitar los recursos que ya no sean necesarios, ejecute el siguiente runbook:

PowerShell

```
#Remove runbook
Remove-AzAutomationRunbook `
    -ResourceGroupName $resourceGroup `
    -AutomationAccountName $automationAccount `
    -Name "miTesting" `
    -Force

# Remove role assignments
```

```
Remove-AzRoleAssignment `
    -ObjectId $UAMI `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName $role1

Remove-AzRoleAssignment `
    -ObjectId $SAMI `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName $role2

Remove-AzRoleAssignment `
    -ObjectId $SAMI `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName $role1
```

Pasos siguientes

En este tutorial, ha creado un runbook de PowerShell en Azure Automation que usa una identidad administrada, en lugar de la cuenta de ejecución, para interactuar con los recursos. Para ver los runbooks de flujo de trabajo de PowerShell, consulte:

Tutorial: Creación de un runbook de flujo de trabajo de PowerShell

- Para solucionar problemas relacionados con la identidad administrada durante la creación del runbook de PowerShell, consulte Solucionar problemas de identidad administrada en Azure Automation.

Last updated on 07/05/2026

Tutorial: Creación de un runbook de flujo de trabajo de PowerShell en Automation

Este tutorial le guiará para crear un runbook de flujo de trabajo de PowerShell en Azure Automation. Los runbooks de flujo de trabajo de PowerShell son runbooks de texto basados en el flujo de trabajo de Windows PowerShell. Puede crear y modificar directamente el código del runbook con el editor de texto en Azure Portal.

> **Nota:**
>
> Este artículo solo es aplicable a PowerShell 5.1. Las versiones posteriores de PowerShell 7+ no admiten flujos de trabajo y no se pueden actualizar los runbooks obsoletos. Se recomienda usar runbooks de texto de PowerShell 7.2 para características avanzadas, como la ejecución de trabajos paralelos. **Obtenga más información** acerca de las limitaciones de los runbooks de flujo de trabajo de PowerShell.

En este tutorial, aprenderá a:

- ✓ Creación de un runbook de flujo de trabajo de PowerShell
- ✓ Prueba y publicación del runbook
- ✓ Ejecución y seguimiento del estado del trabajo de runbook
- ✓ Agregar autenticación para administrar recursos de Azure
- ✓ Actualización de los parámetros de runbook para iniciar una máquina virtual de Azure

Si no tiene una suscripción a Azure, cree una cuenta gratuita antes de empezar.

Prerrequisitos

- Una cuenta de Azure Automation con al menos una identidad administrada asignada por el usuario. Para obtener más información, consulte Habilitación de identidades administradas.
- Módulos de az: `Az.Accounts` y `Az.Compute` se han importado en la cuenta de Automation. Para más información, consulte Importación de módulos Az.
- Dos o más máquinas virtuales de Azure. Puesto que detiene e inicia estas máquinas, no deben ser máquinas virtuales de producción.
- El módulo Azure Az de PowerShell instalado en la máquina. Para instalar o actualizar, vea Instalación del módulo Azure Az de PowerShell.

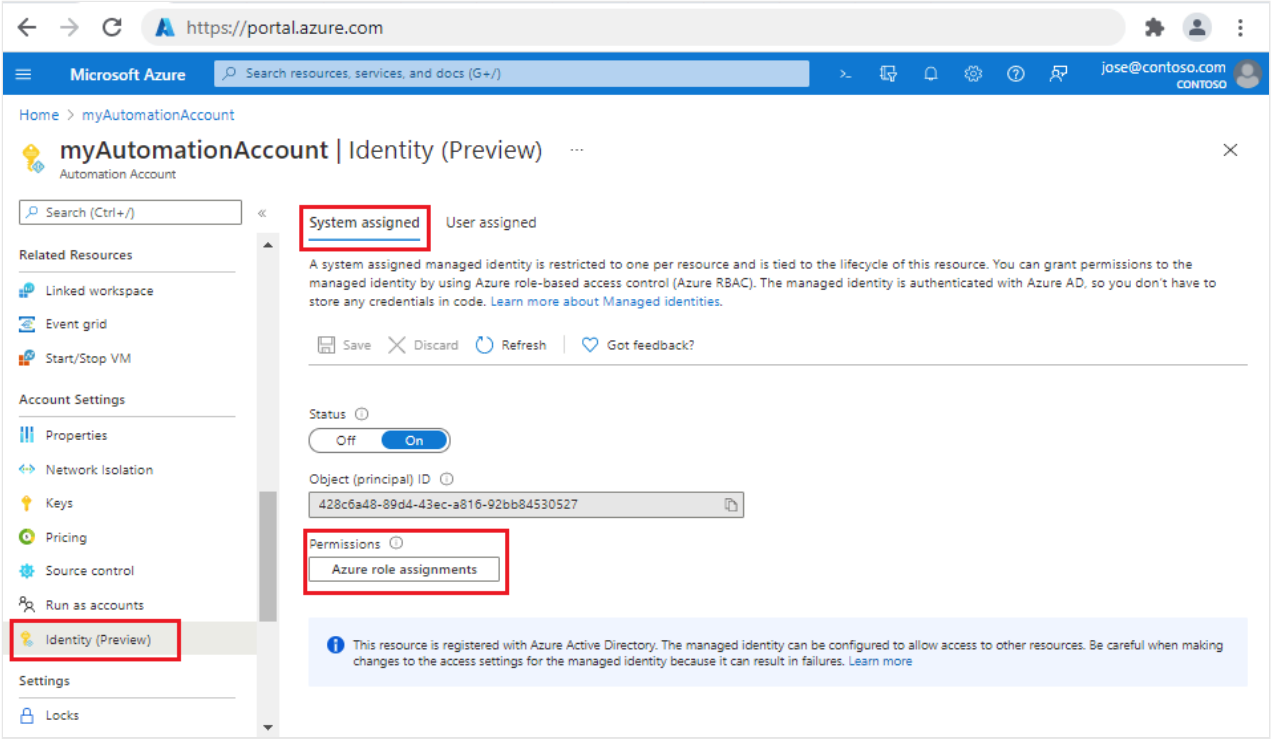
Asignación de permisos a las identidades administradas

Asigne permisos a la identidad administrada apropiada para permitirle detener una máquina virtual. El runbook puede usar la identidad administrada asignada por el sistema de la cuenta de Automation o una identidad administrada asignada por el usuario. Se proporcionan los pasos necesarios para asignar permisos a cada identidad. En los pasos siguientes se usa Azure Portal. Si prefiere usar PowerShell, consulte Asignación de roles de Azure mediante Azure PowerShell.

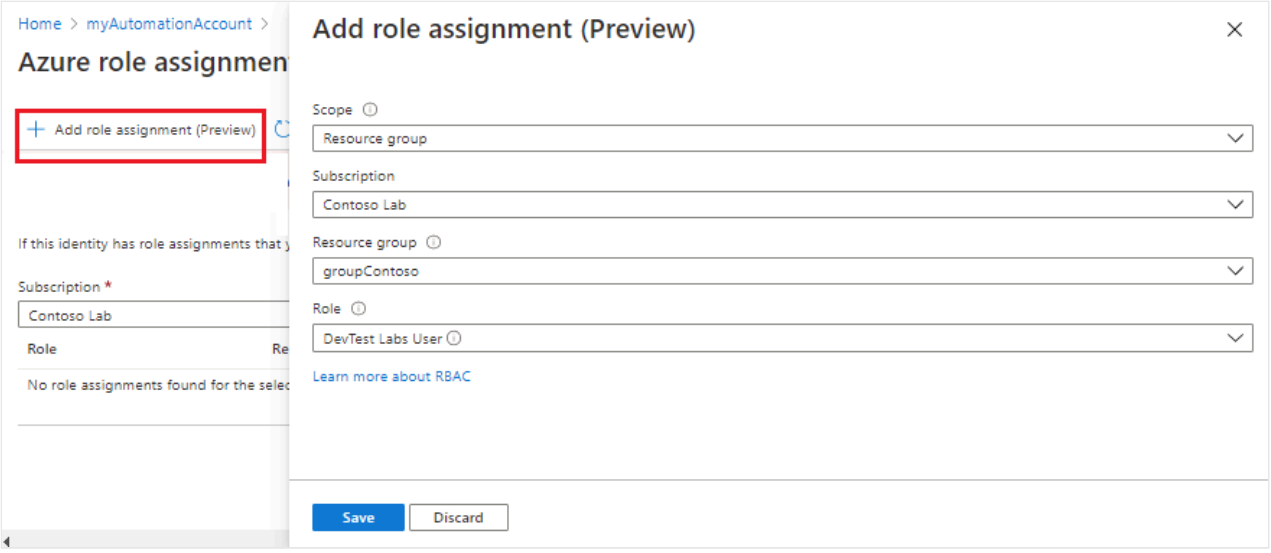
1. Inicie sesión en Azure Portal y vaya a su cuenta de Automation.

2. En **Configuración de la cuenta**, seleccione **Identidad (versión preliminar)** .

3. En la pestaña **Asignado por el sistema**, en **Permisos**, seleccione **Asignaciones de roles de Azure** para abrir la página **Asignaciones de roles de Azure**.



4. Seleccione + **Agregar asignación de roles (versión preliminar)** , para abrir la página **Agregar asignación de roles (versión preliminar)** .



5. Seleccione los valores apropiados.

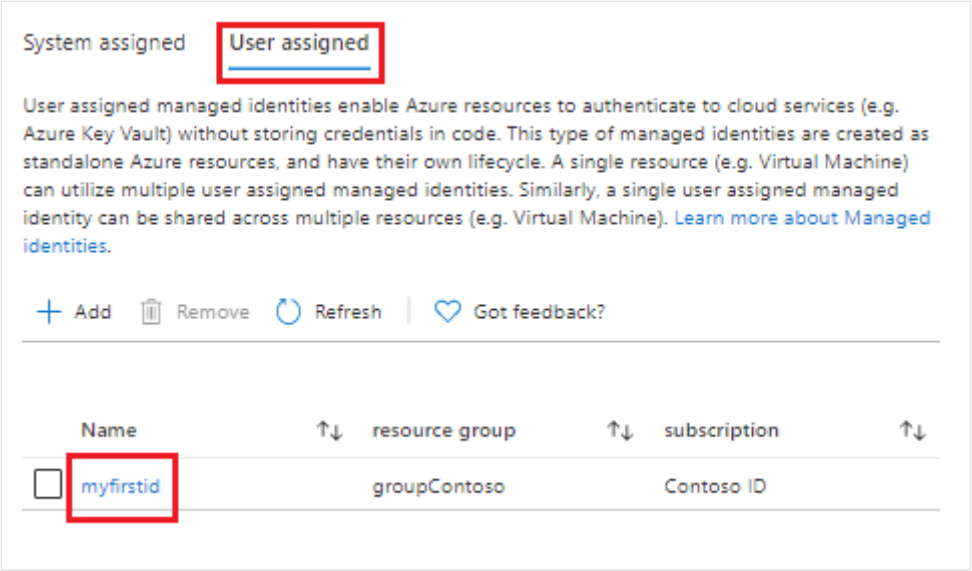
Expandir tabla

Propiedad	Description
Ámbito	Ámbito es el conjunto de recursos al que se aplica la asignación de roles. En la lista desplegable, seleccione **Grupo de recursos**.
Subscription	Este campo se debe rellenar automáticamente con la suscripción.
Grupo de recursos	En la lista desplegable, seleccione el grupo de recursos sobre el que conceder permisos de identidad.
Rol	En la lista desplegable, seleccione **Usuario de DevTest Labs**.

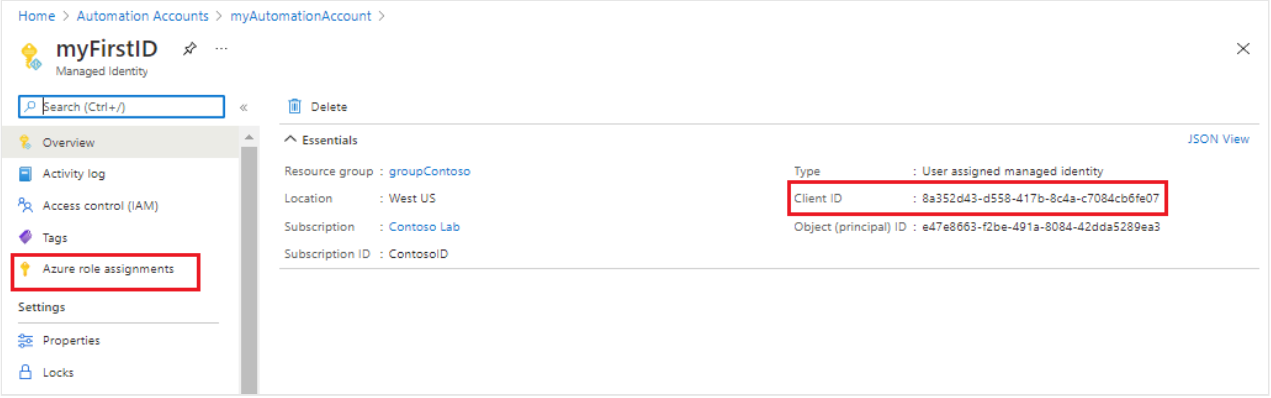
6. Seleccione **Guardar** y cierre la página **Asignaciones de roles de Azure** para volver a la pestaña **Asignado por el sistema**.

7. Seleccione la pestaña **Usuario asignado**.

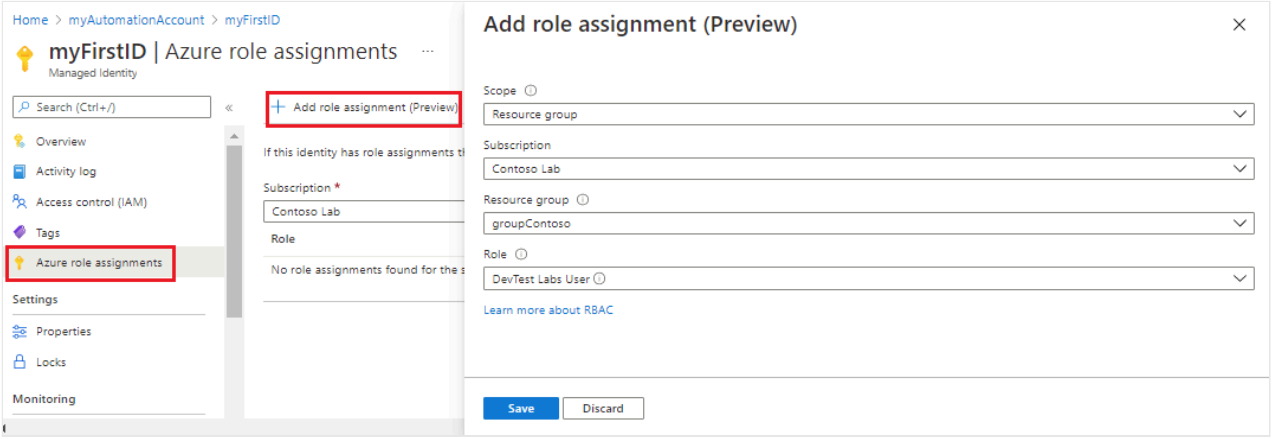
8. Seleccione la identidad administrada asignada por el usuario en la lista para abrir la página **Identidad administrada**.



9. Anote el valor de **Id. de cliente** para usarlo posteriormente.



10. En el menú de la izquierda, seleccione **Asignaciones de roles de Azure** y, después, **+ Agregar asignación de roles (versión preliminar)** para abrir la página **Agregar asignación de roles (versión preliminar)** .



11. Seleccione los valores apropiados.

Expandir tabla

Propiedad	Description
Ámbito	En la lista desplegable, seleccione **Grupo de recursos**.
Subscription	Este campo se debe rellenar automáticamente con la suscripción.
Grupo de recursos	En la lista desplegable, seleccione el grupo de recursos sobre el que conceder permisos de identidad.
Rol	En la lista desplegable, seleccione **Usuario de DevTest Labs**.

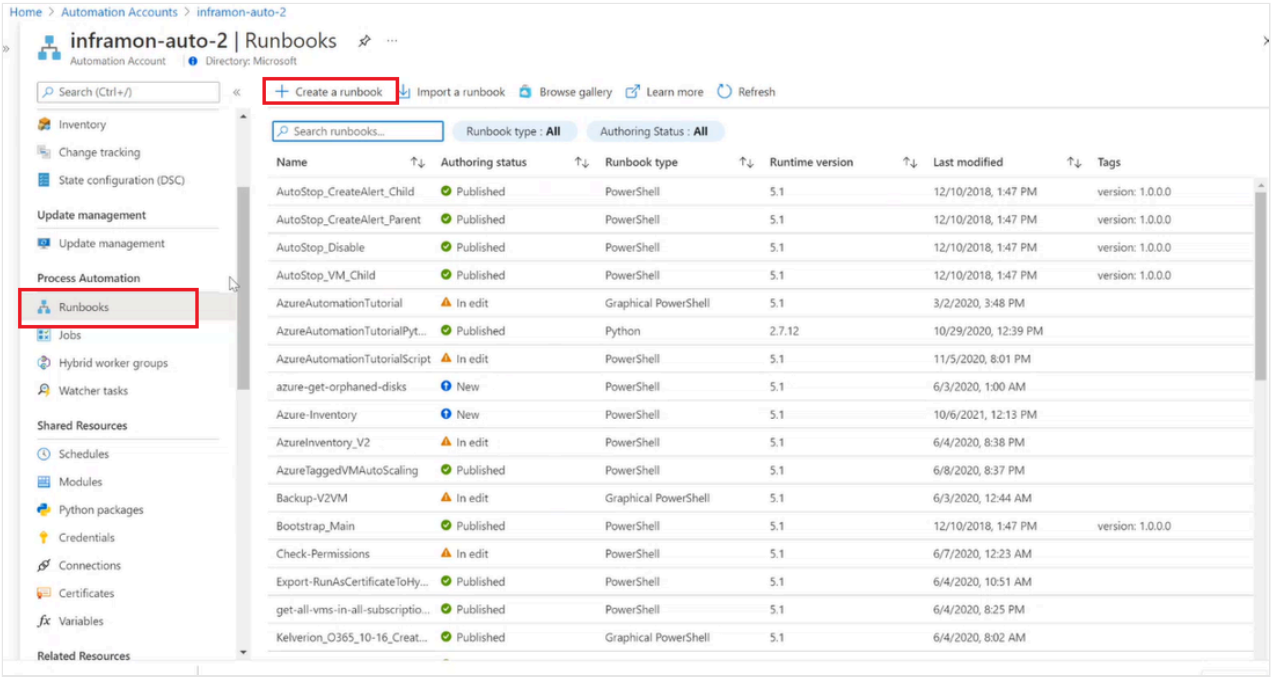
12. Seleccione **Guardar** y cierre la página **Asignaciones de roles de Azure** para volver a la pestaña **Asignado por el usuario**.

Creación de un runbook

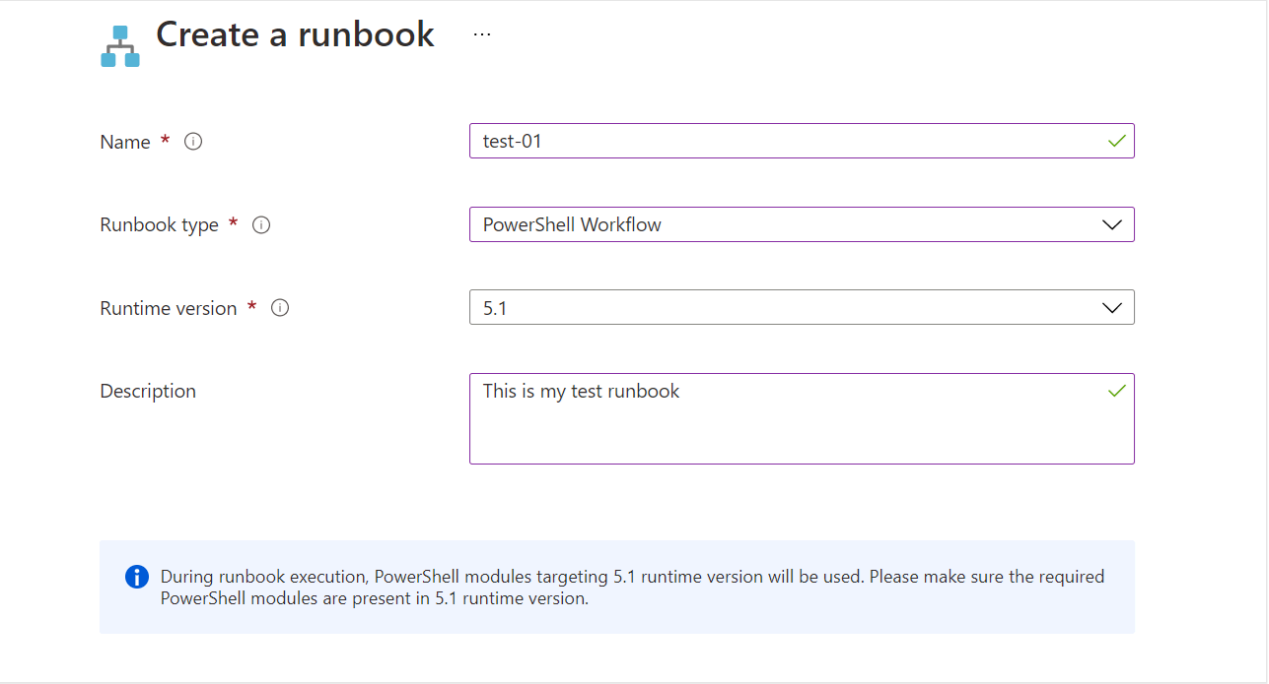
Empiece por crear un runbook de flujo de trabajo de PowerShell sencillo. Una ventaja de los flujos de trabajo de Windows PowerShell es la capacidad para realizar un conjunto de comandos en paralelo en lugar de hacerlo secuencialmente como con un script típico.

> ⓘ **Nota:**
>
> Con la creación de runbooks de la versión, tiene una nueva experiencia en Azure Portal. Al seleccionar la hoja **Runbooks**>**Crear un runbook**, se abre la nueva página **Creación de un runbook** con las opciones aplicables.

1. En la página de la cuenta de Automation abierta, en **Automatización de procesos**, seleccione **Runbooks**.



2. Seleccione + **Crear un runbook**.
   a. Asigne un nombre al runbook. Por ejemplo, prueba.
   b. En el menú desplegable **Tipo de Runbook**, seleccione **Flujo de trabajo de PowerShell**.
   c. En la lista desplegable **Versión del entorno de ejecución**, seleccione **5.1**.
   d. Escriba la **descripción** correspondiente.
   e. Seleccione **Crear**.



Agregar código al runbook

Puede escribir el código directamente en el runbook o seleccionar los cmdlets, runbooks y recursos desde el control Biblioteca y agregarlos al runbook con los parámetros relacionados. En este tutorial, escribirá el código directamente en el runbook.

El runbook está vacío actualmente, solo con la palabra clave `workflow` necesaria, el nombre del runbook y las llaves que encierran el flujo de trabajo completo.

PowerShell

```
workflow MyFirstRunbook-Workflow
{
}
```

1. Puede usar la palabra clave `Parallel` para crear un bloque de script con varios comandos que se ejecutarán simultáneamente. Escriba el código siguiente *entre* corchetes:

   PowerShell

```
parallel
{
   Write-Output "Parallel"
   Get-Date
   Start-Sleep -Seconds 3
   Get-Date
}

Write-Output " `r`n"
Write-Output "Non-Parallel"
Get-Date
Start-Sleep -Seconds 3
Get-Date
```

2. Guarde el Runbook, para lo que debe seleccionar **Guardar**.

Probar el runbook

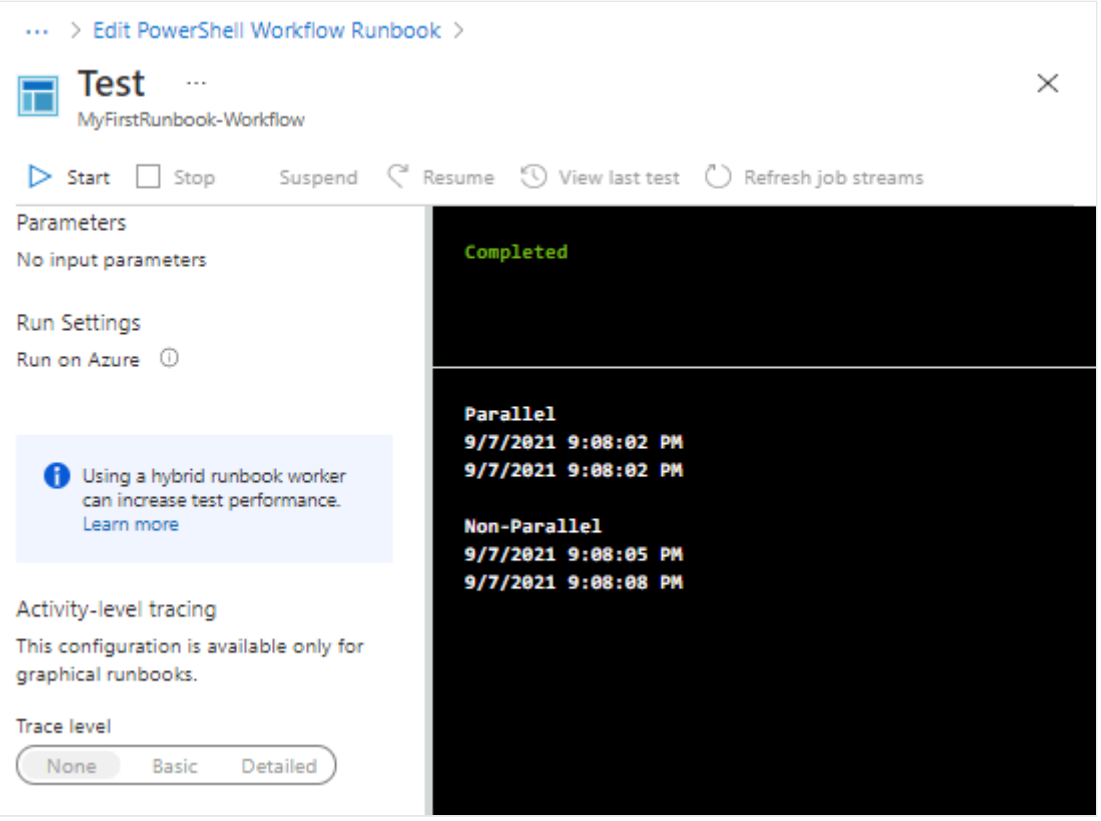
Antes de publicar el runbook para que esté disponible en producción, debe probarlo para asegurarse de que funciona correctamente. La prueba de un runbook ejecuta su versión de borrador y permite ver la salida de forma interactiva.

1. Seleccione **Panel de prueba** para abrir la página **Probar**.

2. Seleccione **Iniciar** para iniciar la prueba. Se crea un trabajo de runbook y su estado se muestra en el panel.

   El estado del trabajo se inicia como En cola, que indica que está esperando a que haya disponible un trabajo de runbook en la nube. El estado cambia a Iniciando cuando un trabajo lo solicita. Por último, el estado pasa a En ejecución cuando el runbook comienza a ejecutarse.

3. Cuando se completa el trabajo del runbook, la página **Prueba** muestra su salida. La salida debe ser similar a la siguiente:



Revise el resultado. Todo el contenido del bloque `Parallel`, incluido el comando `Start-Sleep`, se ejecuta al mismo tiempo. Los mismos comandos fuera del bloque `Parallel` se ejecutaron secuencialmente, como muestran las diferentes marcas de fecha y hora.

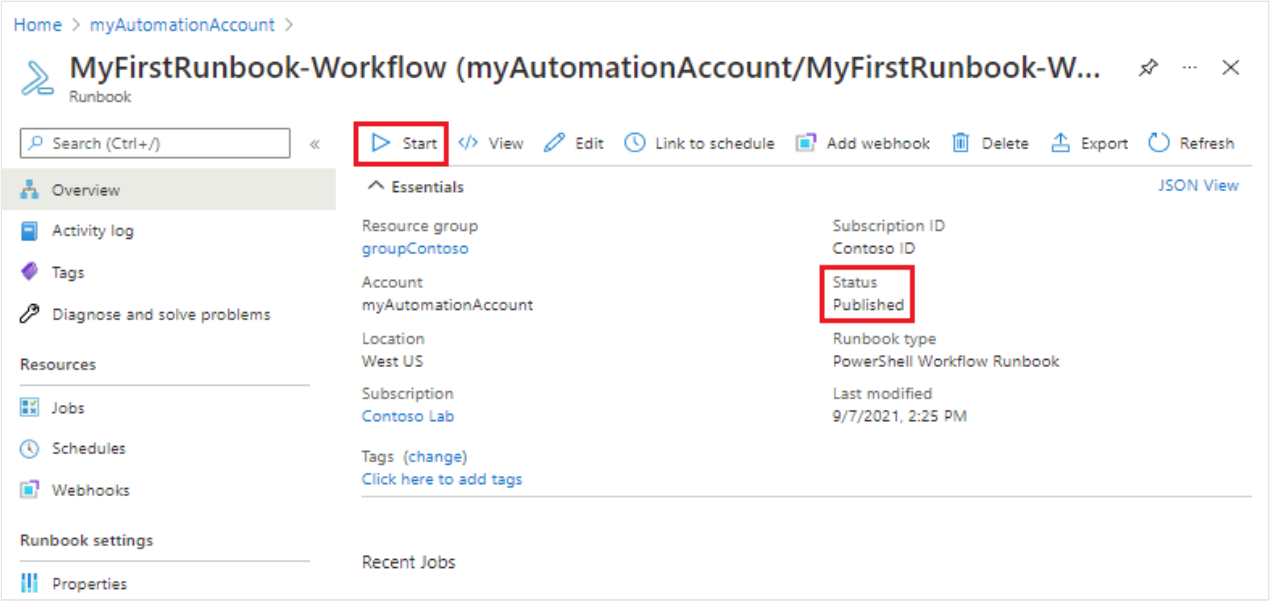
4. Cierre la página **Probar** para volver al lienzo.

Publicar e iniciar el runbook

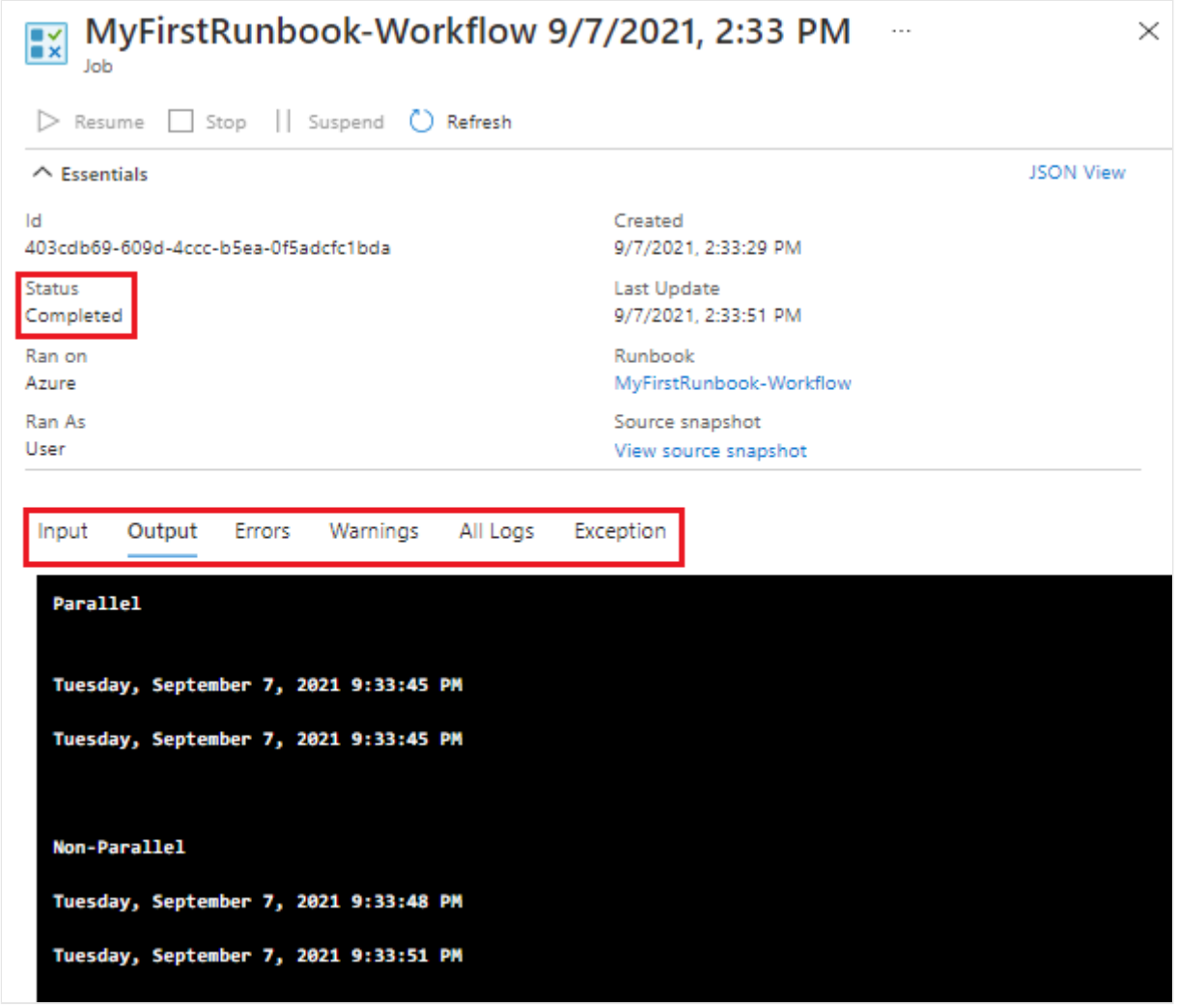
El runbook que ha creado aún está en modo Borrador. Tiene que publicarlo para poder ejecutarlo en producción. Al publicar un runbook, se sobrescribe la versión publicada existente con la versión de borrador. En este caso, no tiene una versión publicada aún porque acaba de crear el runbook.

1. Seleccione **Publicar** para publicar el runbook y **Sí** cuando se le solicite.

2. El campo **Estado** ahora muestra **Publicado**. Examine las opciones que se encuentran en la parte superior permiten iniciar el runbook ahora, programar una hora de inicio futura o crear un webhook para que el runbook pueda iniciarse mediante una llamada HTTP. Para iniciar el runbook, seleccione **Iniciar** y **Sí** cuando se le solicite.



3. Se abre una página **Trabajo** para el trabajo de runbook que se ha creado. En este caso, deje abierto el panel para poder ver el progreso del trabajo. El campo **Estado** coincide con los estados que vio al probar el runbook.



4. Cuando el estado del runbook aparezca como **Completado**, seleccione **Salida**. La salida debería ser similar a la de la prueba.

5. Cierre la página **Trabajo** para volver a la página **Información general** del runbook.

6. Seleccione **Trabajos** en **Recursos**. Esta página enumera todos los trabajos creados por el runbook. Solo debería ver un trabajo en la lista ya que solo ejecutó el trabajo una vez.

7. Seleccione el trabajo para abrir la misma página **Trabajo** que vio cuando se inició el runbook. Use esta página para ver los detalles de cualquier trabajo creado para el runbook. Cierre la página **Trabajo** para volver a la página **Información general** del runbook.

Agregar autenticación para administrar recursos de Azure

Ha probado y publicado su runbook, pero hasta ahora no hace nada útil. Quiere que administre recursos de Azure. Sin embargo, no puede hacerlo a menos que se autentique con las credenciales de la suscripción. El runbook usa la identidad administrada asignada por el sistema de la cuenta de Automation para autenticarse en Azure, con el fin de realizar la acción de administración en la máquina virtual. El runbook se puede modificar fácilmente para que use una identidad administrada asignada por el usuario.

1. Seleccione **Información general** y, después, en **Editar** para abrir el editor de texto.

2. Reemplace el código existente por el siguiente:

   PowerShell

   ```
   workflow MyFirstRunbook-Workflow
   {
    $resourceGroup = "resourceGroupName"

    # Ensures you do not inherit an AzContext in your runbook
    Disable-AzContextAutosave -Scope Process

    # Connect to Azure with system-assigned managed identity
    Connect-AzAccount -Identity

    # set and store context
    $AzureContext = Set-AzContext -SubscriptionId "<SubscriptionID>"
   }
   ```

   Edite la variable `$resourceGroup` con un valor válido que represente el grupo de recursos.

3. Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:

a. En la línea 9, quite `Connect-AzAccount -Identity`,
b. Reemplace el valor por `Connect-AzAccount -Identity -AccountId <ClientId>`.
c. Escriba el identificador de cliente que obtuvo anteriormente.

4. Seleccione **Guardar** y, luego, **Panel de prueba**.

5. Seleccione **Iniciar** para iniciar la prueba. Cuando termine, verá una salida similar a la siguiente, que muestra información básica de su cuenta. Esta acción confirma que la credencial es válida.

```
Completed

PSComputerName         : localhost
PSSourceJobInstanceId  : 61f9755d-a056-429e-80dd-843d6c2ca1b0
Mode                   : Process
ContextDirectory       :
ContextFile            :
CacheDirectory         :
CacheFile              :
Settings               : {}

PSComputerName         : localhost
PSSourceJobInstanceId  : 61f9755d-a056-429e-80dd-843d6c2ca1b0
Environments           : {AzureCloud, AzureUSGovernment, AzureChinaCloud, AzureGermanCloud}
Context                : Microsoft.Azure.Commands.Profile.Models.Core.PSAzureContext
```

6. Cierre la página **Probar** para volver al lienzo.

Incorporación de código para iniciar una máquina virtual

Ahora que el runbook está autenticado en la suscripción a Azure, puede administrar los recursos. Agregue un comando para iniciar una máquina virtual. Puede seleccionar cualquier máquina virtual de su suscripción de Azure. Por ahora, codificará ese nombre en el runbook.

1. Agregue el código siguiente como última línea inmediatamente antes de la llave de cierre. Reemplace `VMName` por el nombre real de la máquina virtual.

PowerShell

```
Start-AzVM -Name "VMName" -ResourceGroupName $resourceGroup -DefaultProfile
$AzureContext
```

2. Pruebe el runbook y confirme que la máquina virtual se ha iniciado. Luego, vuelva al lienzo.

Adición de parámetros de entrada al runbook

Actualmente, el runbook inicia la máquina virtual que ha codificado de forma rígida en el runbook. Será más útil si puede especificar la máquina virtual cuando se inicia el runbook. Agregue parámetros de entrada al runbook para proporcionar esa funcionalidad.

1. Reemplace la línea 3, `$resourceGroup = "resourceGroupName"`, por la siguiente:

   PowerShell
   ```
   Param(
       [string]$resourceGroup,
       [string]$VMName
   )
   ```

2. Reemplace el comando `Start-AzVM` anterior por el siguiente:

   PowerShell
   ```
   Start-AzVM -Name $VMName -ResourceGroupName $resourceGroup -DefaultProfile
   $AzureContext
   ```

3. Pruebe el runbook y confirme que la máquina virtual se ha iniciado. Luego, vuelva al lienzo.

Administración simultanea de varias máquinas virtuales

Puede utilizar la construcción `ForEach -Parallel` para procesar comandos para cada elemento de una colección simultáneamente. Revise el código para que el runbook ahora:

- Acepte una colección de nombres de máquina virtual.
- Acepte un parámetro para detener o iniciar las máquinas virtuales.
- Realice las acciones en paralelo en todas las máquinas virtuales.

1. Reemplace el código existente por el siguiente:

   PowerShell
   ```
   workflow MyFirstRunbook-Workflow
   {
   ```

```
    param
    (
        [string]$resourceGroup,
        [string[]]$VMs,
        [string]$action
    )

    # Ensures you do not inherit an AzContext in your runbook
    Disable-AzContextAutosave -Scope Process

    # Connect to Azure with system-assigned managed identity
    Connect-AzAccount -Identity

    # set and store context
    $AzureContext = Set-AzContext -SubscriptionId "<SubscriptionID>"

    # Start or stop VMs in parallel
    if ($action -eq "Start")
    {
        ForEach -Parallel ($vm in $VMs)
        {
            Start-AzVM -Name $vm -ResourceGroupName $resourceGroup -
DefaultProfile $AzureContext
        }
    }
    elseif ($action -eq "Stop")
    {
        ForEach -Parallel ($vm in $VMs)
        {
            Stop-AzVM -Name $vm -ResourceGroupName $resourceGroup -
DefaultProfile $AzureContext -Force
        }
    }
    else
    {
        Write-Output "`r`n Action not allowed. Please enter 'stop' or 'start'."
    }
}
```

2. Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:
   a. En la línea 9, quite `Connect-AzAccount -Identity`,
   b. Reemplace el valor por `Connect-AzAccount -Identity -AccountId <ClientId>`.
   c. Escriba el identificador de cliente que obtuvo anteriormente.

3. Seleccione **Guardar**, luego **Publicar** y, finalmente, **Sí** cuando se le pida confirmación.

4. En la página **Información general**, seleccione **Iniciar**.

5. Rellene los parámetros y seleccione **Aceptar**.

Expandir tabla

Parámetro	Description
RESOURCEGROUP	Escriba el nombre del grupo de recursos de las máquinas virtuales.
Máquinas virtuales	Escriba los nombres de las máquinas virtuales, para lo que debe usar la sintaxis siguiente: `["VM1","VM2","VM3"]`
Acción	Escriba `stop` o `start`.

6. Vaya a la lista de máquinas virtuales y actualice la página cada pocos segundos. Observe que la acción de cada máquina virtual se produce en paralelo. Sin la palabra clave `-Parallel`, las acciones se habrían realizado secuencialmente. Aunque las máquinas virtuales se iniciarán en paralelo, cada una de ellas puede alcanzar la fase de **ejecución** en momentos ligeramente diferentes en función de sus características.

Limpieza de recursos

Si no va a seguir usando este runbook, elimínelo mediante los siguientes pasos:

1. Vaya a su cuenta de Automation.
2. En **Automatización de procesos**, seleccione **Runbooks**.
3. Seleccione el Runbook.
4. En la página **Información general**, seleccione **Eliminar**.

Pasos siguientes

En este tutorial, ha creado un runbook de flujo de trabajo de PowerShell. Para examinar los runbooks de Python 3, consulte:

Tutorial: Creación de un runbook de Python 3

Last updated on 17/11/2025

Tutorial: Creación de un runbook de Python 3.8

Este tutorial le guía por la creación de un runbook de Python 3.8 en Azure Automation. Los runbooks de Python se compilan en Python 2.7 y 3.8 Puede editar directamente el código del runbook mediante el editor de texto en Azure Portal.

- ✓ Creación de un runbook de Python sencillo
- ✓ Prueba y publicación del runbook
- ✓ Ejecución y seguimiento del estado del trabajo de runbook
- ✓ Actualización del runbook para iniciar una máquina virtual de Azure con parámetros de runbook

Prerrequisitos

Para completar este tutorial, necesita lo siguiente:

- Una suscripción de Azure. Si aún no tiene una, puede activar las ventajas del suscriptor de MSDN o registrarse para obtener una cuenta gratuita.
- Cuenta de Automation para contener el runbook y autenticarse en recursos de Azure. Una identidad administrada se crea automáticamente al crear la cuenta de Automation.
- Una máquina virtual de Azure. Durante este tutorial, iniciará y detendrá esta máquina, por lo que no debe ser una máquina virtual de producción.

Crear un runbook

Para empezar, cree un runbook simple que genere el texto *Hola mundo*.

1. En Azure Portal, abra la cuenta de Automation.

   La página de la cuenta de Automatización permite ver rápidamente los recursos de esta cuenta. Ya debería tener algunos activos. La mayoría de esos recursos son los módulos que se incluyen automáticamente en una nueva cuenta de Automation.

   También debe tener habilitada una identidad administrada que se mencione en los requisitos previos. Para comprobarlo, vea el recurso **Identidad** en **Configuración de la cuenta**.

2. Seleccione **Runbooks** en **Automatización** de procesos para abrir la lista de runbooks.

3. Seleccione **Crear un Runbook** para crear un nuevo runbook.

4. Asigne al runbook el nombre **MyFirstRunbook-Python**.
5. Seleccione **Python** para el **tipo de Runbook**.
6. Seleccione **Python 3.8** para la **versión en tiempo de ejecución**.
7. Seleccione **Crear** para crear el runbook y abra el editor de texto.

Agregar código al runbook

Ahora agrega un comando sencillo para imprimir el texto `Hello World`.

Python
```
print("Hello World!")
```

Seleccione **Guardar** para guardar el runbook.

Probar el runbook

Antes de publicar el runbook para que esté disponible en producción, quieres probarlo para asegurarte de que funciona correctamente. Cuando pruebas un runbook, lo ejecutas en su versión borrador y ves su salida de forma interactiva.

1. Seleccione **Panel de pruebas** para abrir el panel **Prueba** .
2. Seleccione **Iniciar** para iniciar la prueba. Esta opción debe ser la única opción habilitada.
3. Se crea un trabajo de runbook y se muestra su estado. El estado del trabajo se inicia como **En cola**, lo que indica que está esperando a que esté disponible un trabajo de runbook en la nube. Su estado cambia a **Iniciando** cuando un trabajo de runbook solicita el trabajo. Cuando el runbook comienza a ejecutarse realmente, el estado es **En ejecución**.
4. Cuando se complete el trabajo del runbook, se mostrará su resultado. En este caso, deberías ver `Hello World`.
5. Cierre el panel **Prueba** para volver al lienzo.

Publicar e iniciar el runbook

El runbook que has creado todavía está en modo borrador. Tiene que publicarlo antes de que pueda ejecutarlo en producción. Al publicar un runbook, se sobrescribe la versión publicada

existente con la versión de borrador. En este caso, no tiene una versión publicada aún porque acaba de crear el runbook.

1. Seleccione **Publicar** para publicar el runbook y, después, **Sí** cuando se le solicite.

2. Si cierra el **MyFirstRunbook_python** panel, volverá a la página de **Runbooks**, donde debería ver un **Estado de Autoría** de **Publicado**.

3. Seleccione el nombre **MyFirstRunbook-Python** de la lista, volverá al panel **MyFirstRunbook-Python** .

   Las opciones en la parte superior le permiten iniciar el runbook, ver el runbook, editar el runbook, programarlo para empezar más adelante y realizar otras acciones.

4. Seleccione **Iniciar** y, a continuación, **seleccione Aceptar** cuando se abra el panel **Iniciar Runbook**.

5. Se abre un panel **Trabajo** para el trabajo de runbook que acaba de crear. Puede cerrar este panel, pero vamos a mantenerlo abierto para que pueda ver el progreso del trabajo.

6. El estado del trabajo se muestra en el campo **Estado** en **Essentials**. Los valores aquí coinciden con los valores de estado al probar el runbook.

7. Una vez que el estado del runbook aparezca **Completado**, seleccione la pestaña **Salida** . En la pestaña **Salida** , puede ver `Hello World`.

8. Cierre la pestaña **Salida** .

9. Seleccione la pestaña **Todos los registros** para ver los flujos del trabajo del runbook. Solo debería ver `Hello World` en el flujo de salida. No obstante, en esta pestaña se pueden mostrar otros flujos de un trabajo de runbook, como **Error** y **Modo detallado**, si el runbook escribe en ellas.

10. Cierre el panel **Trabajos** para volver al panel **MyFirstRunbook-Python** .

11. Seleccione el recurso **Trabajos** para abrir la página de recursos **Trabajos** para este runbook. Esta página enumera todos los trabajos creados por este runbook. Solo debería ver un trabajo en la lista ya que solo ejecutó el trabajo una vez.

12. Puede seleccionar este trabajo para abrir el mismo panel **Trabajo** que vio al iniciar el runbook. Esta panel permite volver atrás en el tiempo y ver los detalles de cualquier trabajo que se creó para un runbook determinado.

Adición de autenticación para administrar recursos de Azure

Ha probado y publicado su runbook, pero hasta ahora no hace nada útil. Quiere que administre los recursos de Azure. Para administrar recursos, el script tiene que autenticarse.

La manera recomendada de autenticarse es con **la identidad administrada**. Al crear una cuenta de Azure Automation, se crea automáticamente una identidad administrada.

Para usar estos ejemplos, agregue los siguientes paquetes en el recurso **Paquetes de Python** de la cuenta de Automation. Puede agregar los archivos WHL para estos paquetes con estos vínculos.

- azure-core
- azure-identity
- azure-mgmt-compute
- azure-mgmt-core
- msal
- typing-extensions

Cuando agregue estos paquetes, seleccione una versión de runtime que coincida con su runbook.

> **Nota:**
>
> El código siguiente se probó con la versión 3.8 del entorno de ejecución.

Identidad administrada

Para usar la identidad administrada, asegúrese de que está habilitada:

- Para comprobar si la identidad administrada está habilitada para la cuenta de Automation, vaya a su **cuenta de Automation**>**Configuración de la Cuenta**>**Identidad** y establezca el **estado** en **Activado**.
- La identidad administrada tiene un rol asignado para administrar el recurso. En este ejemplo de administración de un recurso de máquina virtual, agregue el rol "Colaborador de máquina virtual" en el grupo de recursos de que contiene la máquina virtual. Para más información, consulte Asignación de roles de Azure mediante Azure Portal.

Con el rol de administración de identidad configurado, puede empezar a agregar código.

1. Para abrir el editor de texto, seleccione **Editar** en el panel **MyFirstRunbook-Python** .

2. Agregue el código siguiente para autenticarse en Azure:

Python

```python
#!/usr/bin/env python3
from azure.identity import DefaultAzureCredential
from azure.mgmt.compute import ComputeManagementClient

SUBSCRIPTION_ID="YOUR_SUBSCRIPTION_ID"

azure_credential = DefaultAzureCredential()

import os
import requests
# printing environment variables
endpoint = os.getenv('IDENTITY_ENDPOINT')+"?resource=https://management.azure.com/"
identityHeader = os.getenv('IDENTITY_HEADER')
payload={}
headers = {
'X-IDENTITY-HEADER' : identityHeader,
'Metadata' : True
}
response = requests.get(endpoint, headers)
print(response.text)
```

Adición de código para crear el cliente de proceso de Python e iniciar la máquina virtual

Para trabajar con máquinas virtuales de Azure, cree una instancia del cliente de Azure Compute para Python.

Python

```python
# Initialize client with the credential and subscription.
compute_client = ComputeManagementClient(
    azure_credential,
    SUBSCRIPTION_ID
)

print('\nStart VM')
async_vm_start = compute_client.virtual_machines.begin_start(
    "MyResourceGroup", "TestVM")
async_vm_start.wait()
print('\nFinished start.')
```

Donde `MyResourceGroup` es el nombre del grupo de recursos que contiene la máquina virtual y `TestVM` es el nombre de la máquina virtual que desea iniciar.

Vuelva a probar y ejecutar el runbook para ver que inicia la máquina virtual.

Uso de parámetros de entrada

El runbook usa actualmente valores codificados de forma rígida para los nombres del grupo de recursos y la máquina virtual. Ahora vamos a agregar código que obtiene estos valores de los parámetros de entrada.

Usas la variable `sys.argv` para obtener los valores de parámetro. Agregue el código siguiente al runbook inmediatamente después de las otras `import` instrucciones:

Python

```python
import sys

resource_group_name = str(sys.argv[1])
vm_name = str(sys.argv[2])
```

Este código importa el `sys` módulo y crea dos variables para contener el grupo de recursos y los nombres de máquina virtual. Observe que el elemento de la lista de argumentos, `sys.argv[0]`, es el nombre del script y el usuario no lo introduce.

Ahora puede modificar las dos últimas líneas del runbook para usar los valores de parámetro de entrada en lugar de usar valores codificados de forma rígida:

Python

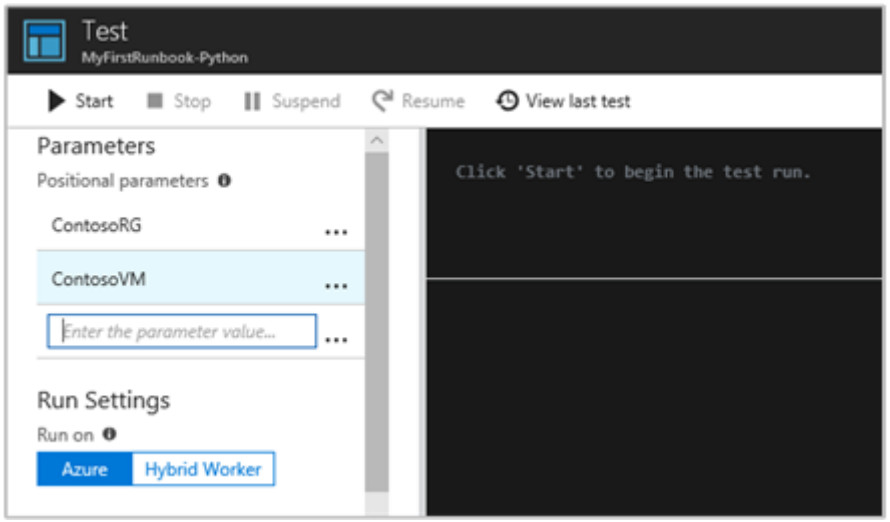
```python
async_vm_start = compute_client.virtual_machines.begin_start(
    resource_group_name, vm_name)
async_vm_start.wait()
```

Al iniciar un runbook de Python (ya sea en el panel **Prueba** o como runbook publicado), se pueden especificar los valores de los parámetros en la página **Iniciar Runbook**, en **Parámetros**.

Después de empezar a escribir un valor en el primer cuadro, aparece un segundo, etc., para que pueda escribir tantos valores de parámetro como sea necesario.

Los valores están disponibles en la matriz `sys.argv` del script, tal como en el código que acaba de agregar.

Escriba el nombre del grupo de recursos como el valor del primer parámetro y el nombre de la máquina virtual que se iniciará como el valor del segundo parámetro.



Seleccione **Aceptar** para iniciar el runbook. El runbook se ejecuta e inicia la máquina virtual que especificó.

Control de errores en Python

También puede usar las siguientes convenciones para recuperar varios flujos de los runbooks de Python, incluidos los flujos WARNING, ERROR y DEBUG.

Python

```
print("Hello World output")
print("ERROR: - Hello world error")
print("WARNING: - Hello world warning")
print("DEBUG: - Hello world debug")
print("VERBOSE: - Hello world verbose")
```

En el ejemplo siguiente se muestra esta convención usada en un `try...except` bloque .

Python

```
try:
    raise Exception('one', 'two')
except Exception as detail:
    print ('ERROR: Handling run-time error:', detail)
```

Pasos siguientes

- Para más información sobre los tipos de runbook, sus ventajas y limitaciones, consulte Tipos de runbook de Azure Automation.
- Para más información sobre el desarrollo para Azure con Python, consulte Azure para desarrolladores de Python.

- Para ver ejemplos de runbooks de Python 3, consulte el repositorio de GitHub de Azure Automation.

Last updated on 17/11/2025

Administración de bases de datos en bases de datos de Azure SQL mediante Azure Automation

Artículo • 15/09/2024

En este artículo, se describe el procedimiento para conectar y administrar bases de datos de Azure SQL mediante la identidad administrada asignada por el sistema de Azure Automation. Con Azure Automation puede administrar bases de datos en Azure SQL Database mediante los cmdlets de Az PowerShell más recientes que están disponibles en Azure Az PowerShell.

Azure Automation tiene estos cmdlets de Azure Az PowerShell disponibles directamente para poder realizar todas las tareas de administración de base de datos SQL dentro del servicio. También puede emparejar estos cmdlets en Azure Automation con los cmdlets de otros servicios de Azure para automatizar tareas complejas entre los servicios de Azure y sistemas de terceros.

Azure Automation también puede emitir comandos de T-SQL (Transact SQL) en servidores SQL mediante PowerShell.

Para ejecutar los comandos en la base de datos, es necesario hacer lo siguiente:

- Asegúrese de que la cuenta de Automation tenga una identidad administrada asignada por el sistema.
- Proporcione los permisos adecuados para la identidad administrada de Automation.
- Configure el servidor SQL para usar la autenticación de Microsoft Entra.
- Cree un usuario en el servidor SQL que se asigne a la identidad administrada de la cuenta de Automation.
- Cree un runbook para conectarse y ejecutar los comandos.
- (Opcional) si el servidor SQL está protegido por un firewall, cree una instancia de Hybrid Runbook Worker (HRW), instale los módulos SQL en ese servidor y agregue la dirección IP de HRW a la lista de permitidos en el firewall.

Conéctese a la base de datos de Azure SQL mediante la identidad administrada asignada por el sistema

Para permitir el acceso desde la identidad administrada del sistema de Automation a la base de datos de Azure SQL, siga estos pasos:

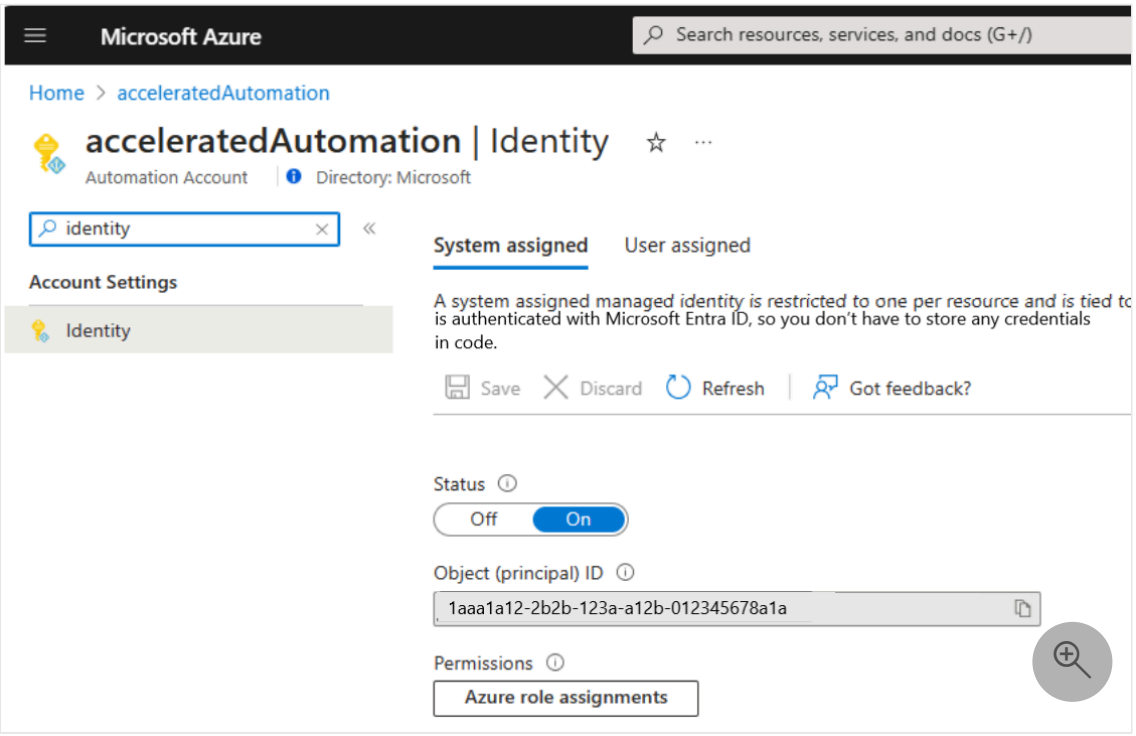
1. Si la identidad administrada del sistema de Automation estuviera establecida en **Desactivado**, haga lo siguiente:

   a. Inicie sesión en Azure Portal.

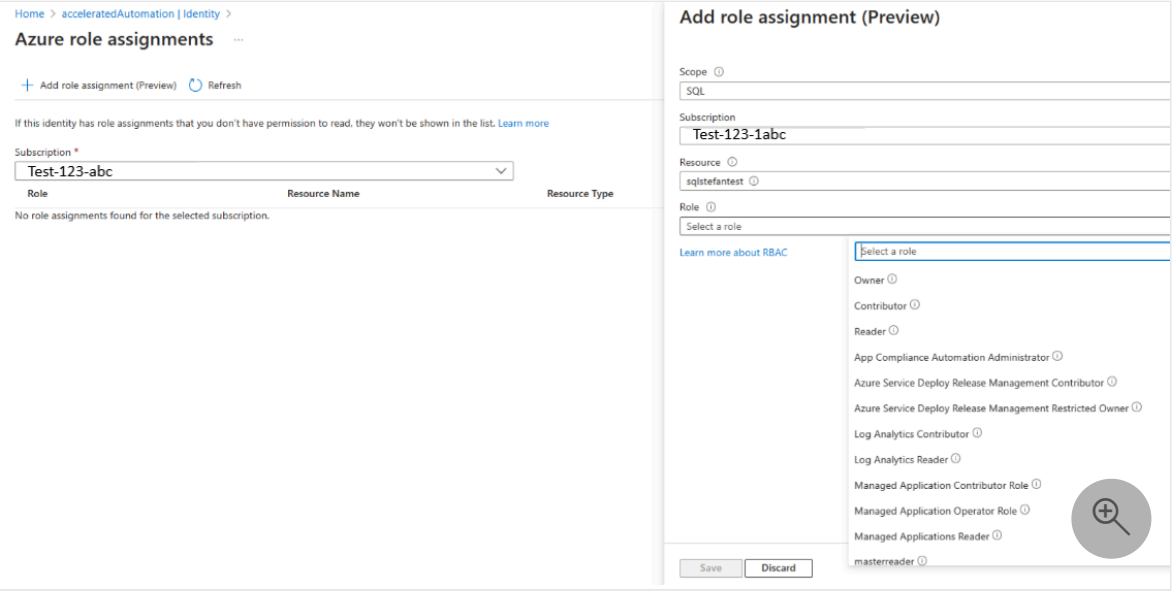
   b. Vaya a su cuenta de Automation.

   c. En la página de la cuenta de Automation, en **Configuración de la cuenta**, seleccione **Identidad**.

   d. En la pestaña **Asignado por el sistema**, seleccione el **Estado** como **Activado**.



2. Una vez que la identidad administrada del sistema esté establecida en **Activado**, deberá proporcionar a la cuenta el acceso necesario mediante estos pasos:
   a. En la página **Cuenta de Automation | Identidad**, en la pestaña **Asignado por el sistema**, en Permisos, seleccione **Asignaciones de roles de Azure**.
   b. En la página Asignaciones de roles de Azure, seleccione **+Agregar asignación de roles (versión preliminar)**.
   c. En **Agregar asignación de roles (versión preliminar)**, establezca el **Ámbito** como *SQL*, seleccione la **Suscripción**, el **Recurso** en la lista desplegable y el **Rol**, según los permisos mínimos necesarios y, a continuación, seleccione **Guardar**.



3. Configure el servidor SQL para la autenticación de Active Directory mediante estos pasos:
   a. Vaya a la página principal de Azure Portal y seleccione **servidores SQL**.
   b. En la página **Servidor SQL**, en **Configuración**, seleccione **Microsoft Entra ID**.
   c. Seleccione **Establecer administrador** para configurar el servidor SQL para la autenticación de AD.

4. Agregue la autenticación en el lado SQL mediante estos pasos:
   a. Vaya a la página principal de Azure Portal y seleccione **servidores SQL**.
   b. En la página **Servidor SQL**, en **Configuración**, seleccione **Bases de datos de SQL Database**.
   c. Seleccione la base de datos para ir a la página base de datos SQL, seleccione **Editor de consultas (versión preliminar)** y ejecute las dos consultas siguientes:

SQL

```
# AutomationAccount - replace with your Automation account's name
# ObjectID - replace with object (principal) ID for your system
managed identity principal from step 1.
CREATE USER "AutomationAccount" FROM EXTERNAL PROVIDER WITH
OBJECT_ID = `ObjectID`
EXEC sp_addrolemember `db_owner`, "AutomationAccount"
```

Código de ejemplo

Conexión a Azure SQL Server

SQL

```
if ($($env:computerName) -eq "Client") {"Runbook running on Azure Client
sandbox"} else {"Runbook running on " + $env:computerName}
Disable-AzContextAutosave -Scope Process
Connect-AzAccount -Identity
$Token = (Get-AZAccessToken -ResourceUrl https://database.windows.net).Token
Invoke-Sqlcmd -ServerInstance azuresqlserverxyz.database.windows.net -
Database MyDBxyz -AccessToken $token -query 'select * from TableXYZ'
```

Compruebe los permisos de la cuenta en el lado SQL

SQL

```
SELECT roles.[name] as role_name, members.name as [user_name]
from sys.database_role_members
Join sys.database_principals roles on
database_role_members.role_principal_id= roles.principal_id
join sys.database_principals members on
database_role_members.member_principal_id=members.principal_id
Order By
roles.[name], members.[name]
```

> ⓘ **Nota**
>
> Cuando un servidor SQL se ejecuta detrás de un firewall, es necesario ejecutar el runbook de Azure Automation en una máquina de su propia red. Asegúrese de configurar esta máquina como Hybrid Runbook Worker para que el firewall no bloquee la dirección IP o la red. Para obtener más información sobre cómo configurar una máquina como Hybrid Worker, consulte crear una instancia de Hybrid Worker.

Uso de Hybrid Worker

Cuando se usa Hybrid Worker, los módulos que usa el runbook deben instalarse localmente desde un símbolo del sistema de PowerShell con privilegios elevados. Por ejemplo, `- Install-module Az.Accounts and Install-module SqlServer`. Para buscar los nombres de módulo necesarios, ejecute un comando en cada cmdlet y, a continuación, compruebe el origen. Por ejemplo, para comprobar el nombre del módulo para el cmdlet `Connect-AzAccounts` que forma parte del módulo Az.Account, ejecute el comando: `get-command Connect-AzAccount`

> ⓘ **Nota**

Se recomienda agregar el código siguiente en la parte superior de cualquier runbook que esté pensado para ejecutarse en instancias de Hybrid Worker: `if ($($env:computerName) -eq "CLIENT") {"Runbook running on Azure CLIENT"} else {"Runbook running on " + $env:computerName}`. El código permite ver el nodo en el que se ejecuta y, en el caso de que se ejecute accidentalmente en la nube de Azure en lugar de la instancia de Hybrid Worker, esto ayudará a determinar el motivo por el que un runbook no funcionó.

Pasos siguientes

- Para obtener más información sobre el uso de credenciales, vea Administración de credenciales en Azure Automation.
- Para obtener más información sobre los módulos, vea Administración de módulos en Azure Automation.
- Si necesita iniciar un runbook, vea Inicio de un runbook en Azure Automation.
- Para obtener información sobre PowerShell, vea Documentación de PowerShell.

Comentarios

¿Le ha resultado útil esta página? Sí No

Proporcionar comentarios sobre el producto | Obtener ayuda en Microsoft Q&A

Configurar los datos basados en la Guía Técnica de Información de Seguridad (STIG)

> Se aplica a: Windows PowerShell 5.1

> **Nota:**
>
> La Configuración de Estado de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration antes de esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

Crear contenido de configuración por primera vez puede ser un reto. En muchos casos, el objetivo es automatizar la configuración de los servidores después de una "línea de base" que, con suerte, se alinea con una recomendación del sector.

> **Nota:**
>
> En este artículo se hace referencia a una solución que se mantiene en la comunidad de código abierto. La compatibilidad solo está disponible en forma de colaboración en GitHub, no con Microsoft.

Proyecto de la comunidad: PowerSTIG

Un proyecto de la comunidad denominado PowerSTIG pretende resolver este problema mediante la generación de contenido DSC basado en la información pública proporcionada sobre la guía de implementación técnica de seguridad (STIG, por sus siglas en inglés).

Gestionar patrones básicos es más complicado de lo que parece. Muchas organizaciones necesitan documentar las excepciones a las reglas y administrar los datos a escala. Para hacer frente a este problema, PowerSTIG proporciona recursos compuestos para abordar cada área de la configuración, en lugar de intentar abordar toda la configuración de un archivo grande.

Después de crear las configuraciones, puede usar los scripts de configuración de DSC para generar archivos MOF y cargarlos en Azure Automation. Para extraer las configuraciones, registre los servidores desde una ubicación local o Azure.

Para probar PowerSTIG, visite la Galería de PowerShell y descargue la solución o seleccione **Sitio del proyecto** para ver la documentación.

Pasos siguientes

- Para comprender DSC de PowerShell, consulte Información general sobre Desired State Configuration de Windows PowerShell.
- Para buscar recursos de DSC de PowerShell, consulte Recursos de DSC.
- Para obtener información detallada sobre la configuración de Configuration Manager, vea Configuración de Configuration Manager local.

Last updated on 26/03/2026

Configurar datos a gran escala para Azure Automation State Configuration

Se aplica a: ✔ Windows PowerShell 5.1

> **Nota:**
>
> State Configuration de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration para esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Importante**
>
> En este artículo se hace referencia a una solución que se mantiene en la comunidad de código abierto. La compatibilidad solo está disponible en forma de colaboración con GitHub, no con Microsoft.

La administración de muchos servidores es un desafío y supone una dificultad para administrar los datos de configuración, ya que implica organizar la información entre construcciones lógicas, como la ubicación, el tipo y el entorno.

Proyecto de la comunidad: Datum

Datum es una solución mantenida por la comunidad creada para resolver este desafío. Datum se basa en las grandes ideas de otras plataformas de administración de la configuración e implementa el mismo tipo de solución para DSC de PowerShell. La información se organiza en archivos de texto basándose en las siguientes en ideas lógicas.

- Configuración que se debe aplicar globalmente
- Configuración que se debe aplicar a todos los servidores de una ubicación

- Configuración que se debe aplicar a todos los servidores de bases de datos
- Configuración de un solo servidor

Configuración de datos a escala

Siga estos pasos para configurar los datos a gran escala para Azure Automation State Configuration:

1. Organice la información en su formato de archivo preferido. Por ejemplo, *JSON*, *Yaml* o *PSD1*.
2. Use los cmdlet para generar archivos de datos de configuración mediante la consolidación de la información de cada archivo en una vista única de un servidor o rol de servidor.
3. Use los archivos generados con scripts de configuración de DSC para generar archivos *MOF* y cargar los archivos MOF en Azure Automation.
4. Para extraer las configuraciones, registre los servidores desde una ubicación local o Azure.

Para descargar la solución, vaya a Galería de PowerShell o seleccione **Sitio de proyecto** para ver la documentación.

Pasos siguientes

- Para comprender DSC de PowerShell, consulte Información general sobre Desired State Configuration de PowerShell.
- Para buscar recursos de DSC de PowerShell, consulte Recursos de DSC.
- Para obtener información detallada sobre la configuración de Configuration Manager, vea Configuración de Configuration Manager local.

Last updated on 17/11/2025

Creación de configuraciones desde servidores existentes

> **Se aplica a:** ✔ Windows PowerShell 5.1

> **Nota:**
>
> State Configuration de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration para esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Importante**
>
> Este artículo hace referencia a una solución que se mantiene en la comunidad de código abierto. La compatibilidad solo está disponible en forma de colaboración en GitHub, no con Microsoft.

En este artículo se explica cómo crear configuraciones desde servidores existentes para una configuración de estado de Azure Automation. Crear configuraciones a partir de un servidor existente puede ser una tarea complicada. Debe conocer la configuración correcta y el orden de aplicación para asegurarse de que la configuración sea correcta.

Proyecto de la comunidad: ReverseDSC

ReverseDSC es una solución mantenida por la comunidad creada para trabajar en esta área a partir de SharePoint. La solución se basa en el recurso SharePointDSC y la amplía para orquestar mediante la recopilación de información de los servidores existentes que ejecutan SharePoint.

La versión más reciente tiene varios modos de extracción para determinar el nivel de información que se va a incluir. El resultado de usar la solución es la generación de datos de configuración que deben usarse con los scripts de configuración de SharePointDSC.

Creación de configuraciones desde servidores existentes para la configuración de estado de Azure Automation

Siga estos pasos para crear una configuración a partir de servidores existentes:

1. Después de generar los archivos de datos, use los archivos con scripts de configuración de DSC para generar archivos **MOF**.
2. Cargue los archivos MOF en Azure Automation.
3. Para extraer las configuraciones, registre los servidores desde una ubicación local o Azure.

Para más información sobre ReverseDSC, visite la Galería de PowerShell y descargue la solución o seleccione **Sitio del proyecto** para ver la documentación.

Pasos siguientes

- Para comprender DSC de PowerShell, consulte Información general sobre Desired State Configuration de PowerShell.
- Para buscar recursos de DSC de PowerShell, consulte Recursos de DSC.
- Para obtener información detallada sobre la configuración de Configuration Manager, vea Configuración de Configuration Manager local.

Last updated on 17/11/2025

Conversión de configuraciones en recursos compuestos

> **Se aplica a:** ✔ Windows PowerShell 5.1

> **Nota:**
>
> State Configuration de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration para esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Importante**
>
> En este artículo se hace referencia a una solución que mantiene la comunidad de código abierto y la compatibilidad solo está disponible en forma de colaboración de GitHub, no de Microsoft.

En este artículo se explica cómo crear configuraciones para administrar los escenarios siguientes:

- Crear un servidor web
- Crear un servidor de nombres de dominio
- Crear un servidor que ejecute SharePoint
- Configurar un clúster de SQL
- Administración de la configuración del firewall
- Administrar la configuración de la contraseña

Se recomienda empaquetar la configuración como un recurso compuesto antes de compartirla con otros usuarios, ya que crear recursos compuestos por primera vez puede ser un esfuerzo

tedioso.

Proyecto de comunidad: CompositeResource

Un CompositeResource es una solución mantenida por la comunidad que automatiza el proceso de creación de un nuevo módulo a partir de la configuración.

Creación de un módulo de recursos compuestos

Siga los pasos para crear un módulo de recursos compuesto:

1. Asegúrese de que el script de configuración está cargado en su sesión PowerShell mediante el origen de puntos del script.
2. En lugar de ejecutar la configuración para generar un archivo **MOF**, use el comando proporcionado por el módulo **CompositeResource** para automatizar una conversión. El comando carga el contenido de la configuración, obtiene la lista de parámetros y genera un nuevo módulo. Después de generar un módulo, puede incrementar la versión y agregar notas de lanzamiento cada vez que realice cambios y publicarlo en su propio repositorio PowerShellGet.
3. Use el módulo en la Composable Authoring Experience en Azure, o agréguelos a los scripts de configuración de DSC para generar archivos MOF y cargar los archivos MOF a Azure Automation.
4. Registre los servidores desde una ubicación local o en Azure para extraer las configuraciones.

> **Nota:**
>
> La actualización más reciente del proyecto también ha publicado runbooks para que Azure Automation automatice el proceso de importación de configuraciones desde la Galería de PowerShell.

Para obtener más información sobre cómo automatizar la creación de recursos compuestos para DSC, consulte Galería de PowerShell y descargue la solución o seleccione **Sitio de proyecto** para ver la documentación.

Pasos siguientes

- Para comprender DSC de PowerShell, consulte Información general sobre Desired State Configuration de PowerShell.

- Para obtener más información sobre los recursos de DSC de PowerShell, consulte Recursos de DSC.
- Para obtener información sobre la configuración del administrador de configuración local, consulte Configuración del administrador de configuración local.

Last updated on 17/11/2025

Introducción a la autenticación de cuentas de Azure Automation

Azure Automation le permite automatizar tareas en recursos de Azure, locales y de otros proveedores de servicios en la nube como Amazon Web Services (AWS). Puede usar runbooks para automatizar las tareas, o bien una instancia de Hybrid Runbook Worker si tiene que administrar procesos empresariales u operativos fuera de Azure. Para trabajar en cualquiera de estos entornos, se requieren permisos para proteger el acceso a los recursos con los derechos mínimos necesarios.

En este artículo se tratan los distintos escenarios de autenticación que se admiten en Azure Automation y se indica cómo empezar a trabajar según los entornos que haya que administrar.

Cuenta de Automatización

Cuando inicia Azure Automation por primera vez, debe crear al menos una cuenta de Automation. Las cuentas de Automation le permiten aislar sus recursos, runbooks, activos y configuraciones de Automation de los recursos de otras cuentas. Puede usar cuentas de Automation para separar los recursos en entornos lógicos independientes o responsabilidades delegadas. Por ejemplo, puede usar una cuenta para desarrollo, otra para producción y otra para su entorno local.

Una cuenta de Azure Automation es diferente de su cuenta Microsoft o de las cuentas creadas en su suscripción de Azure. Para obtener una introducción a la creación de una cuenta de Azure Automation, consulte Crear una cuenta de Azure Automation.

Recursos de Automatización

Los recursos de Automation de cada cuenta de Automation están asociados a una sola región de Azure, pero la cuenta puede administrar todos los recursos de la suscripción de Azure. El principal motivo para crear cuentas de Automation en distintas regiones es si tiene directivas que requieren que los datos y recursos se aíslen en una región específica.

Todas las tareas que se crean en los recursos con Azure Resource Manager y los cmdlets de PowerShell en Azure Automation deben autenticarse en Azure mediante la autenticación basada en credenciales de la identidad organizativa de Microsoft Entra.

Identidades administradas

Una identidad administrada de Microsoft Entra ID permite al runbook acceder fácilmente a otros recursos protegidos por Microsoft Entra. La plataforma Azure administra la identidad y no es necesario que lleve a cabo el aprovisionamiento ni la rotación de los secretos. Para más información sobre las identidades administradas en Microsoft Entra ID, consulte Identidades administradas para recursos de Azure.

Las identidades administradas son la manera recomendada de autenticarse en los runbooks y es el método de autenticación predeterminado para la cuenta de Automation.

Estas son algunas de las ventajas de usar las identidades administradas:

- Las identidades administradas se pueden usar sin ningún costo adicional.
- Puede acceder a los recursos con la identidad administrada de la cuenta de Automation desde un runbook sin crear certificados, conexiones, etc.

Una cuenta de Automation puede autenticarse mediante dos tipos de identidades administradas:

- Una identidad asignada por el sistema está asociada a la aplicación y se elimina si se elimina la aplicación. Una aplicación solo puede tener una identidad asignada por el sistema.
- Una identidad asignada por el usuario es un recurso de Azure independiente que puede asignarse a la aplicación. Una aplicación puede tener varias identidades asignadas por el usuario.

> **Nota:**
>
> Las identidades asignadas por el usuario solo se admiten para trabajos en la nube. Para más información sobre las distintas identidades administradas, vea Administración de tipos de identidad.

Para más información acerca del uso de identidades administradas, consulte Habilitación de la identidad administrada para Azure Automation.

Permisos de suscripción

Necesitas el permiso `Microsoft.Authorization/*/Write`. Este permiso se obtiene mediante la pertenencia a uno de los siguientes roles integrados de Azure:

- Propietario
- Administrador de acceso de usuario

Para más información sobre los permisos de suscripción clásicos, consulte Administradores de la suscripción clásica de Azure.

Permisos de Microsoft Entra

Para poder renovar la entidad de servicio, debe ser miembro de uno de los siguientes roles integrados de Microsoft Entra:

- Administrador de aplicaciones
- Desarrollador de aplicaciones

La pertenencia se puede asignar a **TODOS** los usuarios del inquilino en el nivel de directorio, que es el comportamiento predeterminado. Puede otorgar membresía a cualquiera de los roles a nivel de directorio. Para más información, consulte ¿Quién tiene permiso para agregar aplicaciones a la instancia de Microsoft Entra?

Permisos de la cuenta de Automation

Para poder actualizar la cuenta de Automation, debe ser miembro de uno de los siguientes roles de la cuenta de Automation:

- Propietario
- Colaborador

Para obtener más información sobre los modelos de implementación clásico y de Azure Resource Manager, consulte Implementación de Resource Manager e implementación clásica.

> ⓘ **Nota:**
>
> Las suscripciones del Proveedor de soluciones en la nube (CSP) de Azure solo admiten el modelo de Azure Resource Manager. Los servicios que no son de Azure Resource Manager no están disponibles en el programa. Para más información acerca de las suscripciones de CSP, consulte **Servicios disponibles en las suscripciones de CSP**.

Autenticación de Runbook con Hybrid Runbook Worker

Los runbooks que se ejecutan en un Hybrid Runbook Worker en su centro de datos o en los servicios de computación de otros entornos en la nube, como AWS, no pueden usar el mismo método que se emplea normalmente para los runbooks que se autentican en recursos de

Azure. Esto se debe a que esos recursos se ejecutan fuera de Azure y, por lo tanto, requieren sus propias credenciales de seguridad definidas en Automatización para la autenticación en los recursos a los que tienen acceso localmente. Para más información sobre la autenticación de runbooks con trabajos de runbook, consulte Ejecución de runbooks en Hybrid Runbook Worker.

En el caso de los runbooks que usan Hybrid Runbook Workers en máquinas virtuales de Azure, puede usar la autenticación de runbook con identidades administradas para autenticarse con los recursos de Azure.

Pasos siguientes

- Para crear una cuenta de Automation desde Azure Portal, consulte Creación de una cuenta independiente de Azure Automation.
- Si prefiere crear su cuenta mediante una plantilla, consulte Creación de una cuenta de Automation con una plantilla de Azure Resource Manager.
- Para la autenticación con Amazon Web Services, consulte Autenticar runbooks con Amazon Web Services.
- Para ver una lista de los servicios de Azure que admiten la característica Managed Identities for Azure Resources, consulte Services that support managed identities for Azure resources (Servicios que admiten la característica Managed Identities for Azure Resources).

Last updated on 17/11/2025

Ejecución de runbooks en Azure Automation

La automatización de procesos en Azure Automation permite crear y administrar PowerShell, flujo de trabajo de PowerShell y runbooks gráficos. Para obtener más información, consulte Azure Automation runbooks.

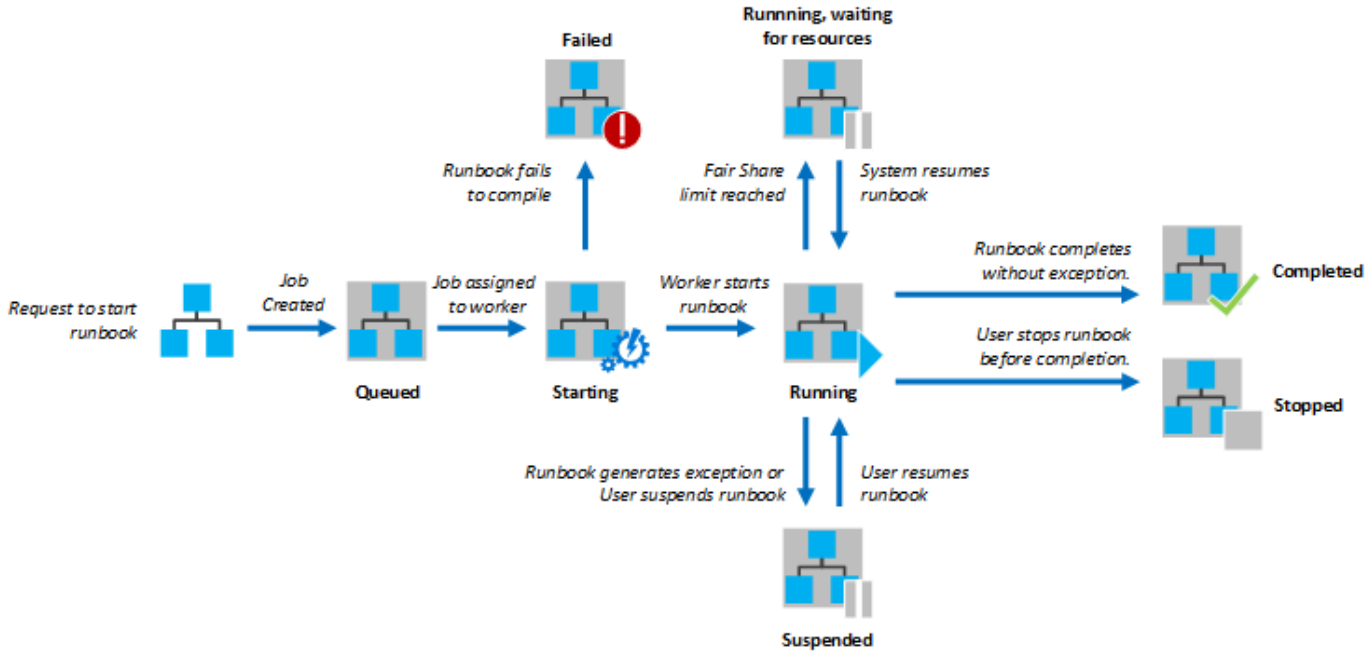
Automation ejecuta los runbooks según la lógica definida dentro de ellos. Si se interrumpe un runbook, se reinicia desde el principio. Este comportamiento requiere que escriba runbooks que admitan el reinicio si se producen problemas transitorios.

Al iniciar un runbook en Azure Automation se crea un trabajo, que es una única instancia de ejecución del runbook. Cada trabajo accede a recursos de Azure estableciendo una conexión a tu suscripción de Azure. El rol de trabajo solo accede a los recursos de su centro de datos si se puede acceder a estos recursos desde la nube pública.

Azure Automation asigna un trabajador para ejecutar cada trabajo durante la ejecución del runbook. Aunque varias cuentas de Automation comparten los trabajos, los trabajos de distintas cuentas de Automation están aislados entre sí. No puede controlar qué servicios de rol de trabajo solicita su trabajo.

Cuando se visualiza la lista de runbooks en el portal de Azure, se muestra el estado de cada tarea que se ha iniciado para cada runbook. Azure Automation almacena los registros de trabajo durante un máximo de 30 días.

En el siguiente diagrama se muestra el ciclo de vida de un trabajo de runbook para runbooks de PowerShell, runbooks de flujo de trabajo de PowerShell y runbooks gráficos.



> ⓘ **Nota:**
>
> Para obtener información sobre cómo ver o eliminar datos personales, consulte Solicitudes generales de interesados para el RGPD, Solicitudes de interesados de Azure para el RGPD o Solicitudes de interesados de Windows para el RGPD, en función de su área y necesidades específicas. Para obtener más información sobre el RGPD, consulte la sección GDPR del Centro de confianza de Microsoft y la sección GDPR del portal de confianza de servicios.

Entorno de ejecución de runbooks

Los runbooks de Azure Automation se pueden ejecutar en un espacio aislado de Azure o en un Hybrid Runbook Worker.

Cuando los runbooks están diseñados para autenticarse y ejecutarse en recursos de Azure, se ejecutan en un espacio aislado de Azure. Azure Automation asigna un trabajador para ejecutar cada trabajo durante la ejecución del runbook en el sandbox. Aunque varias cuentas de Automation comparten los trabajos, los trabajos de distintas cuentas de Automation están aislados entre sí. Los trabajos con el mismo espacio aislado están sujetos a las limitaciones de recursos de dicho espacio. El entorno de espacio aislado Azure no admite operaciones interactivas.

También puede usar una instancia de Hybrid Runbook Worker para ejecutar runbooks directamente en el equipo que hospeda el rol y en los recursos locales del entorno. Azure

Automation almacena y administra runbooks y, a continuación, los entrega a uno o varios equipos asignados.

Habilitar el Azure Firewall en Azure Storage, Azure Key Vault o Azure SQL bloquea el acceso de los runbooks de Azure Automation para esos servicios. El acceso se bloqueará incluso cuando la excepción del firewall para permitir los servicios de Microsoft de confianza esté habilitada, ya que Automatización no forma parte de la lista de servicios de confianza. Con un firewall habilitado, el acceso solo se puede realizar mediante una instancia de Hybrid Runbook Worker y un punto de conexión de servicio de red virtual.

En la tabla siguiente se enumeran algunas tareas de ejecución de runbooks con el entorno de ejecución recomendado que se muestra para cada una.

Expandir tabla

Tarea	Recomendación	Notas
Integración con recursos de Azure	Sandbox de Azure	Hospedada en Azure, la autenticación es más sencilla. Si está usando un Hybrid Runbook Worker en una máquina virtual de Azure, puede utilizar la autenticación de runbook con identidades administradas.
Obtención de un rendimiento óptimo para administrar recursos de Azure	Sandbox de Azure	El script se ejecuta en el mismo entorno, que tiene menos latencia.
Minimizar los costos operativos	Sandbox de Azure	No hay ninguna sobrecarga de proceso y no es necesario para una máquina virtual.
Ejecución de script de ejecución prolongada	Hybrid Runbook Worker	Los espacios aislados de Azure tienen límites de recursos.
Interacción con servicios locales	Hybrid Runbook Worker	Acceda directamente a la máquina host, a los recursos de otros entornos en la nube o en el entorno local.
Requerimiento de software de terceros y archivos ejecutables	Hybrid Runbook Worker	Usted administra el sistema operativo y puede instalar el software.
Supervisar un archivo o carpeta con un runbook	Hybrid Runbook Worker	Use una tarea del Monitor en una instancia de Hybrid Runbook Worker.
Ejecución de un script con un uso intensivo de recursos	Hybrid Runbook Worker	Los espacios aislados de Azure tienen límites de recursos.
Uso de módulos con requisitos específicos	Hybrid Runbook Worker	Algunos ejemplos son: WinSCP: dependencia de winscp.exe

Tarea	Recomendación	Notas
		Administración de IIS: dependencia de la habilitación o administración de IIS
Instalación de un módulo con un instalador	Hybrid Runbook Worker	Los módulos para espacios aislados deben admitir copias.
Uso de runbooks o módulos que requieren .NET Framework con una versión distinta de 4.7.2	Hybrid Runbook Worker	Azure los entornos de prueba admiten .NET Framework 4.7.2 y no se admite la actualización a otra versión.
Ejecución de scripts que requieren elevación	Hybrid Runbook Worker	Los espacios aislados no permiten elevación. Con una instancia de Hybrid Runbook Worker, puede desactivar UAC y usar Invoke-Command al ejecutar el comando que requiere elevación.

Almacenamiento temporal en un espacio aislado

Si necesita crear archivos temporales como parte de la lógica del runbook, puede usar la carpeta Temp (es decir, `$env:TEMP`) en el espacio aislado de Azure para runbooks que se ejecutan en Azure. La única limitación es que no se pueden usar más de 1 GB de espacio en disco, que es la cuota de cada espacio aislado. Al trabajar con flujos de trabajo de PowerShell, este escenario puede generar un problema porque los flujos de trabajo de PowerShell usan puntos de control y el script se podría volver a intentar en otro espacio aislado.

Con el espacio aislado híbrido, puede usar `C:\temp` en función de la disponibilidad del almacenamiento en una instancia de Hybrid Runbook Worker. Sin embargo, según las recomendaciones de máquina virtual de Azure, no debería usar el disco temporal en Windows o Linux para datos que necesitan persistir.

Recursos

Los runbooks deben incluir lógica para tratar con recursos, por ejemplo, las máquinas virtuales, la red y los recursos de la red. Los recursos están vinculados a una suscripción de Azure y los runbooks requieren credenciales adecuadas para acceder a cualquier recurso. Para ver un ejemplo de cómo administrar los recursos de un runbook, consulte el artículo sobre administración de recursos.

Seguridad

Azure Automation usa el Microsoft Defender para la nube para proporcionar seguridad a los recursos y detectar riesgos en los sistemas Linux. La seguridad se asegura en todas sus cargas de trabajo, independientemente de si los recursos están en Azure o no. Consulte Introducción a la autenticación en Azure Automation.

Defender for Cloud aplica restricciones a los usuarios que pueden ejecutar scripts, firmados o sin firmar, en una máquina virtual. Si es un usuario con acceso raíz a una máquina virtual, debe configurar explícitamente la máquina con una firma digital o desactivarla. De lo contrario, solo puede ejecutar un script para aplicar actualizaciones del sistema operativo después de crear una cuenta de Automation y habilitar la característica adecuada.

Suscripciones

Un Azure subscription es un contrato con Microsoft usar uno o varios servicios basados en la nube, por los que se le cobra. Puede administrar varias suscripciones desde la misma cuenta de Automation si la credencial que usa tiene acceso a varias suscripciones.

Credenciales

Un runbook requiere credenciales adecuadas para acceder a cualquier recurso, sea de Azure o de sistemas de terceros. Estas credenciales se almacenan en Azure Automation, Key Vault, etc.

Azure Monitor

Azure Automation puede usar Azure Monitor para supervisar sus operaciones de máquina.

Permisos de runbooks

Un runbook necesita permisos para la autenticación para Azure, a través de credenciales. Consulte Azure Automation visión general de la autenticación.

Módulos

Azure Automation incluye los siguientes módulos de PowerShell:

- Orchestrator.AssetManagement.Cmdlets: contiene varios cmdlets internos que solo están disponibles cuando se ejecutan runbooks en el entorno de espacio aislado de Azure o en un Windows Hybrid Runbook Worker. Estos cmdlets están diseñados para usarse en lugar de

los cmdlets de Azure PowerShell para interactuar con los recursos de tu cuenta de Automation.

- Az.Automation: el módulo de PowerShell recomendado para interactuar con Azure Automation que reemplaza el módulo AzureRM Automation. El módulo Az.Automation no se incluye automáticamente al crear una cuenta de Automation y debe importarlos manualmente.
- AzureRM.Automation: Se instala de manera predeterminada al crear una cuenta de Automation.

También se admiten módulos instalables, en función de los cmdlets que requieren los runbooks y las configuraciones de DSC. Para obtener más información sobre los módulos disponibles para las configuraciones de runbooks y DSC, consulte Administrar módulos en Azure Automation.

Certificados

Azure Automation usa certificates para la autenticación para Azure o los agrega a recursos de Azure o de terceros. Los certificados se almacenan de forma segura para el acceso mediante runbooks y configuraciones de DSC.

Los runbooks pueden usar certificados autofirmados, que no están firmados por una entidad de certificación (CA). Consulte Creación de un certificado nuevo.

Trabajos

Azure Automation admite un entorno para ejecutar trabajos desde la misma cuenta de Automation. Un runbook individual puede tener muchos trabajos que se ejecutan al mismo tiempo. Cuantos más trabajos se ejecuten al mismo tiempo, más a menudo se podrán enviar al mismo espacio aislado. Un máximo de 10 trabajos se puede ejecutar en un espacio aislado. Se quitará un espacio aislado cuando no se ejecute ningún trabajo en él; Por lo tanto, no se debe usar para guardar archivos.

Los trabajos que se ejecutan en el mismo proceso de espacio aislado pueden afectarse entre sí. Un ejemplo es la ejecución del cmdlet Disconnect-AzAccount. La ejecución de este cmdlet desconecta cada trabajo de runbook en el proceso de espacio aislado compartido. Para ver un ejemplo de cómo trabajar en este escenario, consulte Prevención de trabajos simultáneos.

> **Nota:**

Los trabajos de PowerShell iniciados desde un runbook que se ejecuta en un espacio aislado de Azure podrían no ejecutarse en el modo completo de lenguaje PowerShell.

Estados del trabajo

En la tabla siguiente se describen los estados posibles para un trabajo. Puede ver un resumen de estado de todos los trabajos de runbook o profundizar en detalles de un trabajo de runbook específico en el portal de Azure. También puede configurar la integración con el área de trabajo de Log Analytics para reenviar flujos de trabajos y el estado del trabajo del runbook. Para obtener más información sobre la integración con registros de Azure Monitor, consulte Reenviar el estado del trabajo y los flujos de trabajo de Automatización a registros de Azure Monitor. Consulte también Obtención de estados del trabajo para ver un ejemplo de cómo trabajar con estados en un runbook.

Expandir tabla

Estado	Descripción
En activación	El trabajo se está activando.
Completed	El trabajo se completó correctamente.
Con error	Un runbook de flujo de trabajo de PowerShell o gráfico no se pudo compilar. Un runbook de PowerShell no se pudo iniciar o el trabajo generó una excepción. Consulte Tipos de runbooks de Azure Automation.
Error, esperando recursos	Se produjo un error en el trabajo porque ha alcanzado el límite de distribución equilibrada tres veces y se ha iniciado en el mismo punto de control o en el inicio del runbook cada vez.
En cola	El trabajo espera que los recursos en un trabajo de Automation estén disponibles para que se pueda iniciar.
Reanudando	El sistema está reanudando el trabajo después de que se suspendió.
En ejecución	El trabajo se está ejecutando.
En ejecución, esperando recursos	El trabajo se ha descargado porque ha alcanzado el límite de distribución equilibrada. Se reanudará en breve desde su último punto de control.
Iniciando	El trabajo se ha asignado a un trabajador y el sistema lo está iniciando.
Detenido	El trabajo lo detuvo el usuario antes de completarse.
Deteniéndose	El sistema está deteniendo el trabajo.

Estado	Descripción
Suspended	Solo se aplica a runbooks de flujo de trabajo de PowerShell y gráficos. El trabajo lo ha suspendido el usuario, el sistema o un comando en el runbook. Si un runbook no tiene un punto de control, se inicia desde el principio. Si lo tiene, puede volver a empezar y reanudarse desde su último punto de comprobación. El sistema solo suspende el runbook cuando se produce una excepción. De forma predeterminada, la variable `ErrorActionPreference` se establece en Continue, lo que indica que el trabajo sigue ejecutándose tras un error. Si la variable de preferencia se establece en Stop, se suspende el trabajo tras un error.
Suspendiendo	Solo se aplica a runbooks de flujo de trabajo de PowerShell y gráficos. El sistema intenta suspender el trabajo a petición del usuario. El runbook debe alcanzar su siguiente punto de control antes de que se pueda suspender. Si ya ha pasado su último punto de control, se completa antes de que se pueda suspender.
Nuevo	El trabajo se ha enviado recientemente, pero aún no está activado.

> **Nota:**
>
> En caso de error de infraestructura, el trabajo se reintenta internamente durante un máximo de 3 veces.

Registro de actividad

La ejecución de runbooks en Azure Automation escribe los detalles en un registro de actividad de la cuenta de Automation. Para más información sobre el uso del registro, consulte Recuperación de detalles del registro de actividad.

Excepciones

En esta sección se describen algunas maneras de controlar excepciones o problemas intermitentes en los runbooks. Un ejemplo es una excepción de WebSocket. El control de excepciones correcto evita que los errores de red transitorios provoquen un error en los runbooks.

ErrorActionPreference

La variable ErrorActionPreference determina la manera en que PowerShell responde a un error de no terminación. Los errores de terminación siempre terminan y no se ven afectados por `ErrorActionPreference`.

Cuando el runbook usa `ErrorActionPreference`, un error normalmente de no terminación, como `PathNotFound` del cmdlet Get-ChildItem, impide que el runbook se complete. En el siguiente ejemplo se muestra el uso de `ErrorActionPreference`. El comando final Write-Output nunca se ejecuta, a medida que se detiene el script.

PowerShell

```
$ErrorActionPreference = 'Stop'
Get-ChildItem -path nofile.txt
Write-Output "This message will not show"
```

Try Catch Finally

Try Catch Finally se usa en los scripts de PowerShell para controlar los errores de terminación. El script puede usar este mecanismo para detectar excepciones específicas o excepciones generales. La instrucción `catch` se debe usar para realizar el seguimiento de los errores o para intentar controlarlos. En el ejemplo siguiente se intenta descargar un archivo que no existe. Detecta la excepción `System.Net.WebException` y devuelve el último valor para cualquier otra excepción.

PowerShell

```
try
{
   $wc = new-object System.Net.WebClient
   $wc.DownloadFile("http://www.contoso.com/MyDoc.doc")
}
catch [System.Net.WebException]
{
    "Unable to download MyDoc.doc from http://www.contoso.com."
}
catch
{
    "An error occurred that could not be resolved."
}
```

Throw

Throw se puede usar para generar un error de terminación. Este mecanismo puede ser útil al definir su propia lógica en un runbook. Si el script cumple un criterio que debe detenerlo, puede usar la instrucción `throw` para la detención. En el ejemplo siguiente se utiliza esta instrucción para mostrar un parámetro de función necesario.

PowerShell

```
function Get-ContosoFiles
{
  param ($path = $(throw "The Path parameter is required."))
  Get-ChildItem -Path $path\*.txt -recurse
}
```

Errors

Los runbooks deben controlar los errores. Azure Automation admite dos tipos de errores de PowerShell, de terminación y de no terminación.

Los errores de terminación detienen la ejecución del runbook cuando se producen. El runbook se detiene con un estado de trabajo de Error.

Los errores de no terminación permiten que un script continúe después de que sucedan. Un ejemplo de error de no terminación es aquel donde un runbook usa el cmdlet `Get-ChildItem` con una ruta de acceso que no existe. PowerShell se ve que la ruta de acceso no existe, genera un error y pasa a la carpeta siguiente. En este caso, el error no establece el estado del trabajo del runbook en Con error y el trabajo podría incluso completarse. Para forzar a un runbook a detenerse ante un error de no terminación, puede usar `ErrorAction Stop` en el cmdlet.

Llamada a procesos

Los runbooks que se ejecutan en el sandbox de Azure no admiten la llamada a procesos, como archivos ejecutables (**.exe**) o subprocesos. El motivo de esto es que un espacio aislado de Azure es un proceso compartido que se ejecuta en un contenedor, el cual tal vez no pueda acceder a todas las APIs subyacentes. En escenarios donde se requiere software de terceros o llamadas a subprocesos, debe ejecutar un runbook en una instancia de Hybrid Runbook Worker.

Características de dispositivos y aplicaciones

Los trabajos de runbook en espacios aislados de Azure no pueden acceder a las características del dispositivo o la aplicación. La API más común que se usa para consultar las métricas de rendimiento en Windows es WMI, con algunas de las métricas comunes que son memoria y uso de CPU.

webhooks

Los servicios externos, como Azure DevOps Services y GitHub, pueden iniciar un runbook en Azure Automation. Para realizar este tipo de inicio, el servicio usa un webhook a través de una

única solicitud HTTP. El uso de un webhook permite iniciar runbooks sin implementar una característica completa de Azure Automation.

Recursos compartidos

Para compartir recursos entre todos los runbooks de la nube, Azure usa un concepto denominado recurso compartido justo. Con distribución equilibrada, Azure descarga temporalmente o detiene cualquier trabajo que se haya ejecutado durante más de tres horas. Los trabajos de runbooks de PowerShell y runbooks de Python se detienen y no se reinician, y el estado del trabajo cambia a Detenido.

Para las tareas de Azure Automation de larga duración, se recomienda usar un Hybrid Runbook Worker. Las instancias de Hybrid Runbook Worker no están limitadas por la distribución equilibrada y no tienen una limitación del tiempo que se puede ejecutar un runbook. Los demás límites de trabajo se aplican a los espacios aislados de Azure y a Hybrid Runbook Workers. Aunque las instancias de Hybrid Runbook Worker no están restringidas por el límite de distribución equilibrada de tres horas, debe desarrollar los runbooks para que se ejecutan en los roles de trabajo que admitan reinicios si se producen problemas inesperados en la infraestructura local.

Otra opción es optimizar un runbook mediante runbooks secundarios. Por ejemplo, el runbook podría recorrer en bucle la misma función en varios recursos, como, por ejemplo, una operación de base de datos en varias bases de datos. Puede mover esta función a un runbook secundario y hace que el runbook lo llame mediante Start-AzAutomationRunbook. Los runbooks secundarios se ejecutan en paralelo en procesos independientes.

El uso de runbooks secundarios reduce la cantidad total de tiempo que tarda en completarse el runbook primario. El runbook puede usar el cmdlet Get-AzAutomationJob para comprobar el estado del trabajo de un runbook secundario si aún hay más operaciones después de que se complete el secundario.

Pasos siguientes

- Para empezar a trabajar con runbooks de PowerShell, consulte Tutorial: Creación de un runbook de PowerShell.
- Para trabajar con runbooks, consulte Administrar runbooks en Azure Automation.
- Para más información sobre PowerShell, consulte PowerShell Docs.
- Para ver una referencia de los cmdlets de PowerShell, consulte Az.Automation.

- Para solucionar problemas relacionados con los recursos compartidos durante la ejecución del runbook de Azure Automation, consulte Solución de problemas de recursos compartidos de Azure Automation.
- Para solucionar problemas durante la ejecución del runbook de Azure Automation, consulte Solucionar problemas de runbook.

Last updated on 01/05/2026

Introducción a Hybrid Runbook Worker de Automation

22/07/2025

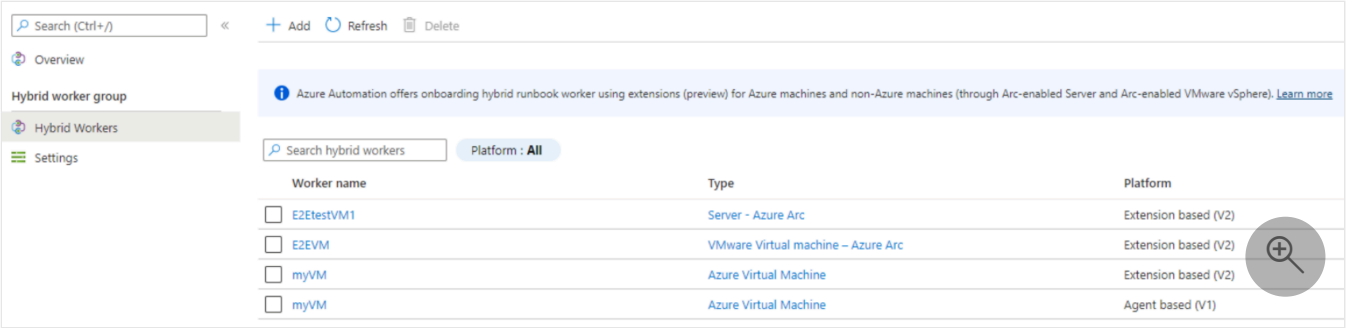
> **Importante**
>
> - A partir del 1 de abril de 2025, se detendrán todos los trabajos que se ejecutan en Hybrid Worker basado en agente.
> - Hybrid Runbook Worker (Windows y Linux) basado en agente de Azure Automation se ha retirado el **31 de agosto de 2024** y ya no se admite. Siga las instrucciones sobre cómo migrar desde una instancia de Hybrid Runbook Worker basada en agente existente a Hybrid Worker basado en extensiones.

Es posible que los runbooks de Azure Automation no tengan acceso a los recursos de otras nubes o del entorno local, porque se ejecutan en plataforma de nube de Azure. La característica Hybrid Runbook Worker de Azure Automation permite ejecutar runbooks directamente en la máquina que hospeda el rol y en los recursos del entorno para administrar dichos recursos locales. Los runbooks se almacenan y administran en Azure Automation y después se entregan a una o más máquinas asignadas.

Azure Automation proporciona una integración nativa del rol Hybrid Runbook Worker a través del marco de extensión de máquina virtual (VM) de Azure mediante la instalación de la extensión de máquina virtual de Hybrid Runbook Worker. El agente de máquina virtual de Azure es responsable de la administración de extensiones en máquinas virtuales de Azure que operan con Windows y Linux, y el agente de Azure Connected Machine en máquinas que no son de Azure, incluidos servidores habilitados para Azure Arc y VMware vSphere habilitado para Azure Arc. Ahora hay dos plataformas de instalación de Hybrid Runbook Worker:

Expandir tabla

Plataforma	Descripción
Basado en extensiones (V2)	Se instala mediante la extensión de máquina virtual Hybrid Runbook Worker, sin ninguna dependencia en el agente de Log Analytics que informa a un área de trabajo de Log Analytics para Azure Monitor. **Esta es la plataforma admitida**.
Basado en agentes (V1) - Retirado	Se instala después de que se completa el agente de Log Analytics que informa a un área de trabajo de Log Analytics de Azure Monitor.



Para las operaciones de Hybrid Runbook Worker después de la instalación, el proceso de ejecución de runbooks en Hybrid Runbook Worker es el mismo. El propósito del enfoque basado en extensiones es simplificar la instalación y administración del rol Hybrid Runbook Worker y eliminar la complejidad del trabajo con la versión basada en agente. La nueva instalación basada en extensiones no afecta a la instalación o administración de un rol de Hybrid Runbook Worker basado en agente. Ambos tipos de Hybrid Runbook Worker pueden coexistir en la misma máquina.

Ventajas de las instancias de Hybrid Worker de usuario basadas en extensiones

El enfoque basado en extensiones simplifica considerablemente la instalación y administración de Hybrid Runbook Worker de usuario, lo que elimina la complejidad de trabajar con el enfoque basado en agente. Estas son algunas ventajas clave:

- **Incorporación sin problemas**: el enfoque basado en agente para la incorporación de Hybrid Runbook Worker depende del agente de Log Analytics, que es un proceso con varios pasos, lento y propenso a errores. El enfoque basado en extensiones ya no depende del agente de Log Analytics.
- **Facilidad de administración**: integración nativa con la identidad de ARM para Hybrid Runbook Worker y proporciona flexibilidad para la gobernanza a escala mediante directivas y plantillas.
- **Autenticación basada en Microsoft Entra ID**: usa identidades administradas asignadas por el sistema de máquina virtual proporcionadas por Microsoft Entra ID. Esto centraliza el control y la administración de identidades y credenciales de recursos.
- **Experiencia unificada**: ofrece una experiencia idéntica para administrar máquinas habilitadas para Azure Arc en Azure y fuera de Azure.
- **Varios canales de incorporación**: puede optar por incorporar y administrar los trabajos basados en extensiones a través de Azure Portal, los cmdlets de PowerShell, Bicep, las plantillas de ARM, la API de REST y la CLI de Azure. También puede instalar la extensión en una máquina virtual de Azure existente o en un servidor habilitado para Arc en la experiencia de Azure Portal de esa máquina a través de la hoja Extensiones.

- **Actualización automática predeterminada**: ofrece la actualización automática de las versiones secundarias de forma predeterminada, lo que reduce significativamente la capacidad de administración de mantenerse actualizado en la versión más reciente. Se recomienda habilitar las actualizaciones automáticas para aprovechar las ventajas de cualquier actualización de características o seguridad sin la sobrecarga manual. También puede dejar de participar en las actualizaciones automáticas en cualquier momento. Actualmente, no se admiten las actualizaciones de versiones principales y se deben administrar manualmente.

Límites de Runbook Worker

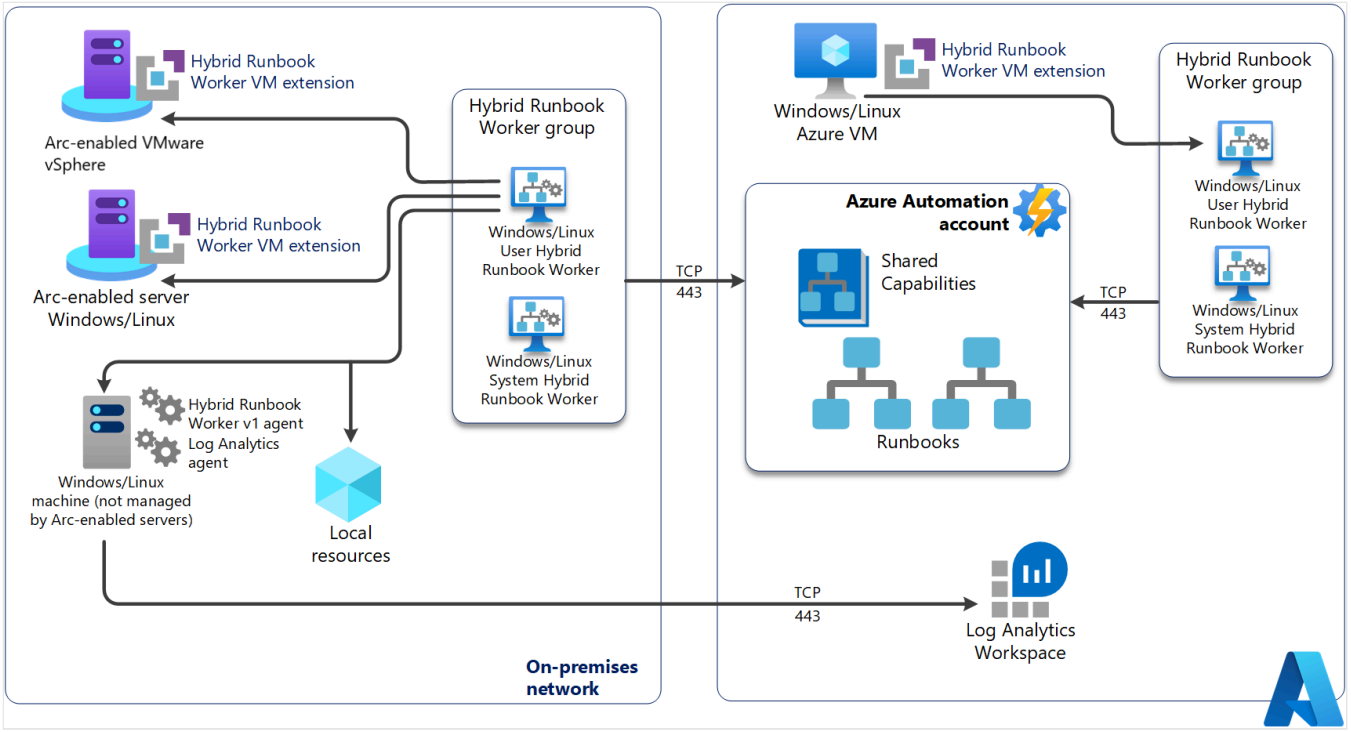
En la tabla siguiente se muestra el número máximo de instancias de Hybrid Runbook Worker de usuario y sistema en una cuenta de Automation. Si tiene que administrar más de 4000 máquinas, se recomienda crear otra cuenta de Automation.

Expandir tabla

Tipo de trabajo	Número máximo admitido por cuenta de Automation.
Sistema	4000
Usuario	4000

¿Cómo funciona?

Cada Hybrid Runbook Worker de usuario es miembro de un grupo de instancias de Hybrid Runbook Worker que especifica cuando instala el trabajo. Un grupo puede incluir solo un trabajo, pero puede incluir varios en un grupo para contar con alta disponibilidad. Cada máquina puede hospedar una instancia de Hybrid Runbook Worker que envía informes a una cuenta de Automation; no se puede registrar la instancia de Hybrid Worker en varias cuentas de Automation. Una instancia de Hybrid Worker solo puede escuchar trabajos de una única cuenta de Automation.



Un grupo de Hybrid Worker con Hybrid Runbook Workers está diseñado para lograr una alta disponibilidad y equilibrio de carga mediante la asignación de trabajos entre varios Workers. Para ejecutar runbooks correctamente, Hybrid Worker debe estar en buen estado y dar un latido. Hybrid Worker funciona en un mecanismo de sondeo para recoger trabajos. Si ninguno de los roles de trabajo del grupo Hybrid Worker ha hecho ping al servicio Automation en los últimos 30 minutos, esto indica que el grupo no tenía ningún rol de trabajo activo. En este escenario, los trabajos se suspenderán después de tres reintentos.

Cuando se inicia un runbook en una instancia de Hybrid Runbook Worker de usuario, se especifica el grupo en el que se ejecuta y no se puede especificar un rol de trabajo determinado. Cada Hybrid Worker activo del grupo sondeará los trabajos cada 30 segundos para ver si hay algún trabajo disponible. El rol de trabajo elige los trabajos siguiendo la premisa de que el primero que llega se lo queda. En función de cuándo se insertó un trabajo, la instancia de Hybrid Worker que esté dentro del grupo de Hybrid Worker que haga ping al servicio Automation primero tomará el trabajo. El tiempo de procesamiento de la cola de trabajos también depende de la carga y del perfil de hardware de Hybrid Worker.

Normalmente, una sola instancia de Hybrid Worker puede seleccionar 4 trabajos por ping (es decir, cada 30 segundos). Si la tasa de inserción de trabajos fuera superior a 4 cada 30 segundos y ningún otro Rol de trabajo tomara el trabajo, es posible que este se suspenda con un error.

Una instancia de Hybrid Runbook Worker no tiene muchos de los límites del espacio aislado de Azure en el espacio en disco, la memoria o los sockets de red. Los límites de una instancia de Hybrid Worker solo están relacionados con los propios recursos del trabajo y no están limitados por el límite de tiempo de distribución equilibrada que tienen los espacios aislados de Azure.

Para controlar la distribución de runbooks en las instancias de Hybrid Runbook Worker y cuándo o cómo se desencadenan los trabajos, puede registrar la instancia de Hybrid Worker en distintos grupos de Hybrid Runbook Worker de la cuenta de Automation. Establezca como destino los trabajos en el grupo o grupos específicos para admitir la organización de ejecuciones.

Escenarios comunes para Hybrid Runbook Worker de usuario

- Para ejecutar runbooks de Azure Automation para la administración de máquinas virtuales invitadas directamente en una máquina virtual de Azure existente y en un servidor fuera de Azure registrado como servidor habilitado para Azure Arc o máquina virtual VMware habilitada para Azure Arc (versión preliminar). Los servidores habilitados para Azure Arc pueden ser máquinas virtuales y servidores físicos de Windows y Linux hospedados fuera de Azure, en la red corporativa o en otros proveedores de nube.
- Para superar la limitación del espacio aislado de Azure Automation, los escenarios comunes incluyen la ejecución de operaciones de larga duración por encima del límite de tres horas para trabajos en la nube, la realización de operaciones de automatización que consumen muchos recursos, la interacción con servicios locales que se ejecutan de forma local o en un entorno híbrido o la ejecución de scripts que requieren permisos elevados.
- Para superar las restricciones de la organización, con el fin de mantener los datos en Azure por motivos de gobernanza y seguridad, ya que no puede ejecutar trabajos de automatización en la nube, puede ejecutarlos en una máquina local que se haya incorporado como Hybrid Runbook Worker de usuario.
- Para automatizar las operaciones en varios recursos fuera de Azure que se ejecutan en entornos locales o de varias nubes. Puede incorporar una de esas máquinas como Hybrid Runbook Worker de usuario y dirigir la automatización al resto de máquinas en el entorno local.
- Para acceder a otros servicios de forma privada desde la red virtual de Azure (VNet) sin abrir una conexión a Internet saliente, puede ejecutar runbooks en una instancia de Hybrid Worker conectada a la red virtual de Azure.

Instalación de una instancia de Hybrid Runbook Worker

Para instalarlo, consulte Implementación de una instancia de Hybrid Runbook Worker de Windows o Linux basada en extensiones en Automation.

Expandir tabla

Sistema operativo	Tipos de implementación
Windows/Linux	En el caso de las instancias de Hybrid Runbook Worker de usuario, consulte Implementación de una instancia de Hybrid Runbook Worker de usuario Windows o Linux basada en extensiones en Automation. Éste es el método recomendado.

> **Nota**
>
> Hybrid Runbook Worker no se admite actualmente en Azure Virtual Machine Scale Sets.

Planeamiento de red

Consulte Configuración de red de Azure Automation para más información sobre los puertos, las direcciones URL y otros detalles de red necesarios para Hybrid Runbook Worker.

Etiquetas de servicio

Azure Automation admite etiquetas de servicio de red virtual de Azure, empezando por la etiqueta de servicio GuestAndHybridManagement. Puede usar etiquetas de servicio para definir controles de acceso a la red en grupos de seguridad de red o Azure Firewall. Las etiquetas de servicio se pueden usar en lugar de direcciones IP específicas al crear reglas de seguridad. Al especificar el nombre de la etiqueta de servicio **GuestAndHybridManagement** en el campo de origen o destino apropiado de una regla, puede permitir o denegar el tráfico del servicio Automation. Esta etiqueta de servicio no admite un control más pormenorizado mediante la restricción de los intervalos de IP a una región específica.

La etiqueta de servicio de Azure Automation solo proporciona direcciones IP que se usan para los siguientes escenarios:

- Desencadene webhooks desde dentro de la red virtual.
- Permitir que los agentes de Hybrid Runbook Worker o State Configuration de la red virtual se comuniquen con el servicio Automation

> **Nota**
>
> La etiqueta de servicio **GuestAndHybridManagement** no admite actualmente la ejecución de trabajos de runbook en un espacio aislado de Azure, solo directamente en una instancia de Hybrid Runbook Worker.

Compatibilidad con el nivel 5 de impacto (IL5)

Hybrid Runbook Worker de Azure Automation se puede usar en Azure Government para admitir cargas de trabajo de nivel 5 de impacto en cualquiera de las dos configuraciones siguientes:

- Máquina virtual aislada. Cuando se implementa, consume todo el host físico de esa máquina, lo que proporciona el nivel necesario de aislamiento para admitir las cargas de trabajo IL5.
- Azure Dedicated Host, proporciona servidores físicos capaces de hospedar una o varias máquinas virtuales, dedicados a una suscripción de Azure.

> **Nota**
>
> El aislamiento de proceso mediante el rol Hybrid Runbook Worker está disponible para las nubes de Azure Commercial y US Government.

Runbooks en Hybrid Runbook Worker

Podría tener runbooks que administren recursos en la máquina local o se ejecuten con recursos en el entorno local en el que se implementa una instancia de Hybrid Runbook Worker de usuario. En este caso, puede elegir ejecutar los runbooks en la instancia de Hybrid Worker en lugar de en una cuenta de Automation. La estructura de los runbooks que se ejecutan en una instancia de Hybrid Runbook Worker es idéntica a la de los que ejecuta en la cuenta de Automation. Consulte Ejecución de runbooks en Hybrid Runbook Worker.

Trabajos de Hybrid Runbook Worker

Los trabajos de Hybrid Runbook Worker se ejecutan en la cuenta **del sistema** local en Windows o en la cuenta nxautomation en Linux. Azure Automation controla los trabajos en Hybrid Runbook Worker de forma diferente a los trabajos que se ejecutan en espacios aislados de Azure. Consulte Dónde ejecutar los runbooks.

Si la máquina host de Hybrid Runbook Worker se reinicia, cualquier trabajo de runbook en ejecución se reinicia desde el principio o desde el último punto de comprobación para runbooks de Flujo de trabajo de PowerShell. Si un trabajo de runbook se reinicia más de tres veces, se suspende.

Permisos del runbook para una instancia de Hybrid Runbook Worker

Puesto que acceden a recursos que no son de Azure, los runbooks que se ejecutan en una instancia de Hybrid Runbook Worker de usuario no pueden usar el mecanismo de autenticación que usan normalmente los runbooks que se autentican en los recursos de Azure. Un runbook proporciona su propia autenticación a los recursos locales o configura la autenticación mediante identidades administradas para los recursos de Azure. También puede especificar una cuenta de ejecución para proporcionar un contexto de usuario para todos los runbooks.

Pasos siguientes

- Consulte Ejecución de runbooks en Hybrid Runbook Worker para más información sobre cómo configurar los runbooks para automatizar los procesos del centro de datos local o de otros entornos de nube.
- Para aprender a solucionar problemas con las instancias de Hybrid Runbook Worker, consulte Solución de incidencias de Hybrid Runbook Worker.

Tipos de runbooks de Azure Automation

La característica de automatización de procesos de Azure Automation admite varios tipos de runbooks, tal como se define en la tabla siguiente.

Expandir tabla

Tipo	Descripción
PowerShell (Recomendaciones)	Runbook de texto basado en scripting de Windows PowerShell. Las versiones actualmente soportadas son PowerShell 7.6, PowerShell 7.4 y PowerShell 5.1. Dado que PowerShell 7.1 y PowerShell 7.2 ya no son compatibles con el producto principal PowerShell, crea libros de ejecución con versiones soportadas a largo plazo como PowerShell 7.6 o PowerShell 7.4.
Flujo de trabajo de PowerShell	Runbook de texto basado en scripting de flujo de trabajo de Windows PowerShell.
Pitón (Recomendaciones)	Runbook de texto basado en scripting de Python. La versión admitida actualmente es Python 3.10. Dado que Python 2.7 y Python 3.8 ya no son compatibles con el producto primario Python, se recomienda crear runbooks en Python 3.10.
Gráfico	Runbook gráfico basado en Windows PowerShell, y creado y editado completamente en el editor gráfico de Azure Portal.
Flujo de trabajo gráfico de PowerShell	Runbook gráfico basado en flujo de trabajo de Windows PowerShell, y creado y editado completamente en el editor gráfico de Azure Portal.

Para más información sobre el entorno de automatización de procesos, consulte Ejecución de un runbook en Azure Automation.

> **Nota:**
>
> Azure Automation seguirá el ciclo de vida de soporte técnico de las versiones del lenguaje PowerShell y Python de acuerdo con las escalas de tiempo publicadas por los productos primarios, PowerShell y Python, respectivamente. Se recomienda usar runbooks con versiones de lenguaje compatibles.

Tenga en cuenta las siguientes consideraciones al determinar qué tipo usar para un runbook concreto:

- No es posible convertir runbooks de tipo gráfico a texto o viceversa.

- Existen limitaciones al utilizar runbooks de diferentes tipos como runbooks secundarios. Para más información, consulte Child runbooks in Azure Automation (Runbooks secundarios en Azure Automation).

Runbooks de PowerShell

Los runbooks de PowerShell están basados en Windows PowerShell. Puede modificar directamente el código del runbook con el editor de texto en el Portal de Azure. También puede usar cualquier editor de texto sin conexión e importar el runbook en Azure Automation.

La versión de PowerShell viene determinada por la **versión en tiempo de ejecución** especificada.

El mismo espacio aislado de Azure y Hybrid Runbook Worker pueden ejecutar varios runbooks de **PowerShell** destinados a diferentes versiones en tiempo de ejecución en paralelo. Las versiones de ejecución de PowerShell 7.6 y PowerShell 7.4 son compatibles tanto para trabajos en la nube como para trabajos híbridos en todas las regiones.

> ⓘ **Nota:**
>
> - En el momento de la ejecución del runbook, si selecciona **Versión de tiempo de ejecución** como **7.4**, se utilizan módulos de PowerShell dirigidos a la versión 7.4, y si selecciona **Versión de tiempo de ejecución** como **5.1**, se utilizan módulos de PowerShell dirigidos a la versión 5.1.

Asegúrese de seleccionar la versión del entorno de ejecución adecuada para los módulos.

Por ejemplo: Si está ejecutando un runbook para un escenario de automatización de SharePoint en la **versión del entorno de ejecución***7.4*, importe el módulo en la **versión del entorno de ejecución7.4**; si está ejecutando un runbook para un escenario de automatización de SharePoint en la **versión del entorno de ejecución5.1**, importe el módulo en la **versión del entorno de ejecución***5.1*.

Ventajas

- Implemente toda la lógica compleja con código de PowerShell sin otras complejidades del flujo de trabajo de PowerShell.
- Se inician con más rapidez que los runbooks de flujo de trabajo de PowerShell, ya que no necesitan compilarse antes de la ejecución.
- Ejecute en Azure y en Hybrid Runbook Worker para Windows y Linux.

Limitaciones y problemas conocidos

A continuación se describen los problemas conocidos y las limitaciones actuales con los runbooks de PowerShell:

PowerShell 7.6

Limitaciones PowerShell 7.6 está disponible en experiencia de entorno de ejecución.

- En la versión de ejecución de PowerShell 7.6, las actividades de los módulos no se extraen para los módulos importados.
- PowerShell 7.x no admite flujos de trabajo. Para más información, consulte Flujo de trabajo de PowerShell.
- PowerShell 7.x no admite actualmente runbooks firmados.
- La integración de control de versiones no soporta PowerShell 7.6. Además, los runbooks de PowerShell 7.6 en control de versiones se crean en la cuenta de Automation como Runtime 5.1.
- El módulo 15.1.0 de Az está instalado por defecto. La lista completa de módulos de componentes de la versión del módulo Az seleccionada se muestra una vez que la versión de Az se configura de nuevo mediante Azure Portal o la API.
- Los módulos importados de PowerShell 7.6 se validan durante la ejecución del trabajo. Asegúrese de que todas las dependencias del módulo seleccionado también se importan para la ejecución correcta del trabajo.
- Azure Automation libros de reglas no soportan Start-Job con credencial -.
- Azure no admite todos los parámetros de entrada de PowerShell. Más información.
  Problemas conocidos
- Los runbooks que dependen de rutas de archivos internas, como `C:\modules`, pueden fallar debido a cambios en la infraestructura de fondo del servicio. Cambie el código del runbook para asegurarse de que no haya dependencias en las rutas de acceso de archivo internas y use Get-ChildItem para obtener la información necesaria del módulo.
- El cmdlet `Get-AzStorageAccount` podría producir un error: *el comando `Get-AzStorageAccount` se encontró en el módulo `Az.Storage`, pero no se pudo cargar el módulo.*

- No se admite `.\child-runbook.ps1` la ejecución de scripts secundarios. **Solución**: Utilice `Start-AutomationRunbook` (cmdlet interno) o `Start-AzAutomationRunbook` (del *módulo Az.Automation*) para iniciar otro libro de ejecución desde el libro de ejecución principal.

- Cuando se usa la versión 3.0.0 o posterior del módulo ExchangeOnlineManagement, puede experimentar errores. Para resolver el problema, asegúrese de cargar explícitamente los módulos PowerShellGet y PackageManagement.

- Cuando se usa el cmdlet `New-AzAutomationVariable` dentro del módulo de automatización de Az.Automation para cargar una variable de tipo **objeto**, la operación no funciona como se esperaba. **Solución alternativa**: convierta el objeto en una cadena JSON usando el cmdlet ConvertTo-Json y después cargue la variable con la cadena JSON como valor. Esta solución asegura el control adecuado de la variable en el entorno de Azure Automation como cadena JSON.

  Ejemplo: Crea un objeto PowerShell que almacene información sobre las máquinas virtuales de Azure

  ```
  azurepowershell

  # Retrieve Azure virtual machines with status information for the
  'northeurope' region
  $AzVM = Get-AzVM -Status | Where-Object {$_.Location -eq "northeurope"}

  $VMstopatch = @($AzVM).Id

  # Create an Azure Automation variable (This cmdlet will not fail, but the
  variable may not work as intended when used in the runbook.)
  New-AzAutomationVariable -ResourceGroupName "mrg" -AutomationAccountName
  "mAutomationAccount2" -Name "complex1" -Encrypted $false -Value $VMstopatch

  # Convert the object to a JSON string
  $jsonString = $VMstopatch | ConvertTo-Json

  # Create an Azure Automation variable with a JSON string value (works
  effectively within the automation runbook)
  New-AzAutomationVariable -ResourceGroupName "mrg" -AutomationAccountName
  "mAutomationAccount2" -Name "complex1" -Encrypted $false -Value $jsonString
  ```

Runbooks del flujo de trabajo de PowerShell

Los runbooks de PowerShell Workflow son runbooks de texto basados en Windows PowerShell Workflow. Puede modificar directamente el código del runbook con el editor de texto en el Portal de Azure. También puede usar cualquier editor de texto sin conexión e importar el runbook en Azure Automation.

> **Nota:**
>
> PowerShell 7.1 (versión preliminar) y PowerShell 7.2 no admiten runbooks de flujo de trabajo.

Ventajas

- Implemente toda la lógica compleja con código del flujo de trabajo de PowerShell.
- Use los puntos de control para reanudar la operación si se produce un error.
- Utilice el procesamiento paralelo para realizar varias acciones en paralelo.
- Se pueden incluir otros runbooks gráficos o de flujo de trabajo de PowerShell como runbooks secundarios para crear flujos de trabajo de alto nivel.

Limitaciones

- El flujo de trabajo de PowerShell no se admite en versiones posteriores de PowerShell 7+. Por lo tanto, los manuales operativos obsoletos no pueden ser mejorados.
- Control ineficaz de la ejecución en paralelo en comparación con las versiones más recientes de PowerShell 7+.
- El flujo de trabajo de PowerShell funciona internamente con varios procesos. Por lo tanto, es posible que los módulos disponibles en un proceso no estén disponibles en otro y provoquen excepciones como *comando no encontrado*.
- Los runbooks deben tratar la complejidad adicional del flujo de trabajo de PowerShell como objetos deserializados.
- Los runbooks tardan más en iniciarse que los runbooks de PowerShell, ya que deben compilarse antes de su ejecución.
- Solo puede incluir runbooks de PowerShell como runbooks secundarios mediante el `Start-AzAutomationRunbook` cmdlet.
- Los runbooks no se pueden ejecutar en una instancia de Hybrid Runbook Worker de Linux.

Runbooks de Python

Los runbooks de Python se compilan en Python 3.10. Puede modificar directamente el código del runbook usando el editor de texto en el portal de Azure. También puede usar cualquier editor de texto sin conexión e importar el runbook en Azure Automation. El producto primario ya no admite Python 2.7 y Python 3.8 y se recomienda crear runbooks en la versión en tiempo de ejecución de Python 3.10.

La versión de ejecución de Python 3.10 es compatible tanto para trabajos en la nube como para trabajos híbridos en todas las regiones.

Ventajas

> **Nota:**
>
> La importación de un paquete de Python puede tardar varios minutos.

- Usa las sólidas bibliotecas de Python.
- Se pueden ejecutar en Azure o en instancias de Hybrid Runbook Worker.
- Los scripts y paquetes de cualquier versión 3.x podrían funcionar si el código es compatible con distintas versiones.
- Para trabajos híbridos de Python 3.10 en máquinas Windows, puede optar por instalar cualquier versión 3.x que quiera usar.
- Para trabajos híbridos de Python 3.10 en máquinas Linux, dependemos de la versión de Python 3 instalada en la máquina para ejecutar DSC OMSConfig y Linux Hybrid Worker. Las distintas versiones deberían funcionar si no hay cambios importantes en los contratos o firmas de método entre las versiones de Python 3.

Limitaciones

Las limitaciones de los runbooks de Python son:

Python 3.10

- En el caso de los módulos de Python 3.10, actualmente solo se admiten los archivos wheel destinados al sistema operativo Linux cp310. Más información
- No se admite la integración del control de código fuente.
- Los paquetes personalizados para Python 3.10 solo se validan durante el tiempo de ejecución del trabajo. Se espera que se produzca un error en el trabajo si el paquete no

es compatible en el runtime o si las dependencias necesarias de los paquetes no se importan en la cuenta de Automation.
- Actualmente, los runbooks de Python 3.10 solo se admiten desde Azure Portal y la API rest.

> ⓘ **Nota:**
>
> No se admite el uso de un webhook para iniciar un runbook de Python.

Varias versiones de Python

Es aplicable para trabajadores híbridos de Windows. En el caso de Runbook Worker de Windows, al ejecutar un runbook de Python 2, este busca primero la variable de entorno `PYTHON_2_PATH` y valida si apunta a un archivo ejecutable válido. Por ejemplo, si la carpeta de instalación es `C:\Python2`, comprobaría si `C:\Python2\python.exe` es una ruta de acceso válida. Si no se encuentra, busca la variable de entorno `PATH` para realizar una comprobación similar.

Para Python 3, primero busca la variable env `PYTHON_3_PATH` y, a continuación, recurre a la variable de entorno `PATH`.

Cuando se usa solo una versión de Python, puede agregar la ruta de instalación a la variable `PATH`. Si quiere usar ambas versiones en Runbook Worker, establezca `PYTHON_2_PATH` y `PYTHON_3_PATH` en la ubicación del módulo para esas versiones.

Problemas conocidos

En el caso de los trabajos en la nube, a veces se produce un error en los trabajos de Python 3.8 con un mensaje de excepción `invalid interpreter executable path`. Podría ver esta excepción si el trabajo se retrasa y empieza más de 10 minutos tarde o usa **Start-AutomationRunbook** para iniciar runbooks de Python 3.8. Si el trabajo se retrasa, reiniciar el runbook debería ser suficiente.

Runbooks gráficos

Puede crear y editar runbooks gráficos y runbooks gráficos de flujo de trabajo de PowerShell mediante el editor gráfico de Azure Portal. Sin embargo, no puede crear o editar este tipo de runbooks con otra herramienta. Principales características de los runbooks gráficos:

- Se exportan a archivos de la cuenta de Automation y se importan a otra cuenta de Automation.
- Generar código de PowerShell.
- Se pueden convertir a y desde runbooks gráficos de flujo de trabajo de PowerShell durante la importación.

Ventajas

- Utilice modelos de creación visual para insertar, vincular y configurar.
- Céntrese en cómo fluyen los datos por el proceso.
- Represente visualmente los procesos de administración.
- Incluya otros runbooks como runbooks secundarios para crear flujos de trabajo de nivel alto.
- Anime a usar la programación modular.

Limitaciones

- No puede crear o editar fuera de Azure Portal.
- Puede requerir una actividad de código que contenga código de PowerShell para ejecutar una lógica compleja.
- No puede convertir a uno de los formatos de texto, ni puede convertir un runbook de texto a un formato gráfico.
- No puede ver ni modificar directamente el código de PowerShell que el flujo de trabajo gráfico crea. Puede ver el código que crea en cualquier actividad de código.
- No puede ejecutar runbooks en Linux Runbook Worker. Consulte Automatización de recursos en los centros de datos o nube con Hybrid Runbook Worker.
- Los runbooks gráficos no se pueden firmar digitalmente.

Pasos siguientes

- Para aprender sobre los runbooks de PowerShell, consulte Tutorial: Creación de un runbook de PowerShell.
- Para aprender sobre los runbooks de flujo de trabajo de PowerShell, consulte Tutorial: Creación de un runbook de flujo de trabajo de PowerShell.
- Para aprender sobre los runbooks gráficos, consulte Tutorial: Creación de un runbook gráfico.
- Para aprender sobre los runbooks de Python, consulte Tutorial: Creación de un runbook de Python.

- Para obtener información sobre la compatibilidad en tiempo de ejecución y la retirada al trabajar con runbooks, consulta Política de compatibilidad y retirada de entornos de ejecución de lenguajes para Azure Automation.

Last updated on 13/08/2026

Entorno de ejecución de Azure Automation

En este artículo se proporciona información general sobre el entorno en runtime, el ámbito y sus funcionalidades.

Acerca del entorno en runtime

El **entorno en runtime** le permite configurar el entorno de ejecución del trabajo y proporciona la flexibilidad de elegir el lenguaje en runtime y la versión en runtime según sus requisitos. Es el único origen de la verdad para definir y administrar el entorno en el que se ejecuta un trabajo. Cada runbook tiene dos componentes:

- Código de script
- Entorno en tiempo de ejecución: define el lenguaje en tiempo de ejecución, la versión en tiempo de ejecución y los paquetes necesarios durante la ejecución del trabajo.

Puede cambiar estos componentes de forma independiente sin afectar al otro.

> ⓘ **Nota:**
>
> Los entornos de ejecución están soportados en todas las nubes públicas y gubernamentales. Puede asociar cada runbook a un único entorno en runtime. Sin embargo, un entorno en runtime podría estar vinculado a varios runbooks.

Componentes del entorno en runtime

El entorno en runtime captura los detalles siguientes sobre el entorno de ejecución del trabajo:

- **Lenguaje**: el lenguaje de scripts destinado a la ejecución del runbook. Por ejemplo, PowerShell y Python.

- **Versión en tiempo de ejecución** : la versión del lenguaje seleccionado para la ejecución del runbook. Por ejemplo, PowerShell 7.4 y Python 3.10.

- **Paquetes** : los paquetes son los ensamblados y los archivos *.dll* que se importan y requieren los runbooks para su ejecución. Hay dos tipos de paquetes admitidos para el entorno en runtime.

Expandir tabla

Tipos de paquete	Descripción
Paquetes predeterminados	Los paquetes permiten administrar los recursos de Azure. Por ejemplo, Az PowerShell 12.3.0, CLI de Azure 2.64.0
Paquetes proporcionados por el cliente	Estos son paquetes personalizados que requieren los runbooks durante la ejecución. Los paquetes pueden ser de: : Galería pública: PSGallery, pypi : Autogenerado

Paquete de la CLI de Azure en entorno en runtime

Los comandos de la CLI de Azure se admiten en runbooks asociados con el entorno en tiempo de ejecución de PowerShell 7.4. **La VERSIÓN 2.64.0** de la CLI de Azure está disponible como un paquete predeterminado en el entorno en tiempo de ejecución de PowerShell 7.4. Azure Automation sigue estrechamente la cadencia de versión de las versiones más recientes de la CLI de Azure y las admite en runbooks.

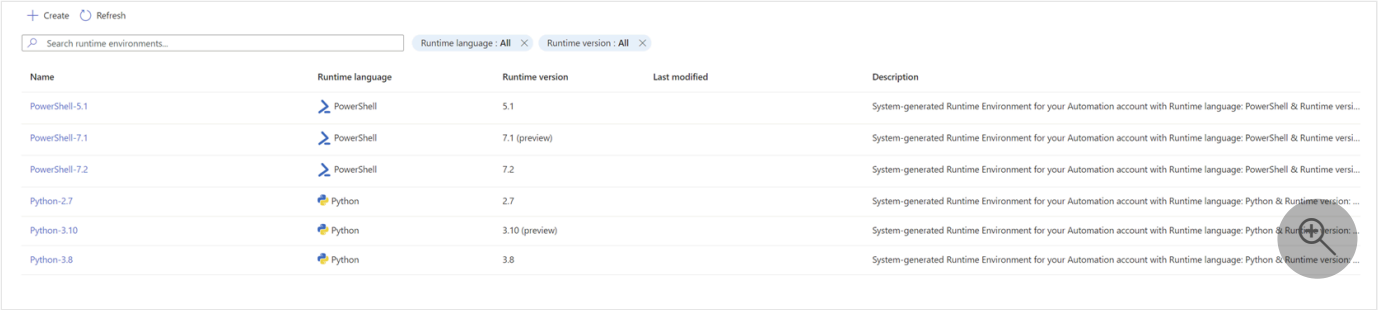
Los runbooks vinculados al entorno en tiempo de ejecución de PowerShell 7.4 siempre se ejecutarían con la versión más reciente de la CLI de Azure compatible con Azure Automation. Del mismo modo, las versiones declaradas de finalización de soporte técnico por parte de la CLI principal de Azure ya no serían compatibles con Azure Automation, ya que podrían sufrir errores o vulnerabilidades de seguridad. Asegúrese de que los runbooks están diseñados para ejecutarse sin problemas en versiones más recientes de la CLI de Azure.

Entornos en runtime generados por el sistema

Azure Automation crea entornos en tiempo de ejecución generados por el sistema en función del lenguaje en tiempo de ejecución, la versión y los paquetes o módulos presentes en la interfaz antigua de la cuenta de Azure Automation. Hay seis entornos en runtime generados por el sistema:

- PowerShell-5.1
- PowerShell-7.1
- PowerShell-7.2
- Python-2.7
- Python-3.8
- Python-3.10

No se pueden editar estos entornos en runtime. Sin embargo, los cambios realizados en módulos o paquetes de la cuenta de Automation se reflejan automáticamente en estos entornos en runtime generados por el sistema.



> ⓘ **Nota:**
>
> - Los paquetes presentes en entornos en runtime generados por el sistema son únicos para la cuenta de Azure Automation y pueden variar entre diferentes cuentas.
> - El entorno en tiempo de ejecución generado por el sistema no está disponible para PowerShell 7.4 o posterior.

Ventajas principales

- **Control granular** : permite configurar el entorno de ejecución de scripts eligiendo el lenguaje runtime, su versión y los módulos dependientes.
- **Actualización de runbooks**: permite la sencilla portabilidad de los runbooks entre diferentes versiones de tiempo de ejecución actualizando su entorno de ejecución para estar al día con las versiones más recientes de PowerShell y Python. Puede probar las actualizaciones antes de publicarlas en producción.
- **Administración de** módulos: permite probar la compatibilidad durante las actualizaciones del módulo y evitar cambios inesperados que podrían afectar a la ejecución de sus escenarios de producción.
- **Funcionalidad de reversión** – Permite revertir fácilmente el runbook a un entorno de ejecución anterior. En caso de que una actualización de runbook presente problemas o un comportamiento inesperado.
- **Código simplificado** : permite organizar fácilmente el código mediante la vinculación de runbooks a distintos entornos en tiempo de ejecución sin necesidad de crear varias cuentas de Automation.

Limitaciones

- Los runbooks de flujo de trabajo de PowerShell gráfico, PowerShell gráfico y flujo de trabajo gráfico de PowerShell solo funcionan con el entorno de runtime de PowerShell-5.1 generado por el sistema.
- Los runbooks creados en la experiencia del entorno en runtime con la versión en runtime de PowerShell 7.2+ se mostrarían como runbooks de PowerShell 5.1 en la experiencia anterior.
- Los permisos RBAC no se pueden asignar al entorno en runtime.
- El entorno en runtime no se puede configurar a través de la extensión de Azure Automation para Visual Studio Code.
- No se pueden recuperar los entornos en runtime eliminados.
- Esta característica se admite actualmente a través de Azure Portal y la API REST.
- Utiliza Paquetes: API REST para cargar paquetes personalizados.
- La administración de módulos para State Configuration de Azure Automation no se admite a través de la experiencia del entorno en runtime. Puede seguir usando la experiencia anterior para administrar módulos y paquetes para State Configuration de Azure Automation.

Cambiar entre la experiencia nueva y antigua

Aunque se recomienda la nueva experiencia de entorno en runtime, también puede cambiar a la experiencia predeterminada en cualquier momento. Obtenga más información sobre cómo alternar entre las dos experiencias.

> **Nota:**
>
> Las actualizaciones del runbook persisten entre la nueva experiencia del entorno en runtime y la experiencia anterior. Los cambios realizados en el entorno en tiempo de ejecución vinculados a un runbook se conservarían durante la ejecución del runbook en la experiencia anterior.

Pasos siguientes

- Para trabajar con runbooks y entorno en runtime, consulte Administración del entorno en runtime.
- Para obtener información sobre la compatibilidad del entorno de ejecución y su retirada, consulte Directiva de compatibilidad y retirada del entorno de ejecución de lenguaje para Azure Automation.
- Para más información sobre PowerShell, consulte PowerShell Docs.

Last updated on 13/08/2026

Límites y cuotas de Azure Automation

En este artículo, se proporciona información general sobre las cuotas o límites predeterminados que se ofrecen a distintos recursos de Azure Automation.

Información general

Azure Automation ofrece límites y cuotas a los recursos asignados para la suscripción de Azure. Estos límites y cuotas suelen variar para distintos tipos de suscripciones. Se asigna un valor predeterminado para cada recurso de una cuenta de Automation para cada suscripción. Hay algunos recursos para los que puede solicitar un aumento de la cuota y, para otros recursos, el valor predeterminado es fijo y no se puede cambiar. Puede solicitar aumentos de cuota solo para los recursos para los que se menciona *solicitud de aumento de cuota*. Para todos los demás recursos, el valor predeterminado es un límite estricto (inmutable) y no se consideraría la solicitud de aumento.

> **Nota:**
>
> Los aumentos de cuota están sujetos a disponibilidad de recursos en la región seleccionada.

Límites de servicio y suscripción

Expandir tabla

Recurso	Límite	Notas
Número máximo de cuentas de Automation activas en una suscripción en una región	10	Las suscripciones Enterprise y CSP podrán crear cuentas de Automation en cualquiera de las regiones públicas admitidas por el servicio. Cree una solicitud de soporte técnico para solicitar el aumento de cuota. Más información.
	2	Las suscripciones de pago por uso, de patrocinio, MSDN, MPN y Azure Pass pueden crear cuentas de Automation en cualquiera de las regiones admitidas por el servicio. Cree una solicitud de soporte técnico para solicitar el aumento de cuota. Más información.

Recurso	Límite	Notas
	1	La Prueba gratuita y las suscripciones de Azure for Student y Azure bajo licencia Open solo pueden crear una cuenta de Automation por región por suscripción. Lista de regiones permitidas: EastUS, EastUS2, WestUS, NorthEurope, SoutheastAsia y JapanWest2 [2]
Número máximo de trabajos en ejecución simultáneos en la misma instancia de tiempo por cuenta de Automation por región	50	Si se alcanza este límite, se producirá un error en las siguientes solicitudes para crear un trabajo. El cliente recibe una respuesta de error. Suscripciones empresariales y de CSP en regiones públicas. Cree una solicitud de soporte técnico para solicitar el aumento de cuota. Más información.
	10	Suscripciones de pago por uso, patrocinadas, MSDN, MPN y Azure Pass en regiones públicas. Cree una Solicitud de soporte técnico para solicitar una cuota. Más información.
	5	Prueba gratuita, suscripciones de Azure for Student y Azure bajo licencia Open en regiones públicas [2].
Cantidad máxima de trabajos nuevos que se puede enviar cada 30 segundos por cuenta de Azure Automation	100	Si se alcanza este límite, se producirá un error en las siguientes solicitudes para crear un trabajo. El cliente recibe una respuesta de error.
Tamaño máximo de almacenamiento de metadatos de trabajo para un período sucesivo de 30 días	10 GB (aproximadamente, 4 millones de trabajos)	Si se alcanza este límite, se producirá un error en las siguientes solicitudes para crear un trabajo.
Límite de flujo de trabajo máximo	1 MiB	Una sola secuencia no puede ser mayor de 1 MiB.
Límite máximo de flujo de trabajo en el portal de Azure Automation	200 KB	Límite del portal para mostrar los registros de tareas.
Cantidad máxima de módulos que se puede importar cada 30 segundos por cuenta de Automation	5	
Tamaño máximo de un módulo	100 MB	

Recurso	Límite	Notas
Tamaño máximo de un archivo de configuración de nodo	1 MB	Se aplica a la configuración de estado
Tiempo de ejecución del trabajo, nivel gratuito	500 minutos por suscripción por mes del calendario	
Cantidad máxima de espacio en disco permitida para cada sandbox [1]	1 GB	Se aplica solo a los entornos limitados de Azure.
Cantidad máxima de memoria que se asigna a un espacio aislado [1]	400 MB	Se aplica solo a los entornos aislados de Azure.
Número máximo de sockets de red permitidos por entorno aislado [1]	1,000	Se aplica únicamente a los Sandboxes de Azure
Tiempo máximo de ejecución permitido para cada manual de procedimientos [1]	Tres horas	Se aplica solo a Azure Sandboxes
Número máximo de runbooks por Cuenta de automatización	800	
Número máximo de instancias de trabajador híbrido de libro de ejecución del sistema por cuenta de Automation	4\.000	
Número máximo de instancias de trabajador híbrido de libro de ejecución del usuario por cuenta de Automation	4\.000	
Número máximo de trabajos simultáneos que se pueden ejecutar en un único Hybrid Runbook Worker	50	
Tamaño máximo de los parámetros de un trabajo de runbook	512 kilobytes	
Número máximo de parámetros del manual de procedimientos	50	Si alcanza el límite de 50 parámetros, puede pasar una cadena JSON o XML a un parámetro y analizarlo con el runbook.

Recurso	Límite	Notas
Tamaño máximo de carga útil de webhook	512 kilobytes	
Máximo de días que se conservan los datos de trabajo	30 días	
Tamaño máximo del estado de flujo de trabajo de PowerShell	5 MB	Se aplica a los runbooks del flujo de trabajo de PowerShell al aplicar el punto de control del flujo de trabajo.
Número máximo de etiquetas que admite una cuenta de Automation	15	
Número máximo de caracteres en el campo de valor de una variable	1 048 576	

[1]Un espacio aislado es un entorno compartido que puede usarse en varios trabajos. Los trabajos que usan el mismo espacio aislado están sujetos a las limitaciones de recursos del espacio aislado.
[2]Las suscripciones gratuitas, incluidas las de cuenta gratuita de Azure y Azure for Students no son aptas para cambios de límite o cuota. Si tiene una suscripción gratuita, puede pasar a una suscripción de pago por uso.
[3]Límites para nubes gubernamentales: 200 trabajos simultáneos en ejecución al mismo tiempo por cada cuenta de Automation, sin límite en el número de cuentas de Automation por suscripción.

Preguntas más frecuentes

¿Qué límites de recursos se están revisando?

Los límites siguientes se están revisando desde el 7 de enero de 2025 en todas las regiones públicas admitidas por el servicio:

- Número máximo de cuentas de Automation activas en una suscripción en una región.
- Número máximo de tareas en ejecución simultánea en un mismo momento por cuenta de Automation.

Las implementaciones están en curso y se aprecia su paciencia durante el período de transición.

¿Cómo puedo comprobar el uso y los límites actuales?

Puede comprobar el uso y los límites actuales de las cuentas de Automation y los trabajos que se ejecutan simultáneamente al crear una solicitud de soporte técnico en la categoría *Límites de servicio y suscripción (cuotas) -> Azure Automation*. Más información

Mi uso actual es mayor que los límites revisados. ¿Cuál debo hacer?

Tenga la seguridad de que el uso actual de ambos recursos (cuentas de Automation y trabajos en ejecución simultáneos) se respeta y no se verá afectado debido a cambios en los límites. Por ejemplo, si es un cliente Enterprise y el nuevo límite es de 10 cuentas de Automation y el uso actual es de 12 cuentas. Aunque el uso sea mayor que el nuevo límite, se respetará el uso de 12 cuentas y, a continuación, se considerará como el nuevo límite. Cuando se supere el nuevo límite de 12 cuentas, se producirá un error.

Necesito más recursos que mis límites actuales. ¿Cuál debo hacer?

Puede solicitar el aumento y la disminución de la cuota en función de los cambios en sus requisitos. Se recomienda planear el aumento de los recursos de antemano y solicitar un aumento de cuota mediante la creación de una solicitud de soporte técnico en la categoría *Límites de servicio y suscripción (cuotas) -> Azure Automation*. Más información

¿Cómo afectan las solicitudes de cuota a la facturación de mi suscripción?

Las solicitudes de soporte técnico para los cambios de cuota no tienen ningún impacto adicional en la facturación de la suscripción. Puede usar el plan de soporte técnico existente para solicitar cambios de cuota.

No puedo solicitar cambios de cuota. ¿Cuál debo hacer?

Asegúrese de que dispone de una cuenta activa de Automation y acceso como Colaborador para solicitar cambios en la cuota.

Pasos siguientes

Si su pregunta no se ha respondido aquí, consulte las fuentes siguientes para preguntas y respuestas adicionales.

- Foro de comentarios
- Obtenga más información sobre cómo ver los límites actuales y solicitar aumento o reducción de cuotas.

Last updated on 26/05/2026

Información general sobre la configuración de estado deseado de Microsoft

Desired State Configuration (DSC) de Microsoft es una plataforma de configuración declarativa. Con DSC, el estado de una máquina se describe mediante un formato que debe ser claro para entender aunque el lector no sea un experto en la materia. A diferencia de las herramientas imperativas, con DSC la definición de un entorno de aplicación es independiente de la lógica de programación que impone esa definición.

La aplicación de línea de comandos DSC (`dsc`) abstrae la gestión de los componentes de software de forma declarativa e idempotente. DSC se ejecuta en Linux, macOS y Windows sin dependencias externas.

Con DSC, puede:

- Cree recursos de DSC para administrar los sistemas en cualquier idioma.
- Invoque recursos individuales directamente.
- Cree documentos de configuración que definan el estado deseado de un sistema.

Documentos de configuración

Los documentos de configuración de DSC son archivos de datos declarativos que definen instancias de recursos. Normalmente, los documentos de configuración definen qué estado se va a aplicar. DSC admite la escritura de documentos de configuración tanto en JSON como en YAML.

Los escenarios de ejemplo incluyen requisitos para un entorno de aplicación o estándares operativos o de seguridad.

Recursos de DSC

Los recursos de DSC definen cómo administrar el estado de un componente de aplicación o sistema determinado. Los recursos describen un esquema para la configuración administrable del componente. Cada recurso se puede usar con las operaciones de **Obtener** y **Test** para recuperar el estado actual de una instancia de recurso y validar si está en el estado deseado. La mayoría de los recursos también admiten la aplicación del estado deseado con la operación Set .

Algunos escenarios de ejemplo son:

- Cómo actualizar el contenido de un archivo.

- Cómo ejecutar una utilidad que cambia el estado de una máquina.
- Cómo configurar los ajustes de una aplicación.

Diferencias con DSC de PowerShell

DSC difiere de la configuración de estado deseado de PowerShell (PSDSC) en algunos aspectos importantes:

- DSC no *depende* de PowerShell, Windows PowerShell ni del módulo de PowerShell PSDesiredStateConfiguration . DSC proporciona compatibilidad total con los recursos PSDSC a través de los recursos y `Microsoft.Windows/WindowsPowerShell` *adaptador* `Microsoft.DSC/PowerShell`.

  Con el recurso de `Microsoft.DSC/PowerShell` adaptador, puede usar cualquier recurso PSDSC implementado como una clase de PowerShell. El recurso controla la detección, validación e invocación de recursos PSDSC en PowerShell. El recurso se incluye en el paquete de instalación de DSC para todas las plataformas.

  Con el recurso de `Microsoft.Windows/WindowsPowerShell` adaptador, puede usar cualquier recurso PSDSC compatible con Windows PowerShell. El recurso controla la detección, validación e invocación de recursos PSDSC en Windows PowerShell. El recurso se incluye en los paquetes de instalación de DSC solo para Windows.

- Dado que DSC no depende de PowerShell, puede usar DSC sin PowerShell instalado y administrar recursos escritos en bash, Python, C#, Rust o cualquier otro lenguaje.

- DSC no incluye un administrador de configuración local. DSC se invoca como un comando. No se ejecuta como servicio.

- Los nuevos recursos de DSC definen sus esquemas con archivos JSON o YAML, no con archivos MOF. Los recursos autónomos definen un *manifiesto de recursos* que indica cómo DSC debe invocar el recurso y qué propiedades puede administrar el recurso. En el caso de los recursos adaptados, como los implementados en PowerShell, el recurso del adaptador indica a DSC cuáles son las propiedades disponibles para el recurso y controla la invocación de los recursos adaptados.

- Los documentos de configuración se definen en archivos JSON o YAML, no en archivos de script de PowerShell. Los documentos de configuración admiten un subconjunto de funcionalidades en las plantillas de ARM, incluidos parámetros, variables, metadatos y funciones de expresión para resolver dinámicamente los datos de la configuración.

Instalación

Instalar DSC manualmente

Para instalar DSC en cualquier plataforma:

1. Descargue la versión más reciente de del repositorio de PowerShell/DSC.
2. Expanda el archivo de versión.
3. Agregue la carpeta que contiene el contenido de archivo expandido al `PATH`.

Instalar DSC en Windows con WinGet

Los siguientes comandos se pueden usar para instalar DSC mediante los paquetes publicados `winget` de Microsoft Store:

Busque la última versión de DSC:

PowerShell

```
winget search DesiredStateConfiguration --source msstore
```

Output

```
Name                              Id           Version Source
-----------------------------------------------------------------
DesiredStateConfiguration         9NVTPZWRC6KQ Unknown msstore
DesiredStateConfiguration-Preview 9PCX3HX4HZ0Z Unknown msstore
```

Instale DSC usando el `id` parámetro:

PowerShell

```
# Install latest stable
winget install --id 9NVTPZWRC6KQ --source msstore
```

PowerShell

```
# Install latest preview
winget install --id 9PCX3HX4HZ0Z --source msstore
```

Integración con DSC

DSC es una herramienta de plataforma que abstrae las preocupaciones para definir e invocar recursos. Las herramientas de orden superior, como WinGet, Microsoft Dev Box y Azure Machine Configuration , son los primeros socios de DSC como agentes de orquestación.

DSC utiliza esquemas JSON para definir la estructura de los recursos, los documentos de configuración y las salidas que devuelve DSC. Estos esquemas facilitan la integración de DSC con otras herramientas, ya que estandarizan y documentan cómo interactuar con DSC.

Para obtener más información, consulte Información general sobre la referencia del esquema JSON de DSC.

Consulte también

- Anatomía de un recurso de DSC basado en comandos para obtener información sobre cómo crear un recurso en el lenguaje que prefiera.
- referencia de línea de comandos para el comando "dsc"
- Información general sobre la referencia del esquema JSON de DSC
- Configuración de WinGet

Last updated on 09/06/2025

Detalles de configuración de la red de Azure Automation

Artículo • 18/12/2024

En esta página se proporcionan los detalles de red necesarios para Hybrid Runbook Worker y State Configuration y para Update Management y Seguimiento de cambios e inventario.

Hybrid Runbook Worker y State Configuration

El siguiente puerto y las direcciones URL son necesarios para Hybrid Runbook Worker, y para State Configuration de Automation a fin de comunicarse con Azure Automation.

- Puerto: solo se requiere el puerto 443 para el acceso saliente a Internet
- Dirección URL global: `*.azure-automation.net`
- Dirección URL global de US Gov Virginia: `*.azure-automation.us`
- Servicio del agente: `https://<workspaceId>.agentsvc.azure-automation.net`

Planeación de la red para Hybrid Runbook Worker

Si usa un firewall para restringir el acceso a Internet, tendrá que configurarlo para que permita el acceso. El siguiente puerto y las direcciones URL son necesarios para Hybrid Runbook Worker, y para State Configuration de Automation a fin de comunicarse con Azure Automation.

Expandir tabla

Propiedad	Descripción
Port	443 para el acceso a Internet de salida
Dirección URL global	*.azure-automation.net
Dirección URL global de US Gov Virginia	*.azure-automation.us

Si tiene una cuenta de Automation definida para una región específica, puede restringir la comunicación de Hybrid Runbook Worker con ese centro de datos regional. Revise los registros DNS necesarios en los registros DNS usados por Azure Automation.

Configuración de redes privadas para State Configuration

Si los nodos se encuentran en una red privada, se necesitan el puerto y las direcciones URL definidas anteriormente. Estos recursos proporcionan conectividad de red para que el nodo administrado y el DSC permitido se comuniquen con Azure Automation.

Si usa recursos de DSC que se comunican entre nodos, como los recursos WaitFor, también debe permitir el tráfico entre nodos. Consulte la documentación de cada recurso de DSC para comprender estos requisitos de red.

Para comprender los requisitos de cliente para TLS 1.2 o superior, consulte TLS 1.2 o superior para Azure Automation.

Update Management y Seguimiento de cambios e inventario

Las direcciones de esta tabla son necesarias tanto para Update Management como para Seguimiento de cambios e inventario. El párrafo que sigue a la tabla también se aplica a ambos servicios.

La comunicación con estas direcciones usa el **puerto 443**.

Expandir tabla

Azure público	Azure Government
*.ods.opinsights.azure.com	*.ods.opinsights.azure.us
*.oms.opinsights.azure.com	*.oms.opinsights.azure.us
*.blob.core.windows.net	*.blob.core.usgovcloudapi.net
*.azure-automation.net	*.azure-automation.us

Al crear reglas de seguridad de grupo de red o configurar Azure Firewall para permitir el tráfico en el servicio Automation y en el área de trabajo de Log Analytics, utilice las etiquetas de servicio **GuestAndHybridManagement** y **AzureMonitor**. De esta forma, se simplifica la administración continua de las reglas de seguridad de red. Para conectarse al servicio Automation desde las máquinas virtuales de Azure de forma segura y privada, revise Uso de Azure Private Link. Para obtener información de intervalos y la etiqueta de servicio actual para incluirla como parte de las configuraciones del firewall local, consulte Archivos JSON descargables.

Pasos siguientes

- Obtenga información sobre la Introducción a Update Management de Azure Automation.
- Obtenga información acerca de Hybrid Runbook Worker.
- Obtenga información acerca de Seguimiento de cambios e inventario.
- Obtenga información acerca de State Configuration de Automation.

Comentarios

¿Le ha resultado útil esta página? Sí No

Proporcionar comentarios sobre el producto | Obtener ayuda en Microsoft Q&A

Creación de runbooks mediante VS Code en Azure Automation

En este artículo se explica la extensión de Visual Studio que puede usar para crear y administrar runbooks.

Azure Automation proporciona una nueva extensión de VS Code para crear y administrar runbooks. Con esta extensión, puede realizar todas las operaciones de administración de runbook, como, crear y editar runbooks, desencadenar un trabajo, realizar un seguimiento de la salida de trabajos recientes, vincular una programación, la administración de recursos y la depuración local.

Requisitos previos

- Una cuenta de Azure con una suscripción activa. Cree una cuenta gratuita.
- Código de Visual Studio.
- Los módulos de PowerShell y los paquetes de Python usados por runbook deben instalarse localmente en la máquina para ejecutar el runbook localmente.

Sistemas operativos admitidos

La matriz de pruebas incluye los siguientes sistemas operativos:

1. **Windows Server 2022** con Windows PowerShell 5.1 y PowerShell Core 7.2.7
2. **Windows Server 2019** con Windows PowerShell 5.1 y PowerShell Core 7.2.7
3. **macOS 11** con PowerShell Core 7.2.7
4. **Ubuntu** 20.04 con PowerShell Core 7.2.7

> **Nota:**
>
> - La extensión debe funcionar en cualquier lugar de VS Code y admite PowerShell 7.2 o posterior. Para Windows PowerShell, solo se admite la versión 5.1.
> - PowerShell Core 6 es de fin de ciclo de vida y no se admite.

Principales características

- **Incorporación simplificada**: puede iniciar sesión con una cuenta de Azure de una manera sencilla y segura.

- **Varios lenguajes** : admite todas las pilas en tiempo de ejecución de Automation, como PowerShell 5, PowerShell 7, Python 2 y Python 3 Runbooks.
- **Compatibilidad**: admite la ejecución de pruebas del trabajo, la publicación de trabajos de Automation y el desencadenamiento de trabajos en Azure y Hybrid Worker. Puede ejecutar runbooks localmente.
- Admite parámetros posicionales de Python y parámetros de PowerShell para desencadenar el trabajo.
- **Webhooks simplificado**: puede crear un webhook, iniciar un trabajo a través de un webhook de una manera más sencilla. Además, compatibilidad para vincular una programación a un Runbook.
- **Administrar recursos de Automation**: puede realizar operaciones de creación, actualización y eliminación en recursos, incluidos certificados, variables, credenciales y conexiones.
- **Ver propiedades**: puede ver las propiedades y seleccionar Grupo de Hybrid Worker para ejecutar trabajos híbridos y ver los últimos 10 trabajos ejecutados.
- **Depuración local:** puede depurar los scripts de PowerShell localmente.
- **Comparación de runbook**: puede comparar el runbook local con el runbook publicado o el borrador de la copia del runbook.

Características clave de v1.0.8

- **Opciones de configuración del directorio local**: puede definir el directorio de trabajo que desea guardar runbooks localmente.
  - **Cambiar directorio: ruta de acceso base**: se usa la ruta de acceso del directorio modificada al volver a abrir el IDE de Visual Studio Code. Para cambiar el directorio mediante la paleta de comandos, use **Ctrl+Mayús+P:> seleccione Cambiar directorio**. Para cambiar la ruta de acceso base desde los valores de configuración de la extensión, seleccione el icono **Administrar** en la barra de actividades de la izquierda y vaya a **Configuraciones** > **Extensiones** > **Azure Automation** >: **Directorio: Ruta de acceso base**.
  - **Cambiar directorio: estructura de carpetas**: puede cambiar la estructura de carpetas del directorio local de *vscodeAutomation/accHash* a *subscription/resourceGroup/automationAccount*. Seleccione el icono **Administrar** de la barra de actividades de la izquierda y vaya a **Extensiones**> **Configuración**> **Azure Automation**> **Directorio: estructura de carpetas**. Puede cambiar la configuración predeterminada *vscodeAutomation/accHash* a formato *subscription/resourceGroupe/automationAccount*.

    ⓘ **Nota:**

> Si su cuenta de automatización está integrada con el control de código fuente, puede especificar como ruta de acceso del directorio la ruta de la carpeta del runbook de su repositorio de GitHub. Por ejemplo: cambiar el directorio a *C:\abc* almacenaría runbooks en *C:\abc\vscodeAutomation.*. o *C:\abc//subscriptionName//resourceGroupName//automationAccountName//runbookname.ps1*.

- **Operaciones de administración de runbook**: Puede crear un runbook, recuperar un borrador de runbook, recuperar un runbook publicado, abrir un runbook local en el editor, comparar un runbook local con una copia publicada o un borrador de runbook, cargar como borrador, publicar un runbook y eliminar un runbook de su cuenta de Automation.

- **Operaciones de ejecución de Runbook**: puede ejecutar una versión local de trabajos de Automation, como iniciar trabajos de Automation, Iniciar trabajo de prueba de Automation, ver salidas de trabajo y ejecutar la versión local del Runbook de PowerShell en modo de depuración, ya que permite agregar puntos de interrupción en el script.

  > ⓘ **Nota:**
  >
  > Actualmente, se admite el uso de cmdlets internos como `Get-AutomationVariable` solo con recursos no cifrados.

- **Trabajar con programaciones, recursos y webhooks**: puede ver las propiedades de una programación, eliminar programación, vincular programación para vincular una programación a un runbook.

- **Agregar webhook**: puede agregar un webhook al runbook.

- **Actualizar las propiedades de los recursos**: puede crear, actualizar, ver propiedades de recursos como Certificados, Conexiones, Credenciales, Variables y Eliminación de recursos de la extensión.

Limitaciones

Actualmente, no se admiten las siguientes características:

- Creación de nuevas programaciones.
- Agregar nuevos certificados en recursos.
- Cargue paquetes de módulos (PowerShell y Python) desde la extensión.

- Sincronización automática de runbooks locales con la cuenta de Azure Automation. Tendrá que realizar la operación para **Capturar** o **Publicar** runbook.
- Administración de grupos de trabajadores híbridos.
- Manual de operaciones gráfico y flujos de trabajo.
- Para Python, no se proporcionan opciones de depuración. Se recomienda instalar cualquier extensión del depurador en el script de Python.
- Actualmente, solo se admiten los recursos sin cifrar en la ejecución local.

Pasos siguientes

- Para las operaciones de administración de Runbook y para probar runbook y trabajos, consulte Uso de la extensión de Azure Automation para Visual Studio Code

---

Last updated on 17/11/2025

Controles de cumplimiento normativo de Azure Policy para Azure Automation

Cumplimiento normativo de Azure Policy proporciona definiciones de iniciativas creadas y administradas por Microsoft, conocidas como *integraciones*, para los **dominios de cumplimiento** y los **controles de seguridad** relativos a distintos estándares de cumplimiento. En esta página se enumeran los **dominios de cumplimiento** y los **controles de seguridad** para Azure Automation. Para que los recursos de Azure sean compatibles con el estándar específico, puede asignar las integraciones a un **control de seguridad** de manera individual.

El título de cada definición de directiva integrada se vincula a la definición de directiva en Azure Portal. Use el vínculo de la columna **Versión de directiva** para ver el origen en el Repositorio de GitHub de Azure Policy.

> **Importante**
>
> Cada control está asociado a una o varias definiciones de Azure Policy. Estas directivas pueden ayudarle a evaluar el cumplimiento con el control. Sin embargo, con frecuencia no hay una correspondencia completa o exacta entre un control y una o varias directivas. Como tal, el término **cumplimiento** en Azure Policy solo se refiere a las propias directivas. Esto no garantiza una compatibilidad total con todos los requisitos de un control. Además, el estándar de cumplimiento incluye controles que no se abordan en ninguna definición de Azure Policy en este momento. Por lo tanto, el cumplimiento en Azure Policy es solo una vista parcial del estado de cumplimiento general. Las asociaciones entre los controles y las definiciones de cumplimiento normativo de Azure Policy para estos estándares de cumplimiento pueden cambiar con el tiempo.

CMMC Nivel 3

Para ver la correspondencia de las integraciones de Azure Policy disponibles para todos los mapas de servicio de Azure con este estándar de cumplimiento, consulte Detalles de la iniciativa integrada de cumplimiento normativo CMMC nivel 3. Para más información sobre este estándar de cumplimiento, consulte Certificación del modelo de madurez de ciberseguridad (CMMC).

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Protección del sistema y de las comunicaciones	SC.3.177	Emplea criptografía validada mediante FIPS cuando se usa para proteger la confidencialidad de CUI.	Las variables de cuenta de Automation deben cifrarse	1.1.0
Protección del sistema y de las comunicaciones	SC.3.191	Protege la confidencialidad de CUI en reposo.	Las variables de cuenta de Automation deben cifrarse	1.1.0

FedRAMP High

Para revisar el modo en que las iniciativas integradas disponibles de Azure Policy de todos los servicios de Azure se asignan a este estándar de cumplimiento, consulte Cumplimiento normativo de Azure Policy: FedRAMP High. Para más información sobre este estándar de cumplimiento, consulte FedRAMP High.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Protección del sistema y de las comunicaciones	SC-12	Establecimiento y administración de claves criptográficas	Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en reposo.	1.0.0
Protección del sistema y de las comunicaciones	SC-28	Protección de la información en reposo	Las variables de cuenta de Automation deben cifrarse	1.1.0
Protección del sistema y de las comunicaciones	SC-28 (1)	Protección criptográfica	Las variables de cuenta de Automation deben cifrarse	1.1.0

FedRAMP Moderate

Para revisar el modo en que las iniciativas integradas disponibles de Azure Policy de todos los servicios de Azure se asignan a este estándar de cumplimiento, consulte Cumplimiento normativo de Azure Policy: FedRAMP Moderate. Para más información sobre este estándar de cumplimiento, consulte FedRAMP Moderate.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Protección del sistema y de las comunicaciones	SC-12	Establecimiento y administración de claves criptográficas	Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en reposo.	1.0.0
Protección del sistema y de las comunicaciones	SC-28	Protección de la información en reposo	Las variables de cuenta de Automation deben cifrarse	1.1.0
Protección del sistema y de las comunicaciones	SC-28 (1)	Protección criptográfica	Las variables de cuenta de Automation deben cifrarse	1.1.0

ISO 27001:2013

Si desea ver la correspondencia que existe entre las integraciones de Azure Policy disponibles para todos los servicios de Azure y este estándar de cumplimiento, consulte este artículo sobre el cumplimiento normativo de Azure Policy: ISO 27001:2013. Para más información sobre este estándar de cumplimiento, consulte ISO 27001:2013.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Cryptography	10.1.1	Directiva sobre el uso de controles criptográficos	Las variables de cuenta de Automation deben cifrarse	1.1.0

Punto de referencia de seguridad en la nube de Microsoft

El punto de referencia de seguridad en la nube de Microsoft proporciona recomendaciones sobre cómo puede proteger sus soluciones de nube en Azure. Para ver la asignación completa de este servicio al punto de referencia de seguridad en la nube de Microsoft, consulte Archivos de asignación de Azure Security Benchmark.

Para revisar el modo en que las integraciones de Azure Policy disponibles para todos los servicios de Azure se corresponden a este estándar de cumplimiento, consulte Cumplimiento normativo de Azure Policy: punto de referencia de seguridad en la nube de Microsoft.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Protección de datos	DP-4	DP-4 Habilitar el cifrado de datos en reposo de forma predeterminada	Las variables de cuenta de Automation deben cifrarse	1.1.0

NIST SP 800-171 R2

Para revisar el modo en que las integraciones de Azure Policy disponibles para todos los servicios de Azure se corresponden a este estándar de cumplimiento, consulte Cumplimiento normativo de Azure Policy: NIST SP 800-171 R2. Para obtener más información sobre este estándar de cumplimiento, consulte NIST SP 800-171 R2.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Protección del sistema y de las comunicaciones	3.13.10	Establece y administra las claves del mecanismo de cifrado que se utiliza en los sistemas de la organización.	Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en reposo.	1.0.0
Protección del sistema y de las	3.13.16	Protege la confidencialidad de CUI	Las variables de cuenta de Automation deben	1.1.0

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
comunicaciones		en reposo.	cifrarse	

NIST SP 800-53 Rev. 4

Para revisar el modo en que las iniciativas integradas disponibles de Azure Policy de todos los servicios de Azure se asignan a este estándar de cumplimiento, consulte Detalles de la iniciativa integrada del cumplimiento normativo de NIST SP 800-53 R4. Para más información sobre este estándar de cumplimiento, consulte NIST SP 800-53 Rev. 4.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Protección del sistema y de las comunicaciones	SC-12	Establecimiento y administración de claves criptográficas	Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en reposo.	1.0.0
Protección del sistema y de las comunicaciones	SC-28	Protección de la información en reposo	Las variables de cuenta de Automation deben cifrarse	1.1.0
Protección del sistema y de las comunicaciones	SC-28 (1)	Protección criptográfica	Las variables de cuenta de Automation deben cifrarse	1.1.0

NIST SP 800-53 Rev. 5

Para revisar el modo en que las iniciativas integradas disponibles de Azure Policy de todos los servicios de Azure se asignan a este estándar de cumplimiento, consulte Detalles de la iniciativa integrada del cumplimiento normativo de NIST SP 800-53 R5. Para más información sobre este estándar de cumplimiento, consulte NIST SP 800-53 Rev. 5.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Protección del sistema y de las comunicaciones	SC-12	Establecimiento y administración de una clave criptográfica	Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en reposo.	1.0.0
Protección del sistema y de las comunicaciones	SC-28	Protección de la información en reposo	Las variables de cuenta de Automation deben cifrarse	1.1.0
Protección del sistema y de las comunicaciones	SC-28 (1)	Protección criptográfica	Las variables de cuenta de Automation deben cifrarse	1.1.0

Tema de la nube de NL BIO

Para revisar cómo se asignan los complementos de Azure Policy disponibles para todos los servicios de Azure a este estándar de cumplimiento, consulte Detalles de cumplimiento normativo de Azure Policy para el tema de la nube de NL BIO. Para obtener más información sobre este estándar de cumplimiento, consulte Ciberseguridad gubernamental de seguridad de la información de base de referencia: gobierno digital (digitaleoverheid.nl).

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
U.05.2 Protección de datos: medidas criptográficas	U.05.2	Los datos almacenados en el servicio en la nube se protegerán al estado más reciente del arte.	Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en reposo.	1.0.0
U.07.3 Separación de datos: características de administración	U.07.3	U.07.3: Los privilegios para ver o modificar datos de CSC o claves de cifrado se conceden de forma controlada y se registra el uso.	Las variables de cuenta de Automation deben cifrarse	1.1.0

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
U.11.1 Servicios criptográficos: directiva	U.11.1	En la directiva de criptografía, se han elaborado al menos los temas de conformidad con la BIO.	Las variables de cuenta de Automation deben cifrarse	1.1.0
U.11.2 Servicios criptográficos: medidas criptográficas	U.11.2	En el caso de los certificados PKIoverheid, use los requisitos de PKIoverheid para la administración de claves. En otras situaciones, use ISO11770.	Las variables de cuenta de Automation deben cifrarse	1.1.0
Criptoservicios U.11.3: cifrado	U.11.3	Los datos confidenciales siempre se cifran, con claves privadas administradas por el CSC.	Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en reposo.	1.0.0

PCI DSS 3.2.1

Para revisar la correspondencia que existe entre las integraciones de Azure Policy disponibles para todos los servicios de Azure y este estándar de cumplimiento, consulte PCI DSS 3.2.1. Para más información sobre este estándar de cumplimiento, consulte PCI DSS 3.2.1.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Requisito 3	3.4	Requisito 3.4 de PCI DSS	Las variables de cuenta de Automation deben cifrarse	1.1.0
Requisito 4	4,1	Requisito 4.1 de PCI DSS	Las variables de cuenta de Automation deben cifrarse	1.1.0
Requisito 6	6.5.3	Requisito 6.5.3 de PCI DSS	Las variables de cuenta de Automation deben cifrarse	1.1.0

PCI DSS v4.0

Para revisar el modo en que las integraciones de Azure Policy disponibles para todos los servicios de Azure se asignan a este estándar de cumplimiento, consulte Detalles del cumplimiento normativo de Azure Policy de PCI DSS v4.0. Para más información sobre este estándar de cumplimiento, vea PCI DSS v4.0.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Requisito 03: Proteger los datos de la cuenta almacenada	3.5.1	El número de cuenta principal (PAN) está protegido dondequiera que se almacene	Las variables de cuenta de Automation deben cifrarse	1.1.0
Requisito 06: Desarrollar y mantener sistemas y software seguros	6.2.4	El software personalizado y a medida se desarrolla de forma segura	Las variables de cuenta de Automation deben cifrarse	1.1.0

Banco de la Reserva de la India: marco informático para NBFC

Para revisar la correspondencia entre las integraciones de Azure Policy disponibles para todos los servicios de Azure y este estándar de cumplimiento, vea Cumplimiento normativo de Azure Policy: Banco de la Reserva de la India: marco informático para NBFC. Para obtener más información sobre este estándar de cumplimiento, consulte Banco de la Reserva de la India: marco informático para NBFC.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Información y ciberseguridad	3.1.h	Infraestructura de clave pública (PKI)-3.1	Las variables de cuenta de Automation deben cifrarse	1.1.0

Marco de TI del Banco de Reserva de la India para bancos v2016

Para revisar la correspondencia entre las integraciones de Azure Policy disponibles para todos los servicios de Azure y este estándar de cumplimiento, vea Cumplimiento normativo de Azure Policy: RBI ITF Banks v2016. Para más información sobre este estándar de cumplimiento, consulte: RBI ITF Banks v2016 (PDF).

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Administración y defensa ante amenazas avanzadas en tiempo real		Advanced Real-Timethreat Defensa y Gestión-13.4	Las variables de cuenta de Automation deben cifrarse	1.1.0

España ENS

A fin de revisar el modo en que las integraciones de Azure Policy disponibles para todos los servicios de Azure se asignan a esta norma de cumplimiento, consulte Detalles del cumplimiento normativo de Azure Policy para España ENS. Para obtener más información sobre este estándar de cumplimiento, consulte CCN-STIC 884.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Medidas de protección	mp.com.3	Protección de las comunicaciones	Las variables de cuenta de Automation deben cifrarse	1.1.0
Marco operativo	op.acc.6	Control de acceso	Las variables de cuenta de Automation deben cifrarse	1.1.0

SWIFT CSP-CSCF v2021

Para revisar cómo las funciones integradas de Azure Policy disponibles para todos los mapas de servicio de Azure se asignan a este estándar de cumplimiento, consulte los detalles del

cumplimiento normativo de Azure Policy para SWIFT CSP-CSCF v2021. Para obtener más información sobre este estándar de cumplimiento, consulte SWIFT CSP CSCF v2021.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Reducir la superficie expuesta a ataques y vulnerabilidades	2.1	Seguridad del flujo de datos interno	Las variables de cuenta de Automation deben cifrarse	1.1.0
Reducir la superficie expuesta a ataques y vulnerabilidades	2.4A	Seguridad del flujo de datos administrativo	Las variables de cuenta de Automation deben cifrarse	1.1.0
Reducir la superficie expuesta a ataques y vulnerabilidades	2.5A	Protección de datos de transmisión externa	Las variables de cuenta de Automation deben cifrarse	1.1.0

SWIFT CSP-CSCF v2022

Para revisar cómo las funciones integradas de Azure Policy disponibles para todos los servicios de Azure se asignan a este estándar de cumplimiento, consulte los detalles del cumplimiento normativo de Azure Policy para SWIFT CSP-CSCF v2022. Para obtener más información sobre este estándar de cumplimiento, consulte SWIFT CSP CSCF v2022.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
2. Reducir la superficie expuesta a ataques y las vulnerabilidades	2.1	Garantice la confidencialidad, integridad y autenticidad de los flujos de datos de aplicación entre los componentes locales relacionados con SWIFT.	Las variables de cuenta de Automation deben cifrarse	1.1.0

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
2. Reducir la superficie expuesta a ataques y las vulnerabilidades	2.4A	Seguridad del flujo de datos administrativo	Las variables de cuenta de Automation deben cifrarse	1.1.0
2. Reducir la superficie expuesta a ataques y las vulnerabilidades	2.5A	Protección de datos de transmisión externa	Las variables de cuenta de Automation deben cifrarse	1.1.0

Controles de sistema y organización (SOC) 2

Para examinar la forma en que los elementos integrados de Azure Policy disponibles para todos los servicios de Azure se asignan a este estándar de cumplimiento, consulte los Detalles de la iniciativa integrada de cumplimiento normativo de Controles de organización y sistema (SOC) 2. Para más información sobre este estándar de cumplimiento, consulte Controles de organización y sistema (SOC) 2.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Controles de acceso lógico y físico	CC6.1	Software, infraestructura y arquitecturas de seguridad del acceso lógico	Las variables de cuenta de Automation deben cifrarse	1.1.0

UK OFFICIAL y UK NHS

Para ver cómo se asignan las integraciones de Azure Policy disponibles para todos los servicios de Azure a este estándar de cumplimiento, consulte el artículo sobre el cumplimiento normativo de Azure Policy: UK OFFICIAL y UK NHS. Para más información sobre este estándar de cumplimiento, consulte el artículo acerca de UK OFFICIAL.

Expandir tabla

Dominio	Identificador de control	Título de control	Directiva (Azure Portal)	Versión de directiva (GitHub)
Resistencia y protección de los recursos	2.3	Protección de los datos en reposo	Las variables de cuenta de Automation deben cifrarse	1.1.0

Pasos siguientes

- Obtenga más información sobre el cumplimiento normativo de Azure Policy.
- Los elementos integrados se pueden encontrar en el repositorio de GitHub de Azure Policy.

Last updated on 20/10/2025

Base de referencia de seguridad de Azure para Automation

Esta línea de base de seguridad aplica las instrucciones del Microsoft Cloud Security Benchmark versión 1.0 a la automatización. El banco de pruebas de seguridad en la nube de Microsoft proporciona recomendaciones sobre cómo proteger las soluciones en la nube en Azure. El contenido se agrupa mediante los controles de seguridad definidos por el banco de pruebas de seguridad en la nube de Microsoft y las instrucciones relacionadas aplicables a automation.

Puede supervisar esta línea de base de seguridad y sus recomendaciones mediante Microsoft Defender for Cloud. Azure Policy definiciones se enumerarán en la sección Cumplimiento normativo de la página del portal de Microsoft Defender for Cloud.

Cuando una característica tiene definiciones de Azure Policy pertinentes, se muestran en esta línea de base para ayudarle a medir el cumplimiento de los controles y recomendaciones de pruebas comparativas de seguridad en la nube de Microsoft. Algunas recomendaciones pueden requerir un plan de Microsoft Defender de pago para habilitar determinados escenarios de seguridad.

> **Nota:**
>
> Se han excluido **las características** no aplicables a Automation. Para ver cómo Automation se mapea completamente al punto de referencia de seguridad de la nube de Microsoft, consulte el archivo completo de mapeo de la línea base de seguridad de Automation.

Perfil de seguridad

El perfil de seguridad resume los comportamientos de alto impacto de Automation, lo que puede dar lugar a un aumento de las consideraciones de seguridad.

Expandir tabla

Atributo de comportamiento del servicio	Importancia
Categoría del producto	MGMT/Governance
El cliente puede acceder a HOST/OS.	Sin acceso
El servicio se puede implementar en la red virtual del cliente.	Cierto
Almacena el contenido del cliente en reposo	Cierto

Seguridad de red

Para obtener más información, consulte la prueba comparativa de seguridad en la nube de Microsoft: Seguridad de red.

NS-1: Establecer límites de segmentación de red

Características

Integración de red virtual

Descripción: el servicio admite la implementación en la red virtual privada (VNet) del cliente. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Falso	Cliente

Guía de configuración: implemente el servicio en una red virtual. Asigne direcciones IP privadas al recurso (si procede). Esta es la configuración recomendada desde un punto de vista de seguridad; Sin embargo, esto requiere que configure Hybrid Runbook Worker conectado a una red virtual de Azure y actualmente no admite trabajos en la nube.

Conectividad de red para Azure Automation

Referencia: Uso de Azure Private Link para conectar redes de forma segura a Azure Automation

NS-2: Protección de servicios en la nube con controles de red

Características

Azure Private Link

Descripción: funcionalidad de filtrado de IP nativa del servicio para filtrar el tráfico de red (no confundirse con NSG o Azure Firewall). Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Falso	Cliente

Guía de configuración: implemente puntos de conexión privados para todos los recursos de Azure que admiten la característica Private Link para establecer un punto de acceso privado para los recursos.

Referencia: Uso de Azure Private Link para conectar redes de forma segura a Azure Automation

Deshabilitar el acceso a la red pública

Descripción: el servicio admite la deshabilitación del acceso a la red pública mediante el uso de una regla de filtrado de ACL de IP de nivel de servicio (no NSG o Azure Firewall) o mediante un modificador de alternancia "Deshabilitar acceso a red pública". Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Falso	Cliente

Notas de características: el servicio Azure Automation admite la deshabilitación del acceso a la red pública a través de una instancia integrada de Azure Policy o también puede usar el cmdlet de PowerShell: establecimiento de marcas de acceso de red pública.

Guía de configuración: deshabilite el acceso a la red pública mediante el cmdlet de PowerShell o un conmutador de alternancia para el acceso a la red pública.

Referencia: Las cuentas de Automation deben deshabilitar el acceso a la red pública

Administración de identidades

Para obtener más información, consulte El banco de pruebas de seguridad en la nube de Microsoft: Administración de identidades.

IM-1: Uso del sistema de autenticación e identidad centralizado

Características

Autenticación de Azure AD necesaria para el acceso al plano de datos

Descripción: el servicio admite el uso de la autenticación de Azure AD para el acceso al plano de datos. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Cierto	Microsoft

Notas de características: la característica de usuario Hybrid Runbook Worker basada en extensiones (v2) de Azure Automation se utiliza para ejecutar runbooks directamente en una máquina de Azure o en una máquina que no es de Azure. Esta ejecución se realiza a través de servidores habilitados para Azure Arc registrados, utilizando la autenticación de Azure AD.

Guía de configuración: no se requieren configuraciones adicionales, ya que está habilitada en una implementación predeterminada.

Referencia: Introducción a la autenticación de cuentas de Azure Automation

Métodos de autenticación local para el acceso al plano de datos

Descripción: métodos de autenticación locales admitidos para el acceso al plano de datos, como un nombre de usuario local y una contraseña. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Falso	Cliente

Notas de características: el servicio Azure Automation admite el método de autenticación local basado en certificados para el acceso al plano de datos a través de ventanas basadas en agente (v1) o Linux Hybrid Runbook Worker, pero no es el enfoque recomendado para incorporar Hybrid Worker. Use el método de instalación de Hybrid Runbook Worker basado en extensiones (v2) como enfoque recomendado. Evite el uso de métodos o cuentas de autenticación local, que deben deshabilitarse siempre que sea posible. en su lugar, use Azure AD para autenticarse siempre que sea posible.

Guía de configuración: restrinja el uso de métodos de autenticación local para el acceso al plano de datos. En su lugar, use Azure Active Directory (Azure AD) como método de autenticación predeterminado para controlar el acceso al plano de datos.

Deshabilitación de la autenticación local en Automation

Referencia: Implementar un Hybrid Runbook Worker de Windows basado en agente en Automation

IM-3: Administrar identidades de aplicación de forma segura y automática

Características

Identidades administradas

Descripción: las acciones del plano de datos admiten la autenticación mediante identidades administradas. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Cierto	Microsoft

Notas de características: la identidad administrada del sistema se crea de forma predeterminada si la cuenta se crea a través del portal, pero no de forma predeterminada si la cuenta se crea a través de API o cmdlet. También se puede habilitar la creación de cuentas posteriores.

Guía de configuración: no se requieren configuraciones adicionales, ya que está habilitada en una implementación predeterminada.

Referencia: Identidades administradas

Entidades de servicio

Descripción: el plano de datos admite la autenticación mediante entidades de servicio. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Falso	No es aplicable	No es aplicable

Guía de configuración: esta característica no se admite para proteger este servicio.

IM-8: Restringir la exposición de credenciales y secretos

Características

Integración y almacenamiento de credenciales de servicio y secretos con soporte en Azure Key Vault.

Descripción: el plano de datos admite el uso nativo de Azure Key Vault para el almacén de credenciales y secretos. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Cierto	Microsoft

Notas de características: Los recursos seguros de Azure Automation incluyen credenciales, certificados, conexiones y variables cifradas. Estos recursos se cifran y se almacenan en Automation con una clave única que se genera para cada cuenta de Automation. Automation almacena la clave en el servicio de Key Vault administrado por el sistema. Antes de almacenar un recurso seguro, Automation carga la clave desde Key Vault y después la usa para cifrar el recurso.

Guía de configuración: no se requieren configuraciones adicionales, ya que está habilitada en una implementación predeterminada.

Referencia: Administración de credenciales en Azure Automation

Acceso con privilegios

Para obtener más información, consulte la prueba comparativa de seguridad en la nube de Microsoft: Acceso con privilegios.

PA-7: Siga el principio de administración suficiente (privilegios mínimos)

Características

RBAC de Azure para el plano de datos

Descripción: Azure Role-Based Access Control (Azure RBAC) se puede usar para administrar el acceso a las acciones del plano de datos del servicio. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Cierto	Microsoft

Notas de características: Automation se integra con Azure RBAC para administrar sus recursos. Con RBAC, se administra el acceso a los recursos de Azure a través de asignaciones de roles. Puede asignar roles a usuarios, grupos, entidades de servicio e identidades administradas. Algunos recursos tienen roles predefinidos e integrados. Puede realizar un inventario o consultar estos roles mediante herramientas como la CLI de Azure, Azure PowerShell o Azure Portal.

Guía de configuración: no se requieren configuraciones adicionales, ya que está habilitada en una implementación predeterminada.

Referencia: Administración de permisos de rol y seguridad en Azure Automation

PA-8: Determinación del proceso de acceso para la compatibilidad con proveedores en la nube

Características

Caja de seguridad del cliente

Descripción: La Customer Lockbox puede utilizarse para el acceso de soporte técnico de Microsoft. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Falso	No es aplicable	No es aplicable

Notas de características: Lockbox no se implementa para Azure Automation; en su lugar, el servicio Azure Automation cifra el script de Runbook y la configuración de DSC con claves

administradas por el cliente antes de almacenar los datos en SQL Database, haciendo que los recursos de automatización estén cifrados.

/en-us/azure/automation/whats-new-archive#added-capability-to-keep-automation-runbooks-and-dsc-scripts-encrypted-by-default

Guía de configuración: esta característica no se admite para proteger este servicio.

Protección de los datos

Para obtener más información, consulte la prueba comparativa de seguridad en la nube de Microsoft: Protección de datos.

DP-3: Cifrar datos confidenciales en tránsito

Características

Cifrado de datos en tránsito

Descripción: el servicio admite el cifrado de datos en tránsito para el plano de datos. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Cierto	Microsoft

Guía de configuración: no se requieren configuraciones adicionales, ya que está habilitada en una implementación predeterminada.

Referencia: TLS 1.2 para Azure Automation

DP-4: Habilitar el cifrado de datos en reposo de forma predeterminada

Características

Cifrado de datos en reposo mediante claves de plataforma

Descripción: se admite el cifrado de datos en reposo mediante claves de plataforma, cualquier contenido del cliente en reposo se cifra con estas claves administradas por Microsoft. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Cierto	Microsoft

Notas de características: Los recursos seguros de Azure Automation incluyen credenciales, certificados, conexiones y variables cifradas. Estos recursos están protegidos en Azure Automation mediante varios niveles de cifrado. De forma predeterminada, la cuenta de Azure Automation usa claves administradas por Microsoft.

Guía de configuración: no se requieren configuraciones adicionales, ya que está habilitada en una implementación predeterminada.

Referencia: Claves administradas por Microsoft

Supervisión de Microsoft Defender for Cloud

Definiciones integradas de Azure Policy: Microsoft.Automation:

Expandir tabla

Nombre (Azure Portal)	Description	Effect(s)	Versión (GitHub)
Las variables de cuenta de Automation deben cifrarse	Es importante habilitar el cifrado de recursos de variables de cuentas de Automation al almacenar datos confidenciales.	Auditar, Denegar, Deshabilitado	1.1.0

DP-5: Usar la opción de clave administrada por el cliente en el cifrado de datos en reposo cuando sea necesario

Características

Cifrado de datos en reposo mediante CMK

Descripción: el cifrado de datos en reposo mediante claves administradas por el cliente es compatible con el contenido del cliente almacenado por el servicio. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Falso	Cliente

Guía de configuración: si es necesario para el cumplimiento normativo, defina el caso de uso y el ámbito de servicio donde se necesita el cifrado mediante claves administradas por el cliente. Habilite e implemente el cifrado de datos en reposo mediante la clave administrada por el cliente para esos servicios.

Referencia: Cifrado de recursos seguros en Azure Automation

DP-6: Uso de un proceso seguro de administración de claves

Características

Administración de claves en Azure Key Vault

Descripción: el servicio admite la integración de Azure Key Vault para cualquier clave de cliente, secretos o certificados. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Falso	Cliente

Notas de características: Azure Automation no admite la integración con Key Vault de forma nativa para almacenar los secretos personalizados usados por sus runbooks de Automation, pero pueden acceder a Key Vault mediante los cmdlets de Key Vault desde el código del runbook de Automation.

Guía de configuración: use Azure Key Vault para crear y controlar el ciclo de vida de las claves de cifrado, incluida la generación de claves, la distribución y el almacenamiento. Gire y revoque las claves en Azure Key Vault y el servicio en función de una programación definida o cuando haya una retirada o riesgo de claves. Cuando sea necesario usar la clave administrada por el cliente (CMK) en el nivel de carga de trabajo, servicio o aplicación, asegúrese de seguir los procedimientos recomendados para la administración de claves: use una jerarquía de claves para generar una clave de cifrado de datos (DEK) independiente con la clave de cifrado de claves (KEK) en el almacén de claves. Asegúrese de que las claves se registran con Azure Key Vault y se hace referencia a ellas a través de identificadores de clave desde el servicio o la

aplicación. Si necesita traer su propia clave (BYOK) al servicio (por ejemplo, importar claves protegidas con HSM desde los HSM locales a Azure Key Vault), siga las instrucciones recomendadas para realizar la generación inicial de claves y la transferencia de claves.

DP-7: Uso de un proceso de administración de certificados seguro

Características

Administración de certificados en Azure Key Vault

Descripción: el servicio admite la integración de Azure Key Vault para cualquier certificado de cliente. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Falso	Cliente

Notas de características: Los recursos seguros de Azure Automation incluyen credenciales, certificados, conexiones y variables cifradas. Estos recursos se cifran y se almacenan en Automation con una clave única que se genera para cada cuenta de Automation. Automation almacena la clave en el servicio de Key Vault administrado por el sistema. Antes de almacenar un recurso seguro, Automation carga la clave desde Key Vault y después la usa para cifrar el recurso.

Guía de configuración: use Azure Key Vault para crear y controlar el ciclo de vida del certificado, incluida la creación, importación, rotación, revocación, almacenamiento y purga del certificado. Asegúrese de que la generación de certificados sigue los estándares definidos sin usar ninguna propiedad no segura, como: tamaño de clave insuficiente, período de validez excesivamente largo, criptografía no segura. Configure la rotación automática del certificado en Azure Key Vault y el servicio de Azure (si se admite) en función de una programación definida o cuando haya una expiración del certificado. Si no se admite la rotación automática en la aplicación, asegúrese de que siguen girando mediante métodos manuales en Azure Key Vault y la aplicación.

Referencia: Administración de certificados en Azure Automation

Administración de recursos

Para obtener más información, consulte El banco de pruebas de seguridad en la nube de Microsoft: Administración de recursos.

AM-2: Usar solo servicios aprobados

Características

Compatibilidad con Azure Policy

Descripción: las configuraciones de servicio se pueden supervisar y aplicar a través de Azure Policy. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Cierto	Microsoft

Guía de configuración: no se requieren configuraciones adicionales, ya que está habilitada en una implementación predeterminada.

Referencia: Definiciones integradas de Azure Policy para Azure Automation

Registro y detección de amenazas

Para obtener más información, consulte la prueba comparativa de seguridad en la nube de Microsoft: Registro y detección de amenazas.

LT-1: Habilitación de las funcionalidades de detección de amenazas

Características

Oferta de microsoft Defender para servicio o producto

Descripción: El servicio tiene una solución específica de Microsoft Defender para supervisar y alertar sobre problemas de seguridad. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Falso	No es aplicable	No es aplicable

Guía de configuración: esta característica no se admite para proteger este servicio.

LT-4: Habilitación del registro para la investigación de seguridad

Características

Registros de recursos de Azure

Descripción: el servicio genera registros de recursos que pueden proporcionar métricas y registros específicos del servicio mejorados. El cliente puede configurar estos registros de recursos y enviarlos a su propio receptor de datos, como una cuenta de almacenamiento o un área de trabajo de Log Analytics. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Cierto	Falso	Cliente

Notas de características: Azure Automation puede enviar el estado del trabajo y flujos de trabajo de runbook al área de trabajo de Log Analytics. Los registros de trabajos y los flujos de trabajo están visibles en Azure Portal o con PowerShell para trabajos individuales.

Guía de configuración: habilite los registros de recursos para el servicio. El contenido de los registros de recursos varía según el tipo de recurso y el servicio de Azure. Azure Automation puede enviar flujos de trabajo y estado del trabajo de runbook al área de trabajo de Log Analytics. Los registros de trabajos y los flujos de trabajo están visibles en Azure Portal o con PowerShell para trabajos individuales.

Referencia: Reenvío de registros de diagnóstico de Azure Automation a Azure Monitor

Copia de seguridad y recuperación

Para obtener más información, consulte la prueba comparativa de seguridad en la nube de Microsoft: Copia de seguridad y recuperación.

BR-1: Garantizar copias de seguridad automatizadas normales

Características

Azure Backup

Descripción: el servicio de Azure Backup puede realizar una copia de seguridad del servicio. Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Falso	No es aplicable	No es aplicable

Notas de características: no se admite la copia de seguridad de Azure Automation a través de Azure Backup. Es su responsabilidad asegurarse de mantener una copia de seguridad válida de la configuración de Automatización, como libros de ejecución y activos.

Puede usar Azure Resource Manager para implementar cuentas de Automation y recursos relacionados. Puede exportar plantillas de Azure Resource Manager para usarlas como copias de seguridad para restaurar cuentas de Automation y recursos relacionados. Use Automation para llamar a la API de exportación de plantillas de Azure Resource Manager con regularidad.

Siga (Automation Data Backup) [/azure/automation/automation-managing-data#data-backup] para la configuración de esta característica. Revise y determine si su organización quiere configurar esta característica de seguridad. También puede aprovechar las instrucciones sobre la configuración (recuperación ante desastres)[/azure/automation/automation-disaster-recovery?tabs=win-hrw%2Cps-script%2Coption-one] para las cuentas de Automation.

También puede utilizar la función de integración con el control de código fuente para mantener los runbooks de la cuenta de Automation actualizados con los scripts del repositorio de control de código fuente.

Guía de configuración: esta característica no se admite para proteger este servicio.

Funcionalidad de copia de seguridad nativa del servicio

Descripción: el servicio admite su propia funcionalidad de copia de seguridad nativa (si no usa Azure Backup). Más información.

Expandir tabla

Compatible	Habilitado de forma predeterminada	Responsabilidad de configuración
Falso	No es aplicable	No es aplicable

Notas de características: Azure Automation no proporciona un mecanismo nativo de copia de seguridad. Es su responsabilidad asegurarse de mantener una copia de seguridad válida de la configuración de Automation, como libros de ejecución y recursos.

Puede usar Azure Resource Manager para implementar cuentas de Automation y recursos relacionados. Puede exportar plantillas de Azure Resource Manager para usarlas como copias de seguridad para restaurar cuentas de Automation y recursos relacionados. Use Automation para llamar a la API de exportación de plantillas de Azure Resource Manager con regularidad.

También puede usar la función de integración del control de código fuente para mantener los runbooks de la cuenta de Automation actualizados con los scripts en el repositorio de control de versiones.

Copia de seguridad automática de datos

Revise y determine si su organización quiere configurar esta característica de seguridad. También puede aprovechar las instrucciones sobre cómo configurar la recuperación ante desastres para las cuentas de Automation.

Guía de configuración: esta característica no se admite para proteger este servicio.

Pasos siguientes

- Consulte la introducción al banco de pruebas de seguridad en la nube de Microsoft.
- Más información sobre las líneas base de seguridad de Azure

---

Last updated on 03/02/2026

Procedimientos recomendados de seguridad en Azure Automation

En este artículo se detallan los procedimientos recomendados para ejecutar de forma segura los trabajos de automatización. Azure Automation proporciona la plataforma para orquestar tareas operativas y de administración de infraestructura frecuentes, propensas a errores, así como operaciones críticas. Este servicio permite ejecutar scripts, conocidos como runbooks de automatización sin problemas en entornos híbridos y en la nube.

Los componentes de plataforma del servicio Azure Automation se protegen activamente. El servicio pasa por comprobaciones sólidas de seguridad y cumplimiento. El banco de pruebas de seguridad en la nube de Microsoft detalla los procedimientos recomendados y recomendaciones para ayudar a mejorar la seguridad de las cargas de trabajo, los datos y los servicios en Azure. Consulte también Línea base de seguridad de Azure para Azure Automation.

Configuración segura de la cuenta de Automation

Esta sección le guía en la configuración de la cuenta de Automation de forma segura.

Permisos

1. Siga el principio de privilegios mínimos para realizar el trabajo al conceder acceso a los recursos de Automation. Implemente los roles RBAC detallados de Automation y evite asignar roles o ámbitos más amplios, como el nivel de suscripción. Al crear los roles personalizados, incluya solo los permisos que necesitan los usuarios. Al limitar los roles y los ámbitos, también se limitan los recursos en riesgo si la entidad de seguridad llegara a verse comprometida. Para obtener información detallada sobre los conceptos de control de acceso basado en rol, consulte Procedimientos recomendados para el control de acceso basado en roles de Azure.

2. Evite los roles que incluyen acciones que tienen un carácter comodín (*), ya que implica el acceso total al recurso de Automation o a un subrecurso, por ejemplo *automationaccounts/*/read*. En su lugar, use acciones específicas únicamente para el permiso requerido.

3. Configure el acceso basado en roles en un nivel de runbook si el usuario no requiere acceso a todos los runbooks de la cuenta de Automation.

4. Limite el número de roles con privilegios elevados, como colaborador de Automation, para reducir la posibilidad de infracción por parte de un propietario en peligro.

5. Use Microsoft Entra Privileged Identity Management para proteger las cuentas con privilegios frente a ataques cibernéticos malintencionados para aumentar la visibilidad de su uso a través de informes y alertas.

Protección del rol de Hybrid Runbook Worker

1. Instale Hybrid Worker mediante la extensión de máquina virtual hybrid Runbook Worker que no tiene ninguna dependencia en el agente de Log Analytics. Recomendamos esta plataforma, ya que usa autenticación basada en Microsoft Entra ID. La característica Hybrid Runbook Worker de Azure Automation permite ejecutar runbooks directamente en la máquina que hospeda el rol en Azure o en una máquina que no es de Azure para ejecutar trabajos de Automation en el entorno local.

   - Use solo usuarios con privilegios elevados o roles personalizados de Hybrid Worker para los usuarios responsables de administrar operaciones como registrar o anular el registro de Hybrid Worker y grupos híbridos y ejecutar runbooks en grupos de Hybrid Runbook Worker.
   - El mismo usuario también requeriría acceso de colaborador de máquina virtual en la máquina que hospeda el rol de Hybrid Worker. Puesto que el colaborador de la máquina virtual es un rol con privilegios elevados, asegúrese de que solo un conjunto de derechos limitado de usuarios tenga acceso para administrar trabajos híbridos, lo que reduce la posibilidad de vulneración por parte de un propietario en peligro.

   Siga los procedimientos recomendados de RBAC de Azure.

2. Siga el principio de privilegios mínimos y conceda solo los permisos necesarios a los usuarios para la ejecución de runbooks en Hybrid Worker. No proporcione permisos sin restricciones a la máquina que hospeda el rol de Hybrid Runbook Worker. En caso de acceso sin restricciones, un usuario con el rol Colaborador de máquina virtual o con permisos para ejecutar comandos en el equipo de trabajo híbrido puede usar el certificado Run As de la cuenta de Automation desde ese equipo y podría permitir que un usuario malintencionado obtuviera acceso como colaborador de la suscripción. Esto podría poner en peligro la seguridad del entorno de Azure. Use los roles personalizados de Hybrid Worker para los usuarios responsables de administrar runbooks de Automation en Hybrid Runbook Worker y grupos de Hybrid Runbook Worker.

3. Anule el registro de todos los trabajadores híbridos no utilizados o que no responden.

4. Se recomienda que nunca configure la extensión de Hybrid Worker en una máquina virtual que hospeda el controlador de dominio. Los procedimientos recomendados de seguridad no aconsejan esta configuración debido a la naturaleza de alto riesgo de exponer controladores de dominio a posibles vectores de ataque a través de trabajos de Azure Automation. Los controladores de dominio deben estar altamente protegidos y aislados de servicios no esenciales para evitar el acceso no autorizado y mantener la integridad del entorno de Active Directory Domain Services (ADDS).

Certificado de autenticación e identidades

1. Para la autenticación de runbook, se recomienda usar identidades administradas. Una identidad administrada de Microsoft Entra ID permite que el runbook acceda fácilmente a otros recursos protegidos por Microsoft Entra, como Azure Key Vault. La plataforma Azure administra la identidad y no requiere que aprovisione ni rote secretos. Para más información acerca de las identidades administradas en Azure Automation, consulte Identidades administradas para Azure Automation.

   Puede autenticar una cuenta de Automation mediante dos tipos de identidades administradas:

   - Una **identidad asignada por el sistema** está asociada a la aplicación y se elimina si se elimina la aplicación. Una aplicación solo puede tener una identidad asignada por el sistema.
   - Una **identidad asignada por el usuario** es un recurso de Azure independiente que puede asignarse a la aplicación. Una aplicación puede tener varias identidades asignadas por el usuario.

   Siga las recomendaciones de prácticas recomendadas para identidades administradas para más información.

2. Rote las claves de Azure Automation periódicamente. La regeneración de claves impide que los futuros registros de nodos de DSC o de trabajo híbrido usen claves anteriores. Se recomienda usar los trabajadores híbridos basados en extensiones que utilizan la autenticación de Microsoft Entra en lugar de las claves de automatización. Microsoft Entra ID centraliza el control y la administración de identidades y credenciales de recursos.

Seguridad de los datos

1. Proteja los recursos de Azure Automation incluidas credenciales, certificados, conexiones y variables cifradas. Estos recursos se protegen en Azure Automation mediante varios niveles

de cifrado. De manera predeterminada, los datos se cifran con claves administradas por Microsoft. Para tener un mayor control sobre las claves de cifrado, puede proporcionar claves administradas por el cliente para usar el cifrado de recursos de Automation. Estas claves deben estar presentes en Azure Key Vault para que el servicio Automation pueda acceder a las claves. Consulte Cifrado de recursos seguros mediante claves administradas por el cliente.

2. No imprima ninguna credencial ni detalles de certificado en la salida del trabajo. Un operador de trabajo de Automation que es el usuario con pocos privilegios puede ver la información confidencial.

3. Mantenga una copia de seguridad válida de la configuración de Automation, como runbooks y recursos, asegurándose de que las copias de seguridad se validan y protegen para mantener la continuidad empresarial después de un evento inesperado.

Aislamiento de red

1. Use Azure Private Link para conectar de forma segura instancias de Hybrid Runbook Worker a Azure Automation. El punto de conexión privado de Azure es una interfaz de red que le conecta de forma privada y segura a un servicio de Azure Automation con tecnología de Azure Private Link. El punto de conexión privado usa una dirección IP privada de la red virtual para traer eficazmente el servicio Automation a la red virtual.

Si quiere acceder a otros servicios y administrarlos de forma privada a través de runbooks desde la red virtual de Azure sin necesidad de abrir una conexión saliente a Internet, puede ejecutar runbooks en una instancia de Hybrid Worker que esté conectado a la red virtual de Azure.

Directivas para Azure Automation

Revise las recomendaciones de Azure Policy para Azure Automation y actúe según corresponda. Consulte Directivas de Azure Automation.

Pasos siguientes

- Para obtener información sobre cómo usar el control de acceso basado en rol de Azure (RBAC de Azure), consulte Administración de los permisos de rol y la seguridad en Azure Automation.
- Para obtener información sobre cómo Azure protege su privacidad y protege los datos, consulte Seguridad de los datos de Azure Automation.

- Para obtener información sobre cómo configurar la cuenta de Automation para usar el cifrado, consulte Cifrado de recursos seguros en Azure Automation.
- Para obtener información sobre la compatibilidad con versiones en tiempo de ejecución, la retirada y las actualizaciones de seguridad, consulte Compatibilidad y directiva de retirada de Language Runtime para Azure Automation.

Last updated on 29/06/2026

Cifrado de recursos seguros en Azure Automation

Azure Automation protege recursos como credenciales, certificados, conexiones y variables cifradas que usan varios niveles de cifrado. Esto ayuda a mejorar la seguridad de estos recursos. Además, para garantizar una mayor seguridad y privacidad para el código del cliente, los runbooks y los scripts de DSC también se cifran. El cifrado de Azure Automation sigue dos modelos, en función de la clave de nivel superior que se use para el cifrado:

- Usando claves administradas por Microsoft
- Uso de claves que administra

Claves administradas por Microsoft

De forma predeterminada, la cuenta de Azure Automation usa claves de cifrado administradas por Microsoft.

Cada recurso seguro se cifra y se almacena en Azure Automation con una clave única(clave de cifrado de datos) que se genera para cada cuenta de Automation. Estas claves se cifran y almacenan en Azure Automation con otra clave única que se genera para cada cuenta, denominada clave de cifrado de cuenta (AEK). Estas claves de cifrado de cuenta se cifran y almacenan en Azure Automation mediante claves administradas por Microsoft.

Claves que se administran con Key Vault

Puede administrar el cifrado de recursos seguros para su cuenta de Automation con sus propias claves. Cuando especifica una clave administrada por el cliente en el nivel de la cuenta de Automation, esa clave se usa para proteger y controlar el acceso a la clave de cifrado de cuenta para la cuenta de Automation. Esta, a su vez, se usa para cifrar y descifrar todos los recursos seguros. Las claves administradas por el cliente ofrecen más flexibilidad para crear, rotar, deshabilitar y revocar controles de acceso. También puede auditar las claves de cifrado que se usan para proteger los recursos seguros.

Use Azure Key Vault para almacenar las claves administradas por el cliente. Puede crear sus propias claves y almacenarlas en un almacén de claves, o puede usar las API de Azure Key Vault para generarlas.

La habilitación de Azure Firewall en Azure Key Vault bloquea el acceso desde runbooks de Azure Automation para ese servicio. El acceso se bloqueará incluso cuando la excepción de firewall para permitir servicios de Microsoft de confianza esté habilitada, ya que Automation no forma parte de la lista de servicios de confianza. Con un firewall habilitado, el acceso solo se

puede realizar mediante una instancia de Hybrid Runbook Worker y un punto de conexión de servicio de red virtual. Sin embargo, cuando se habilita un vínculo privado de Key Vault, Azure Automation pierde el acceso a Key Vault. Incluso si habilita un vínculo privado para Hybrid Runbook Worker, solo permitirá el acceso al servicio Azure Automation, y no a Key Vault.

Para obtener más información sobre Azure Key Vault, consulte ¿Qué es Azure Key Vault?

Uso de las claves administradas por el cliente para una cuenta de Automation

Al usar el cifrado con claves administradas por el cliente para una cuenta de Automation, Azure Automation encapsula la clave de cifrado de la cuenta con la clave administrada por el cliente en el almacén de claves asociado. La habilitación de claves administradas por el cliente no afecta al rendimiento, y la cuenta se cifra con la nueva clave inmediatamente sin retraso alguno.

Las cuentas de Automation nuevas siempre se cifran mediante claves administradas por Microsoft. No es posible habilitar claves administradas por el cliente en el momento en que se crea la cuenta. Las claves administradas por el cliente se almacenan en Azure Key Vault, y el almacén de claves se debe aprovisionar con directivas de acceso que concedan permisos de clave a la identidad administrada que está asociada a la cuenta de Automation. La identidad administrada está disponible sólo después de la creación de la cuenta de automatización.

Al modificar la clave que se usa para el cifrado de recursos seguros de Azure Automation mediante la habilitación o deshabilitación de las claves administradas por el cliente, la actualización de la versión de la clave o la especificación de otra clave, el cifrado de la clave de cifrado de cuenta cambia, pero los recursos seguros de la cuenta de Azure Storage no tienen que volver a cifrarse.

> **Nota:**
>
> Para habilitar las claves administradas por el cliente con llamadas API REST de Azure Automation, tendrá que usar la versión de API 2020-01-13-preview.

Requisitos previos para el uso de claves administradas por el cliente en Azure Automation

Antes de habilitar las claves administradas por el cliente para una cuenta de Automation, debe asegurarse de que se cumplen los siguientes requisitos previos:

- Una instancia de Azure Key Vault con las propiedades **Eliminación temporal** y **No purgar** habilitadas. Estas propiedades son necesarias para permitir la recuperación de claves en caso de eliminación accidental.
- Solo se admiten claves RSA con para el cifrado de Azure Automation. Para obtener más información acerca de las claves, consulte Información acerca de claves, secretos y certificados de Azure Key Vault.
- La cuenta de Automation y el almacén de claves pueden estar en distintas suscripciones, pero tienen que estar en el mismo inquilino de Microsoft Entra.
- Al usar PowerShell, compruebe que el módulo Azure Az de PowerShell está instalado. Para instalar o actualizar, vea Instalación del módulo Azure Az de PowerShell.

Generación y asignación de una nueva identidad asignada por el sistema para una cuenta de Automation

Para usar las claves administradas por el cliente con una cuenta de Automation, la cuenta de Automation debe autenticarse en el almacén de claves que contiene las claves administradas por el cliente. Azure Automation usa identidades administradas asignadas por el sistema para autenticar la cuenta con Azure Key Vault. Para más información sobre las identidades administradas, consulte el artículo sobre Qué son las identidades administradas para recursos de Azure.

Mediante PowerShell

Use el cmdlet Set-AzAutomationAccount de PowerShell para modificar una cuenta de Azure Automation existente. El parámetro `-AssignSystemIdentity` genera y asigna una nueva identidad asignada por el sistema para que la cuenta de Automation la use con otros servicios como Azure Key Vault. Para obtener más información, vea ¿Qué son las identidades administradas para recursos de Azure? y Acerca de Azure Key Vault. Ejecute el código siguiente:

PowerShell

```
# Revise variables with your actual values.
$resourceGroup = "ResourceGroupName"
$automationAccount = "AutomationAccountName"
$vaultName = "KeyVaultName"
$keyName = "KeyName"

Set-AzAutomationAccount `
```

```
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount `
    -AssignSystemIdentity
```

La salida debe tener una apariencia similar a la siguiente:

```
SubscriptionId        : ContosoID
ResourceGroupName     : ContosoLab
AutomationAccountName : ContosoAutomation
Location              : westus
State                 : Ok
Plan                  : Basic
CreationTime          : 6/24/2021 5:02:33 PM +00:00
LastModifiedTime      : 6/25/2021 7:32:49 PM +00:00
LastModifiedBy        :
Tags                  : {}
Identity              : Microsoft.Azure.Management.Automation.Models.Identity
Encryption            : Microsoft.Azure.Management.Automation.Models.EncryptionProperties
PublicNetworkAccess   :
```

Obtenga el valor `PrincipalId` para usarlo más adelante. Ejecute el código siguiente:

PowerShell

```
$principalID = (Get-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount).Identity.PrincipalId

$principalID
```

Uso de REST

Configure una identidad administrada asignada por el sistema para la cuenta de Automation mediante la siguiente llamada API REST:

HTTP

```
PATCH https://management.azure.com/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e/resourceGroups/resource-group-name/providers/Microsoft.Automation/automationAccounts/automation-account-name?api-version=2020-01-13-preview
```

Cuerpo de la solicitud:

JSON

```
{
 "identity":
 {
  "type": "SystemAssigned"
  }
}
```

La identidad asignada por el sistema para la cuenta de Automation se devuelve en una respuesta similar a la siguiente:

JSON

```
{
 "name": "automation-account-name",
 "id": "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e/resourceGroups/resource-group-name/providers/Microsoft.Automation/automationAccounts/automation-account-name",
 ..
 "identity": {
    "type": "SystemAssigned",
    "principalId": "aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa",
    "tenantId": "bbbbbbbb-bbbb-bbbb-bbbb-bbbbbbbbbbbb"
 },
..
}
```

Configuración de la directiva de acceso de Key Vault

Una vez que se asigna una identidad administrada asignada por el sistema a la cuenta de Automation, se configura el acceso al almacén de claves que contiene las claves administradas por el cliente. Azure Automation necesita los permisos de operación **Get**, **Recover**, **WrapKey** y **UnwrapKey** para que la identidad acceda a las claves administradas por el cliente.

Mediante PowerShell

Use el cmdlet Set-AzKeyVaultAccessPolicy de PowerShell para conceder los permisos necesarios. Después, use Add-AzKeyVaultKey para crear una clave en el almacén de claves. Ejecute el código siguiente:

PowerShell

```
Set-AzKeyVaultAccessPolicy `
    -VaultName $vaultName `
    -ObjectId $principalID `
    -PermissionsToKeys Get, Recover, UnwrapKey, WrapKey

Add-AzKeyVaultKey `
    -VaultName $vaultName `
    -Name $keyName `
    -Destination 'Software'
```

La salida debe tener una apariencia similar a la siguiente:

```
Vault/HSM Name : contosovault
Name           : contosokey
Key Type       : RSA
Key Size       : 2048
Version        : 136065337ca7446
Id             : https://contosovault.azure.net:443/keys/contosoKey/12345

Enabled        : True
Expires        :
Not Before     :
Created        : 6/25/2021 8:50:06 PM
Updated        : 6/25/2021 8:50:06 PM
Recovery Level : Recoverable
Tags           :
```

Uso de REST

La directiva de acceso se puede establecer mediante la siguiente llamada API REST:

HTTP

```http
PUT https://management.azure.com/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e/resourceGroups/sample-group/providers/Microsoft.KeyVault/vaults/sample-vault/accessPolicies/add?api-version=2018-02-14
```

Cuerpo de la solicitud:

JSON

```json
{
  "properties": {
    "accessPolicies": [
      {
        "tenantId": "bbbbbbbb-bbbb-bbbb-bbbb-bbbbbbbbbbbb",
        "objectId": "aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa",
        "permissions": {
          "keys": [
            "get",
            "recover",
            "wrapKey",
            "unwrapKey"
          ],
          "secrets": [],
          "certificates": []
        }
      }
    ]
  }
}
```

ⓘ **Nota:**

Se deben proporcionar los campos **tenantId** y **objectId** con valores de **identity.tenantId** y **identity.principalId** respectivamente de la respuesta de la identidad administrada para la cuenta de Automation.

Nueva configuración de la cuenta de Automation para usar la clave administrada por el cliente

Si quiere cambiar la cuenta de Automation de claves administradas por Microsoft a claves administradas por el cliente, puede hacerlo mediante Azure PowerShell o con una plantilla de Azure Resource Manager.

Mediante PowerShell

Use el cmdlet Set-AzAutomationAccount de PowerShell para volver a configurar la cuenta de Automation para que use claves administradas por el cliente.

PowerShell

```
$vaultURI = (Get-AzKeyVault -VaultName $vaultName).VaultUri
$keyVersion = (Get-AzKeyVaultKey -VaultName $vaultName -KeyName $keyName).Version

Set-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount `
    -AssignSystemIdentity `
    -KeyName $keyName `
    -KeyVaultUri $vaultURI `
    -KeyVersion $keyVersion `
    -KeyVaultEncryption
```

Puede comprobar el cambio si ejecuta el siguiente comando:

PowerShell

```
(Get-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount).Encryption `
    |  ConvertTo-Json
```

La salida debe tener una apariencia similar a la siguiente:

```
{
    "KeyVaultProperties":  {
                                "KeyvaultUri":  "https://contosovault.vault.azure.net/",
                                "KeyName":  "contosoKey
                                "KeyVersion":  "6df37485 09281153"
                            },
    "KeySource":  1,
    "Identity":  {
                    "UserAssignedIdentity":  null
                 }
}
```

Uso de REST

Use la siguiente llamada API REST:

HTTP

```
PATCH https://management.azure.com/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e/resourceGroups/resource-group-name/providers/Microsoft.Automation/automationAccounts/automation-account-name?api-version=2020-01-13-preview
```

Cuerpo de la solicitud:

JSON

```
{
    "identity": {
    "type": "SystemAssigned"
    },
    "properties": {
        "encryption": {
            "keySource": "Microsoft.Keyvault",
            "keyvaultProperties": {
                "keyName": "sample-vault-key",
                "keyvaultUri": "https://sample-vault-key12.vault.azure.net",
                "keyVersion": "7c73556c521340209371eaf623cc099d"
            }
        }
    }
}
```

Respuesta de muestra

JSON

```
{
  "name": "automation-account-name",
  "id": "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e/resourceGroups/resource-group-name/providers/Microsoft.Automation/automationAccounts/automation-account-name",
  ..
```

```
  "properties": {
    ..
    "encryption": {
      "keyvaultProperties": {
         "keyName": "sample-vault-key",
          "keyvaultUri": "https://sample-vault-key12.vault.azure.net",
          "keyVersion": "7c73556c521340209371eaf623cc099d"
      },
      "keySource": "Microsoft.Keyvault"
    },
    ..
  }
}
```

Rotación de una clave administrada por el cliente

Las claves administradas por el cliente se pueden rotar en Azure Key Vault según las directivas de cumplimiento. Cuando la clave rota, hay que actualizar la cuenta de Automation para usar el nuevo identificador URI de la clave.

El giro de la clave no desencadena un nuevo cifrado de los recursos seguros en la cuenta de Automation. No se necesita ninguna acción adicional.

Revocación del acceso de una clave administrada por el cliente

Para revocar el acceso a las claves administradas por el cliente, use PowerShell o la CLI de Azure. Para más información, consulte la referencia de PowerShell para Azure Key Vault o la referencia de la CLI para Azure Key Vault. La revocación del acceso bloquea de manera eficaz el acceso a todos los recursos seguros de la cuenta de Automation, ya que Azure Automation no puede acceder a la clave de cifrado.

Pasos siguientes

- Para obtener información sobre las directrices de seguridad, consulte Procedimientos recomendados de seguridad en Azure Automation.
- Para comprender Azure Key Vault, vea ¿Qué es Azure Key Vault?.
- Para trabajar con certificados, vea Administración de certificados en Azure Automation.
- Para manipular las credenciales, vea Administración de credenciales en Azure Automation.
- Para usar variables de automatización, Administración de variables en Azure Automation.
- Para obtener ayuda al trabajar con conexiones, vea Administración de conexiones en Azure Automation.

Last updated on 17/11/2025

Administración de datos de Azure Automation

Este artículo contiene varios temas que explican cómo se protegen y protegen los datos en un entorno de Azure Automation.

TLS para Azure Automation

Para garantizar la seguridad de los datos en tránsito a Azure Automation, le recomendamos encarecidamente que configure el uso de seguridad de la capa de transporte (TLS). A continuación se muestra una lista de métodos o clientes que se comunican a través de HTTPS con el servicio Automation:

- Llamadas de webhook
- Trabajadores híbridos de Runbook de usuario (basados en extensiones y en agentes)
- Máquinas gestionadas por Azure Automation Update Management y Azure Automation Change & tracking inventory
- nodos de DSC de Azure Automation

Las versiones anteriores de TLS/Capa de sockets seguros (SSL) han demostrado ser vulnerables y, si bien todavía funcionan para permitir la compatibilidad con versiones anteriores, **no se recomiendan**. No se recomienda establecer explícitamente el agente para que solo use TLS 1.2, a menos que sea necesario, ya que esto puede interrumpir las características de seguridad a nivel de la plataforma que le permiten detectar y aprovechar automáticamente las ventajas de los protocolos más seguros más recientes a medida que estén disponibles, como TLS 1.3.

Para obtener información sobre la compatibilidad de TLS con el agente de Log Analytics para Windows y Linux, que es una dependencia del rol Hybrid Runbook Worker, consulte Log Analytics agent overview - TLS.

Actualización del protocolo TLS para las llamadas a instancias de Hybrid Worker y Webhook

Desde **el 1 de marzo de 2025**, todos los Runbook Workers híbridos de usuario basados en agentes y en extensiones, webhooks, nodos DSC, las máquinas gestionadas por Azure

Automation para la administración de actualizaciones y el seguimiento de cambios, que usen los protocolos de Seguridad de la Capa de Transporte (TLS) 1.0 y 1.1, ya no podrán conectarse a Azure Automation. Se producirá un error en todos los trabajos que se ejecutan o programan en instancias de Hybrid Worker que usan los protocolos TLS 1.0 y 1.1.

Asegúrese de que las llamadas a Webhook que desencadenan runbooks navegan por TLS 1.2 o posterior. Obtenga información sobre cómo deshabilitar los protocolos TLS 1.0/1.1 en un trabajador híbrido de Windows y habilitar TLS 1.2 o superior en una máquina con Windows.

Para Linux Hybrid Worker, ejecute el siguiente script de Python para actualizar al protocolo TLS más reciente.

Python

```python
import os

# Path to the OpenSSL configuration file as per Linux distro
openssl_conf_path = "/etc/ssl/openssl.cnf"

# Open the configuration file for reading
with open(openssl_conf_path, "r") as f:
    openssl_conf = f.read()

# Check if a default TLS version is already defined
if "DEFAULT@SECLEVEL" in openssl_conf:
    # Update the default TLS version to TLS 1.2
    openssl_conf = openssl_conf.replace("CipherString = DEFAULT@SECLEVEL",
"CipherString = DEFAULT@SECLEVEL:TLSv1.2")

    # Open the configuration file for writing and write the updated version
    with open(openssl_conf_path, "w") as f:
        f.write(openssl_conf)

    # Restart any services that use OpenSSL to ensure that the new settings are
applied
    os.system("systemctl restart apache2")
    print("Default TLS version has been updated to TLS 1.2.")
else:
    # Add the default TLS version to the configuration file
    openssl_conf += """
    Options = PrioritizeChaCha,EnableMiddleboxCompat
    CipherString = DEFAULT@SECLEVEL:TLSv1.2
    MinProtocol = TLSv1.2
    """

    # Open the configuration file for writing and write the updated version
    with open(openssl_conf_path, "w") as f:
        f.write(openssl_conf)

    # Restart any services that use OpenSSL to ensure that the new settings are
```

```
applied
    os.system("systemctl restart apache2")
    print("Default TLS version has been added as TLS 1.2.")
```

Guía específica para la plataforma

> **Nota:**
>
> Windows Server 2008, 2008 R2, 2012 y 2012 R2 alcanzaron el fin del soporte (EOS). Revise el uso y planee las actualizaciones y migraciones del sistema operativo en consecuencia. Para obtener más información, consulte Fin del soporte técnico para:
>
> - Windows Server 2008 y Windows Server 2008 R2
> - Windows Server 2012
> - Windows Server 2012 R2
>
> Actualización local a Windows Server 2016, 2019, 2022 o 2025.

Expandir tabla

Plataforma/lenguaje	Soporte técnico	Más información
Linux	Las distribuciones de Linux tienden a basarse en OpenSSL para la compatibilidad con TLS 1.2.	Compruebe el registro de cambios de OpenSSL para confirmar si su versión de OpenSSL es compatible.
Windows 8.0 - 10	Soportado y habilitado por defecto.	Para confirmar que aún usa la configuración predeterminada.
Windows Server 2012 - 2016	Soportado y habilitado por defecto.	Para confirmar que aún usa la configuración predeterminada
Windows 7 SP1 y Windows Server 2008 R2 SP1	Compatible, pero no habilitado de manera predeterminada.	Consulte la página Configuración del registro de TLS para obtener más información sobre cómo se habilita.

Retención de datos

Al eliminar un recurso en Azure Automation, se conserva durante muchos días con fines de auditoría antes de la eliminación permanente. Durante este tiempo, no es posible ver ni usar el recurso. Esta directiva también se aplica a los recursos que pertenecen a una cuenta de Automation eliminada. La directiva de retención se aplica a todos los usuarios y, por el momento, no se puede personalizar. Sin embargo, si necesita mantener los datos durante un período más

largo, puede reenviar los datos de trabajos de Azure Automation a los registros de Azure Monitor.

La tabla siguiente resume la directiva de retención para distintos recursos.

Expandir tabla

Datos	Directiva
Cuentas	Una cuenta se elimina de manera permanente 30 días después de que un usuario la elimine.
Activos	Un recurso se elimina de forma permanente 30 días después de que un usuario lo elimina o 30 días después de que un usuario elimina la cuenta que contiene el recurso. Los recursos incluyen variables, programaciones, credenciales, certificados, Python 2 paquetes y conexiones.
Nodos de DSC	Un nodo DSC se elimina permanentemente 30 días después de ser anulado el registro de una cuenta de Automation usando el portal de Azure o el cmdlet Unregister-AzAutomationDscNode en Windows PowerShell. Los nodos también se eliminan de forma permanente 30 días después de que el usuario elimine la cuenta que contiene el nodo.
Trabajos	Un trabajo se elimina de forma permanente 30 días después de cualquier modificación, por ejemplo, después de que el trabajo se complete, detenga o suspenda.
Módulos	Un módulo se elimina de forma permanente 30 días después de que un usuario lo elimina o 30 días después de que un usuario elimina la cuenta que contiene el módulo.
Configuraciones de nodo y archivos MOF	La configuración de nodo anterior se elimina de forma permanente 30 días después de que se genere una nueva configuración.
Informes de nodo	Un informe de nodo se elimina de forma permanente 90 días después de que se genere un nuevo informe para ese nodo.
Runbooks	Un runbook se elimina de forma permanente 30 días después de que un usuario lo elimina o 30 días después de que un usuario elimina la cuenta que contiene el recurso[1].

[1]El runbook puede recuperarse dentro de la ventana de 30 días presentando un incidente de soporte de Azure ante Microsoft Azure Support. Vaya al sitio Soporte técnico de Azure y seleccione **Submitar una solicitud de soporte técnico**.

Copia de seguridad de datos

Al eliminar una cuenta de Automation en Azure, se eliminan todos los objetos de la cuenta. Entre los objetos se incluyen runbooks, módulos, configuraciones, ajustes, trabajos y activos. Puede recuperar una cuenta de Automation eliminada en un plazo de 30 días. También puede usar la información siguiente para crear una copia de seguridad de los contenidos de la cuenta de Automation antes de eliminarla:

Runbooks

Puede exportar los runbooks a archivos de script mediante el portal de Azure o el cmdlet Get-AzAutomationRunbookContent en Windows PowerShell. Puede importar estos archivos de script en otra cuenta de Automation, como se describe en Administrar runbooks en Azure Automation.

Módulos de integración

No se pueden exportar módulos de integración desde Azure Automation, deben estar disponibles fuera de la cuenta de Automation.

Activos

No se pueden exportar recursos Azure Automation: certificados, conexiones, credenciales, programaciones y variables. En su lugar, puede usar el portal de Azure y Azure cmdlets para anotar los detalles de estos recursos. Posteriormente, puede usar esos detalles para crear todos los recursos utilizados por los runbooks que importe a otra cuenta de Automation.

No es posible recuperar el valor de variables cifradas o del campo de contraseña de las credenciales mediante cmdlets. Si no conoce estos valores, puede recuperarlos en un runbook (libro de operaciones). Para recuperar valores de variable, consulte Activosvariables en Azure Automation. Para obtener más información sobre cómo recuperar valores de credenciales, consulte Recursos de credenciales en Azure Automation.

Configuraciones DSC

Puede exportar las configuraciones de DSC a archivos de script mediante el portal de Azure o el cmdlet Export-AzAutomationDscConfiguration en Windows PowerShell. Puede importar y usar estas configuraciones en otra cuenta de Automation.

Ubicación de datos

Especifique una región durante la creación de una cuenta de Azure Automation. Los datos de servicio, como los recursos, la configuración y los registros, se almacenan en esa región y pueden

transitar o procesarse en otras regiones dentro de la misma geografía. Estos puntos de conexión globales son necesarios para proporcionar una experiencia de alto rendimiento y baja latencia a los usuarios finales, independientemente de su ubicación. Solo para la región Sur de Brasil (Estado de Sao Paulo) de la geografía de Brasil, la región sudeste de Asia (Singapur) y la región Este de Asia (Hongkong) de la geografía de Asia Pacífico, almacenamos Azure Automation datos en la misma región para dar cabida a los requisitos de residencia de datos para estas regiones.

Pasos siguientes

- Para obtener información sobre las directrices de seguridad, consulte Prácticas recomendadas de seguridad en Azure Automation.
- Para obtener más información sobre los recursos seguros en Azure Automation, consulte Cifrado de recursos seguros en Azure Automation.
- Para más información sobre la replicación geográfica, consulte Creación y uso de la replicación geográfica activa.

Last updated on 05/05/2026

Confiabilidad en Azure Automation

Azure Automation es un servicio que ejecuta tareas de administración en su nombre. Puede definir scripts, *denominados runbooks*, que desea ejecutar y Azure Automation proporciona la infraestructura para ejecutar esos scripts. Este artículo se centra en la automatización de procesos, que es la funcionalidad principal del servicio. Los trabajadores de runbooks híbridos, que se ejecutan en infraestructura administrada por el cliente, quedan fuera del alcance de este artículo.

Cuando se usa Azure, la confiabilidad es una responsabilidad compartida. Microsoft proporciona una variedad de capacidades para apoyar la resiliencia y la recuperación. Es responsable de comprender cómo funcionan esas funcionalidades dentro de todos los servicios que usa y de seleccionar las funcionalidades que necesita para cumplir los objetivos empresariales y los objetivos de tiempo de actividad.

En este artículo se describe cómo hacer que Azure Automation sea resistente a diversas interrupciones y problemas potenciales, incluidos errores transitorios, interrupciones de zonas de disponibilidad, interrupciones regionales y mantenimiento del servicio. También se describen las opciones de copia de seguridad y restauración, así como información clave sobre el acuerdo de nivel de servicio (SLA) de Azure Automation.

Recomendaciones de implementación de producción para la confiabilidad

Para cargas de trabajo de producción que usan la automatización de procesos, siga estas recomendaciones:

- ✓ Controle los errores transitorios que se producen cuando los runbooks interactúan con Azure servicios y API mediante la adición de la lógica de reintento adecuada a los scripts.
- ✓ Diseña tus runbooks para que sean resistentes a las interrupciones. Use puntos de control para mantener el progreso entre los reinicios del trabajo y, si necesita almacenar el estado, use el almacenamiento externo.

Introducción a la arquitectura de confiabilidad

En esta sección se describen algunos de los aspectos importantes de cómo funciona el servicio que es más relevante desde una perspectiva de confiabilidad. En la sección se presenta la arquitectura lógica, que incluye algunos de los recursos y características que se implementan y usan. También se describe la arquitectura física, que proporciona detalles sobre cómo funciona el servicio en segundo plano.

Arquitectura lógica

Al implementar Azure Automation, se crea una *cuenta de automation*, que es un contenedor lógico para los recursos que ejecutan la automatización.

- *Runbooks*, que representan el trabajo que se va a realizar. *Los runbooks textuales* son scripts escritos en PowerShell o Python. *Los runbooks gráficos* se crean mediante un editor gráfico.
- Recursos que comparten runbooks, incluidos módulos, conexiones, credenciales, certificados y variables.
- Recursos que inician la ejecución de los runbooks, incluidas las programaciones y los observadores.

Para obtener más información sobre estos recursos, consulte Ejecución de runbooks en Azure Automation.

En este artículo se tratan la confiabilidad y resistencia de estas funcionalidades, que forman parte de la *automatización de procesos* en Azure Automation.

Arquitectura física

Los runbooks se ejecutan en una infraestructura informática. Existen dos modelos de implementación para la automatización de procesos:

- Trabajos en la nube **(Microsoft administrados):** de forma predeterminada, los runbooks se ejecutan en la infraestructura de nube proporcionada por Microsoft. Microsoft es responsable de la alta disponibilidad y administración de esta infraestructura. Al enviar un trabajo de runbook, Azure Automation asigna un trabajo en la nube desde su grupo de recursos de proceso disponibles, ejecuta el runbook y, a continuación, devuelve el recurso al grupo.

  A veces, es posible que los trabajos en la nube se interrumpan mientras se ejecutan. Diseñe los runbooks en la suposición de que un trabajo podría reiniciarse en una infraestructura diferente y que los datos escritos en el almacenamiento temporal en la instancia anterior ya no son accesibles.

- **Hybrid Runbook Workers:** Opcionalmente, puede configurar su propia infraestructura de proceso (máquinas virtuales en Azure, otras nubes o locales) para ejecutar runbooks. Cuando utilice trabajadores de runbooks híbridos, es su responsabilidad configurarlos para que cumplan sus requisitos de fiabilidad. Los trabajadores de runbooks híbridos quedan fuera del alcance de este artículo.

Resistencia a errores transitorios

Los errores transitorios son errores breves e intermitentes en los componentes. Se producen con frecuencia en un entorno distribuido como la nube y son una parte normal de las operaciones. Los errores transitorios se corrigen después de un breve período de tiempo. Es importante que las aplicaciones puedan controlar errores transitorios, normalmente mediante el reintento de solicitudes afectadas.

Todas las aplicaciones hospedadas en la nube deben seguir las instrucciones de control de errores transitorios de Azure cuando se comunican con cualquier API, bases de datos y otros componentes hospedados en la nube. Para obtener más información, consulte Recomendaciones para controlar errores transitorios.

Eres responsable de escribir guías operativas que gestionen errores transitorios en los servicios y las API con los que interactúan. En el caso de los runbooks textuales, implemente la lógica de reintento mediante bucles y control de errores. Para obtener instrucciones y ejemplos, consulte Control de errores transitorios en un script dependiente del tiempo. En el caso de los runbooks gráficos, configure el comportamiento de reintento para las actividades de su flujo de trabajo. Para obtener detalles sobre la configuración, consulte Reintentar una actividad en runbooks gráficos.

El mantenimiento de la infraestructura u otros eventos de la plataforma pueden interrumpir los trabajos de los runbooks. Diseñe los runbooks para controlar estas interrupciones:

- **Implementar puntos de control.** En el caso de los runbooks de flujos de trabajo de PowerShell, utiliza puntos de control para guardar el progreso en momentos clave del flujo de trabajo. Si un trabajo se interrumpe y se reinicia, puede reanudarse desde el último punto de control en lugar de volver a empezar. Para obtener más información, consulte Uso de puntos de control en un flujo de trabajo.

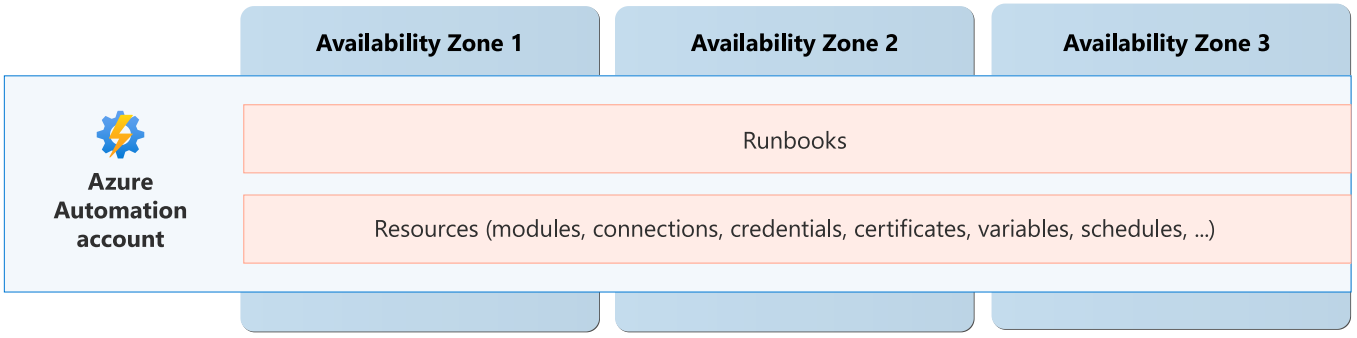
- **Comprenda los límites de las tareas.** Los trabajos en la nube tienen límites de uso equitativo en cuanto a la duración de la ejecución. Para obtener más información sobre los límites de ejecución de los trabajos y cómo se aplican, consulta Ejecución de runbooks.

- **Almacenar el estado persistente externamente.** Los trabajos no mantienen el estado entre ejecuciones. Si se interrumpe un trabajo y se reinicia en otra instancia, es posible que se pierda algo escrito en el almacenamiento temporal por la primera ejecución. Si necesita conservar los datos entre ejecuciones de trabajo, almacénelo en almacenamiento externo, como Azure Blob Storage o una base de datos.

Resistencia a errores de zona de disponibilidad

Las zonas de disponibilidad son grupos físicamente independientes de centros de datos dentro de una región de Azure. Cuando una zona falla, los servicios pueden transferirse a una de las zonas restantes.

En las regiones admitidas, las cuentas de Automation y los trabajos en la nube tienen *redundancia de zona*, lo que significa que el servicio distribuye los recursos entre varias zonas de disponibilidad. Microsoft habilita automáticamente la redundancia de zona y no requiere ninguna configuración.



Requisitos

Compatibilidad regional: Al implementar una cuenta de automatización en una de las siguientes regiones, pasa a tener redundancia de zona automáticamente:

Expandir tabla

Americas	Europe	Oriente Medio	Africa	Asia Pacífico
Brazil South	France Central	Israel Central	Norte de Sudáfrica	Australia East
Canada Central	Centro-oeste de Alemania	Qatar Central		Central India
Central US	Norte de Italia			Norte de

Americas	Europe	Oriente Medio	Africa	Asia Pacífico
				China 3
Este de EE. UU.	Norte de Europa			East Asia
Este de EE. UU. 2	Norway East			Japan East
Centro-sur de EE. UU.	Poland Central			Korea Central
Gobierno de EE.UU. de Virginia	Sweden Central			Southeast Asia
Oeste de EE. UU. 2	UK South			
Oeste de EE. UU. 3	West Europe			

Para obtener la lista actual de regiones admitidas, consulte Compatibilidad con zonas de disponibilidad para Azure Automation.

Costo

No hay ningún cargo adicional por la redundancia de zona. En el caso de la automatización de procesos, la facturación se basa en el tiempo que se ejecutan los trabajos y los observadores. Para obtener más información, consulte precios de Azure Automation.

Configurar soporte de zonas de disponibilidad

Cuando se crea una cuenta de automatización en una región compatible, esta es automáticamente redundante por zona. No se puede deshabilitar la redundancia de zona. Para obtener más información, consulte Compatibilidad con zonas de disponibilidad para Azure Automation.

Comportamiento cuando todas las zonas están en buen estado

En esta sección se describe qué cabe esperar cuando la cuenta de automatización es redundante por zona y todas las zonas de disponibilidad de la región están operativas.

- **Operación entre zonas:** Las operaciones de administración de cuentas de Automation y los trabajos en la nube se distribuyen automáticamente entre zonas de disponibilidad en la región. Una solicitud o un trabajo pueden ser atendidos por cualquier instancia de cualquier zona de disponibilidad.

- **Replicación de datos entre zonas:** La configuración de la cuenta de Automation, los scripts de runbook y otros recursos que se implementan en la cuenta de Automation se replican sincrónicamente en varias zonas de disponibilidad.

Comportamiento durante un fallo de zona

En esta sección se describe qué cabe esperar cuando su cuenta de automatización es redundante por zona y se produce una interrupción en una de las zonas de disponibilidad de la región.

- **Detección y respuesta:** La plataforma Azure Automation es responsable de detectar un error en una zona de disponibilidad. No es necesario hacer nada para iniciar una conmutación por error de la zona.

- **Notificación:** Microsoft no le notifica automáticamente cuando una zona está inactiva. Sin embargo, puede usar Azure Service Health para comprender el estado general del servicio, incluidos los errores de zona, y puede configurar alertas de Service Health para notificarle problemas.

- **Solicitudes activas:** Cualquier ejecución de trabajo en curso en la zona que no funciona correctamente podría interrumpirse. Azure Automation inicia automáticamente una nueva ejecución de un trabajo utilizando infraestructura de zonas en buen estado. Diseñe los runbooks para que sean resistentes a errores transitorios e interrupciones para que puedan reiniciarse de forma segura.

- **Pérdida de datos prevista:** Las ejecuciones de trabajos no conservan el estado, por lo que no se espera que un error de zona provoque la pérdida de datos de los trabajos en curso. Si un trabajo necesita almacenar los datos que puede usar para recuperarse de la interrupción, como puntos de control, almacene esa información en un servicio de almacenamiento en la nube persistente, como Azure Storage o una base de datos.

  La configuración de la cuenta de Automation y los datos de runbook se replican entre zonas y permanecen accesibles incluso cuando una zona no está disponible.

- **Tiempo de inactividad esperado:** Durante una interrupción de zona, la cuenta de Automation puede experimentar una breve interrupción mientras el servicio detecta el error y redistribuye la carga de trabajo a zonas correctas.

- **Redistribución:** El servicio reequilibra automáticamente la capacidad entre las zonas restantes que funcionan correctamente. Las nuevas ejecuciones de trabajos, los observadores y las programaciones continúan ejecutándose en la infraestructura de las

zonas que funcionan correctamente. La recuperación no depende de que la zona que ha fallado vuelva a estar operativa.

Recuperación de zona

Cuando una zona con error vuelve al servicio, Azure Automation vuelve a reinsertarla automáticamente en la rotación de zona. No se requiere ninguna acción de usted. El servicio supervisa el estado de la zona y redistribuye la carga de trabajo en todas las zonas a medida que se reanudan las operaciones normales.

Prueba de fallos de zona

Azure Automation administra el enrutamiento del tráfico, la conmutación por error y la recuperación de zona para los recursos con redundancia de zona. No es necesario iniciar ninguna acción ni validar los procesos ante fallos de la zona de disponibilidad. Ponga a prueba sus manuales de procedimientos para confirmar que resisten las interrupciones.

Resistencia a errores en toda la región

Azure Automation es un servicio de una sola región. Si la región deja de estar disponible, la cuenta de Automation tampoco está disponible.

Soluciones personalizadas de varias regiones para la resistencia

Puede implementar cuentas de Automation independientes en varias regiones y cambiar entre ellas cuando sea necesario. Usted es responsable de desplegar las cuentas en cada región, configurarlas de forma adecuada, distribuir las solicitudes entre las cuentas y gestionar la conmutación por error si una región no está disponible. Para obtener información detallada sobre los enfoques que puede tener en cuenta, consulte Recuperación ante desastres para Azure Automation.

Copias de seguridad y restauración

Para la mayoría de las soluciones, no debe confiar exclusivamente en copias de seguridad. En su lugar, utilice las otras capacidades descritas en esta guía para apoyar los requisitos de resiliencia. Sin embargo, las copias de seguridad protegen contra algunos riesgos que otros enfoques no. Para más información, consulte ¿Qué son la redundancia, la replicación y la copia de seguridad?.

Azure Automation no proporciona copias de seguridad integradas para la configuración de la cuenta de Automation ni para el contenido del runbook. Mantenga sus propias copias fuera del servicio para que pueda volver a implementarlas si es necesario.

- **Utiliza infraestructura como código (IaC) para la configuración de la cuenta de automatización.** Defina cuentas de automatización y recursos relacionados en archivos de Bicep, plantillas de ARM o Terraform. Almacene las plantillas en el control de versiones y use su pipeline de implementación para recrear el entorno en la misma región o en otra. Incluya certificados, variables, programaciones y referencias de credenciales en sus artefactos y procesos de despliegue. Almacene secretos en servicios como Azure Key Vault en lugar de insertar valores directamente en el código de runbook.

- **Almacene los scripts del runbook en un sistema de control de código fuente.** Mantenga el origen de PowerShell y Python runbooks en un sistema de control de código fuente como Git. Utiliza el control de versiones, la creación de ramas y la revisión de solicitudes de incorporación de cambios para proteger la calidad de los scripts y permitir la reversión a versiones que se sabe que funcionan correctamente.

- **Haga una copia de seguridad del estado desde el almacén de datos correspondiente.** Los trabajos no conservan el estado. Si necesita mantener registros de trabajos detallados o cualquier otro dato que generen los runbooks, almacénelo en otro servicio de almacenamiento o base de datos de Azure y realice una copia de seguridad desde allí.

Resistencia a la eliminación accidental

Si elimina accidentalmente una cuenta de Automation, es posible que pueda restaurarla dentro de un período de tiempo limitado. Para más información, consulte Restauración de una cuenta de Automation eliminada.

Resistencia al mantenimiento del servicio

Microsoft aplica periódicamente actualizaciones de servicio y realiza otro mantenimiento. La plataforma Azure controla estas actividades automáticamente, lo que garantiza que el mantenimiento sea transparente y sin problemas. No se prevé ningún tiempo de inactividad durante las operaciones de mantenimiento, a menos que se te haya informado a través de el mantenimiento planificado de Azure Service Health.

Acuerdo de nivel de servicio

El acuerdo de nivel de servicio (SLA) para Azure servicios describe la disponibilidad esperada de cada servicio y las condiciones que la solución debe cumplir para lograr esa expectativa de disponibilidad. Para obtener más información, consulte Acuerdos de Nivel de Servicio para servicios en línea.

Contenido relacionado

- Confiabilidad en Azure
- Recuperación ante desastres para Azure Automation
- Administración de runbooks en Azure Automation
- Ejecución de runbooks en Azure Automation

ⓘ**Nota:** El autor ha creado este artículo con ayuda de inteligencia artificial. Más información

---

Last updated on 07/07/2026

Recuperación ante desastres para Azure Automation

Se aplica a: ✔ Máquinas virtuales Linux ✔ Máquinas virtuales Windows

Este artículo explica la estrategia de recuperación ante desastres para gestionar un fallo a nivel regional.

Debes contar con una estrategia de recuperación ante desastres para gestionar una falla de servicio a nivel regional. Una estrategia de recuperación ante desastres ayuda a reducir el impacto y los efectos derivados de eventos impredecibles en tu negocio y clientes. Eres responsable de configurar la recuperación ante desastres de cuentas de automatización y sus recursos dependientes, como módulos, conexiones, credenciales, certificados, variables y calendarios. Un aspecto importante de un plan de recuperación ante desastres es prepararse para realizar la conmutación por error a la réplica de la cuenta de Automatización creada previamente en la región secundaria, si la cuenta de Automatización de la región primaria deja de estar disponible. Asegúrese de que la estrategia de recuperación ante desastres tenga en cuenta la cuenta de Automation y los recursos dependientes.

Algunas regiones se combinan con otras para proporcionar protección frente a desastres regionales o geográficos de gran envergadura. Independientemente de si la región primaria tiene un par regional o no, la estrategia de recuperación ante desastres para la cuenta de Automation sigue siendo la misma.

Para saber más sobre la fiabilidad de Azure Automation, incluyendo alta disponibilidad a través del soporte de zonas de disponibilidad, consulta Fiabilidad en Azure Automation.

Habilitación de la recuperación ante desastres

Cada cuenta de Automation que cree requiere una ubicación que se debe usar para la implementación. Esta sería la región primaria de la cuenta de Automation e incluye activos, runbooks creados para la cuenta de Automation, datos de ejecución de trabajos y registros. Para la recuperación ante desastres, la cuenta de Automation de réplica debe estar ya implementada y lista en la región secundaria.

- Empiece por crear una cuenta de Automation de réplica en cualquier región alternativa.

- Seleccione la región secundaria de su elección: región emparejada o cualquier otra región en la que esté disponible Azure Automation.
- Además de crear una réplica de la cuenta de Automation, replique los recursos dependientes (como runbooks, módulos, conexiones, credenciales, certificados, variables, programaciones y permisos asignados para la cuenta de ejecución e identidades administradas) de la cuenta de Automation en la región primaria a la cuenta de Automation en la región secundaria. Puede usar el script de PowerShell para migrar recursos de la cuenta de Automation de una región a otra.
- Si utiliza plantillas de ARM para definir e implementar runbooks de Azure Automation, puede usar estas plantillas para implementar esos mismos runbooks en cualquier otra región de Azure en la que cree la cuenta de Azure Automation de réplica. En caso de una interrupción generalizada en toda la región primaria, puede ejecutar los runbooks replicados en la región secundaria para seguir operando con normalidad. Este enfoque garantiza que la región secundaria actúe para continuar el trabajo si la región primaria sufre una interrupción o fallo.

> **Nota:**
>
> Debido a los requisitos de residencia de datos, los datos de trabajos y los registros presentes en la región primaria no están disponibles en la región secundaria.

Escenarios para trabajos híbridos y en la nube

Escenario: Ejecución de trabajos en la nube en la región secundaria

En el caso de los trabajos en la nube, habría un tiempo de inactividad insignificante, siempre que una cuenta de Automation de réplica y todos los recursos y runbooks dependientes ya estén implementados y disponibles en la región secundaria. Puede usar la cuenta de réplica para ejecutar trabajos como de costumbre.

Escenario: Ejecución de trabajos en Hybrid Runbook Worker implementados en una región diferente de la región primaria de error

Si Hybrid Runbook Worker de Windows o Linux se implementa mediante el enfoque basado en extensiones en una región *diferente* de la región primaria de error, siga estos pasos para

continuar ejecutando los trabajos híbridos:

1. Elimine la extensión instalada en Hybrid Runbook Worker en la cuenta de Automation de la región primaria.
2. Agregue la misma instancia de Trabajo híbrido de runbook a un grupo de Hybrid Worker en la cuenta de Automatización de la región secundaria. La extensión de trabajador de Hybrid se instala en la máquina de la réplica de la cuenta de Automation.
3. Ejecute los trabajos en la instancia de Hybrid Runbook Worker creada en el paso 2.

Escenario: Ejecutar trabajos en Hybrid Runbook Worker implementado en la región primaria afectada por el fallo

Si Hybrid Runbook Worker se implementa en la región primaria y hay un error de proceso en esa región, la máquina no estará disponible para ejecutar trabajos de Automation. Debe aprovisionar una nueva máquina virtual en una región alternativa y registrarla como Hybrid Runbook Worker en la cuenta de Automation en la región secundaria.

- Consulte los pasos de instalación en cómo implementar un Hybrid Runbook Worker de usuario de Windows o Linux basado en extensiones.

Script para migrar recursos de cuenta de Automation de una región a otra

Puede utilizar estos scripts para migrar los recursos de una cuenta de Automation desde la cuenta de la región primaria hasta la cuenta de la región secundaria. Estos scripts se usan para migrar solo runbooks, módulos, conexiones, credenciales, certificados y variables. La ejecución de estos scripts no afecta a la cuenta de Automation ni a sus recursos presentes en la región primaria.

Requisitos previos

1. Asegúrese de que la cuenta de Automation en la región secundaria se crea y está disponible para que los recursos de la región primaria se puedan migrar a ella. Es preferible si la cuenta de automatización de destino es una sin recursos personalizados, ya que evita posibles conflictos de recursos debido al mismo nombre y pérdida de datos.

2. Asegúrese de que las identidades administradas asignadas por el sistema están habilitadas en la cuenta de Automation de la región primaria.

3. Asegúrese de que las identidades administradas asignadas por el sistema de la cuenta principal de Automatización tengan acceso de colaborador a la suscripción a la que pertenece.

4. Asegúrese de que la identidad administrada de la cuenta de Automation principal tenga acceso de colaborador con permisos de lectura y escritura a la cuenta de Automation en la región secundaria. Para habilitarla, proporcione los permisos necesarios para las identidades administradas de la cuenta secundaria de Automation. Más información.

5. Asegúrese de que el script tiene acceso a los recursos de la cuenta de Automation en la región primaria. Por lo tanto, debe ejecutarse como un runbook en esa cuenta de Automation para que la migración se realice correctamente.

6. Los módulos necesarios son:

    - Az.Accounts (versión 2.8.0)
    - Az.Resources (versión 6.0.0)
    - Az.Automation (versión 1.7.3)
    - Az.Storage (versión 4.6.0)

7. Asegúrese de que las cuentas de Automation de origen y destino pertenezcan al mismo inquilino de Microsoft Entra.

Cree y ejecute el libro de procedimientos

Puede usar el script de PowerShell o el runbook de flujo de trabajo de PowerShell o importarlo desde la galería de Runbook y ejecutarlo para habilitar la migración de recursos de una cuenta de Automation a otra.

Siga los pasos para importar y ejecutar el runbook:

Script de PowerShell

1. Inicie sesión en Azure Portal.
2. Ve a la cuenta de Automatización que quieres migrar a otra región.
3. En **Automatización de procesos**, seleccione **Runbooks**.
4. Seleccione **Examinar galería** y, en el cuadro de búsqueda, escriba *Migrar los recursos de una cuenta de Automation de una región a otra* y seleccione **script de PowerShell**.
5. En la página **Importar un runbook**, escriba un nombre para el runbook.
6. Seleccione **Versión del entorno de ejecución** como 5.1 o 7.1 (versión preliminar)

7. Escriba la descripción y seleccione **Importar**.
8. En la página **Editar runbook de PowerShell**, edite los parámetros necesarios y ejecútelo.

Puede elegir cualquiera de las opciones para editar y ejecutar el script. Puede proporcionar los siete parámetros obligatorios como se indica en la opción 1 **o** tres parámetros obligatorios proporcionados en la opción 2 para editar y ejecutar el script.

Las opciones son:

Opción 1

Expandir tabla

Nombre	Obligatorio	Descripción
SourceAutomationAccountName	Cierto	Nombre de la cuenta de Automation en la región primaria desde donde se deben migrar los recursos.
DestinationAutomationAccountName	Cierto	Nombre de la cuenta de Automation en la región secundaria a la que se deben migrar los recursos.
SourceResourceGroup	Cierto	Nombre del grupo de recursos de la cuenta de automatización en la región primaria.
DestinationResourceGroup	Cierto	Nombre del grupo de recursos de la cuenta de Automation en la región secundaria.
Id. de suscripción de origen	Cierto	Identificador de suscripción de la cuenta de Automation en la región primaria
DestinationSubscriptionId	Cierto	Identificador de suscripción de la cuenta de Automation en la región secundaria.
Tipo[]	Cierto	Matriz que consta de todos los tipos de recursos que deben migrarse, los valores posibles son Certificados, Conexiones, Credenciales, Módulos, Runbooks y Variables.

Limitaciones

- El script solo migra módulos PowerShell personalizados. No se migran los módulos predeterminados ni los paquetes de Python a la cuenta de Automation de la réplica.
- El script no migra las **Programaciones** ni las **Identidades administradas** presentes en la cuenta de Automation de la región primaria. Estos tendrían que crearse manualmente en la cuenta de Automation de réplica.
- Los datos de trabajos y los registros de actividad no se migrarían a la cuenta de réplica.

Pasos siguientes

- Confiabilidad en Azure Automation
- Regiones que admiten zonas de disponibilidad.

Last updated on 13/07/2026

Creación de una cuenta independiente de Azure Automation

En este artículo, se muestra cómo crear una cuenta de Azure Automation con Azure Portal. Puede usar la cuenta de Automation del portal para evaluar y obtener información acerca de Automation sin usar características de administración o integración adicionales con los registros de Azure Monitor. Puede agregar características de administración o realizar la integración con los registros de Azure Monitor para disfrutar de una supervisión avanzada de los trabajos de runbook en cualquier momento en el futuro.

Con una cuenta de Automation, podrá autenticar runbooks mediante la administración de recursos en Azure Resource Manager o en el modelo de implementación clásica. Una cuenta de Automation puede administrar los recursos de todas las regiones y suscripciones de un inquilino determinado.

Con esta cuenta que se ha creado automáticamente, podrá generar e implementar rápidamente runbooks que contribuyan a satisfacer sus necesidades de automatización.

Permisos necesarios para crear una cuenta de Automation

Para crear o actualizar una cuenta de Automation y completar las tareas que se describen en este artículo, debe tener los siguientes privilegios y permisos:

Para crear una cuenta de Automation, la cuenta de usuario de Microsoft Entra debe agregarse a un rol con permisos equivalentes a los del rol del propietario de los recursos de `Microsoft.Automation`. Para más información, consulte Control de acceso basado en rol en Azure Automation.

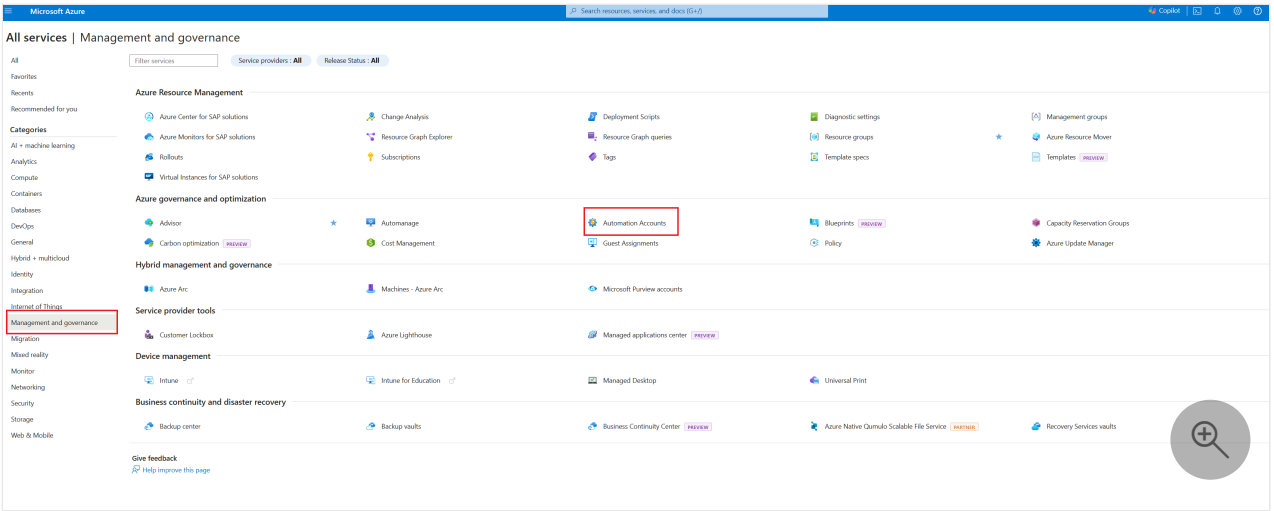
Creación de una nueva cuenta de Automation en Azure Portal

Para crear una cuenta de Azure Automation en Azure Portal, complete los pasos siguientes:

1. Inicie sesión en Azure Portal con una cuenta que sea miembro del rol Administradores de la suscripción y coadministrador de la suscripción.

2. Seleccione + **Crear un recurso**.

3. Busque **Automation**. En los resultados de la búsqueda, haga clic en **Automation**.



Las opciones de la nueva cuenta de Automation se organizan en pestañas en la **página Creación de una cuenta de Automation**. En las secciones siguientes se describe cada una de las pestañas y sus opciones.

Conceptos básicos

En la pestaña **Aspectos básicos**, proporcione la información esencial de la cuenta de Automation. Después de completar la pestaña **Aspectos básicos**, puede personalizar aún más la nueva cuenta de Automation; para ello, establezca las opciones en las otras pestañas, o bien seleccione **Revisar y crear** para aceptar las opciones predeterminadas y continuar con la validación y la creación de la cuenta.

> **Nota:**
>
> De forma predeterminada, una identidad administrada asignada por el sistema está habilitada para la cuenta de Automation.

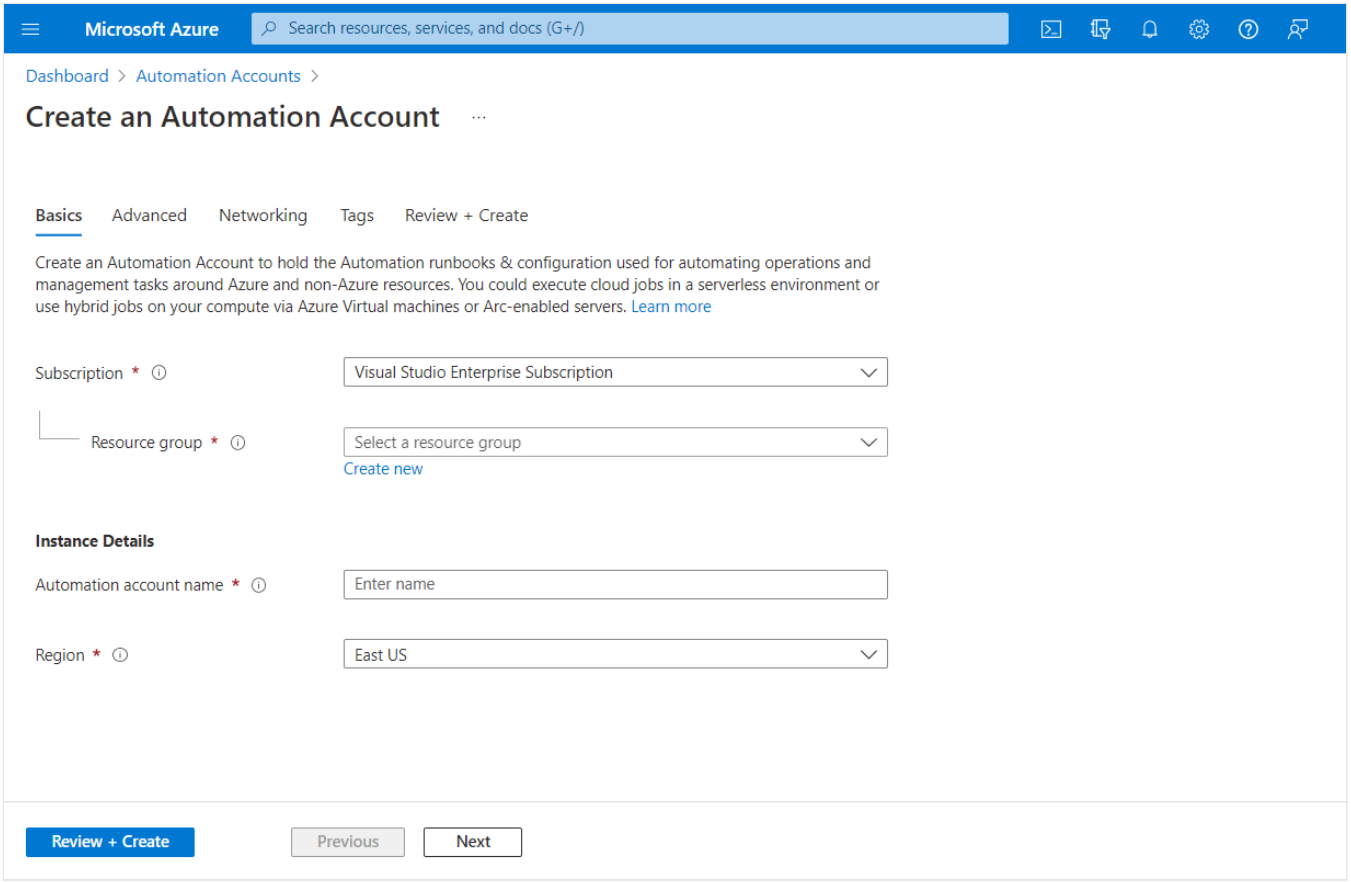
En la siguiente tabla se describen los campos de la pestaña **Aspectos básicos**.

Expandir tabla

Campo	Obligatorio o opcional	Descripción
Suscripción	Obligatorio	En la lista desplegable, seleccione la suscripción de Azure para la cuenta.
Grupo de recursos	Obligatorio	En la lista desplegable, seleccione el grupo de recursos existente o seleccione **Crear**.

Campo	Obligatorio o opcional	Descripción
Nombre de la cuenta de Automation	Obligatorio	Escriba un nombre único para su ubicación y grupo de recursos. Es posible que los nombres de las cuentas de Automation que se hayan eliminado no estén disponibles de inmediato. No se puede cambiar el nombre de la cuenta una vez que se ha escrito en la interfaz de usuario.
Región	Obligatorio	En la lista desplegable, seleccione una región para la cuenta. Para obtener una lista actualizada de ubicaciones en las que se puede implementar una cuenta de Automation, consulte Productos disponibles por región.

En la imagen siguiente se muestra una configuración estándar para una nueva cuenta de Automation.



Avanzado

En la pestaña **Opciones avanzadas**, puede configurar la opción de identidad administrada para la nueva cuenta de Automation. La opción de identidad administrada asignada por el usuario también se puede configurar después de crear la cuenta de Automation.

Para obtener instrucciones sobre cómo crear una identidad administrada asignada por el usuario, consulte Creación de una identidad administrada asignada por el usuario.

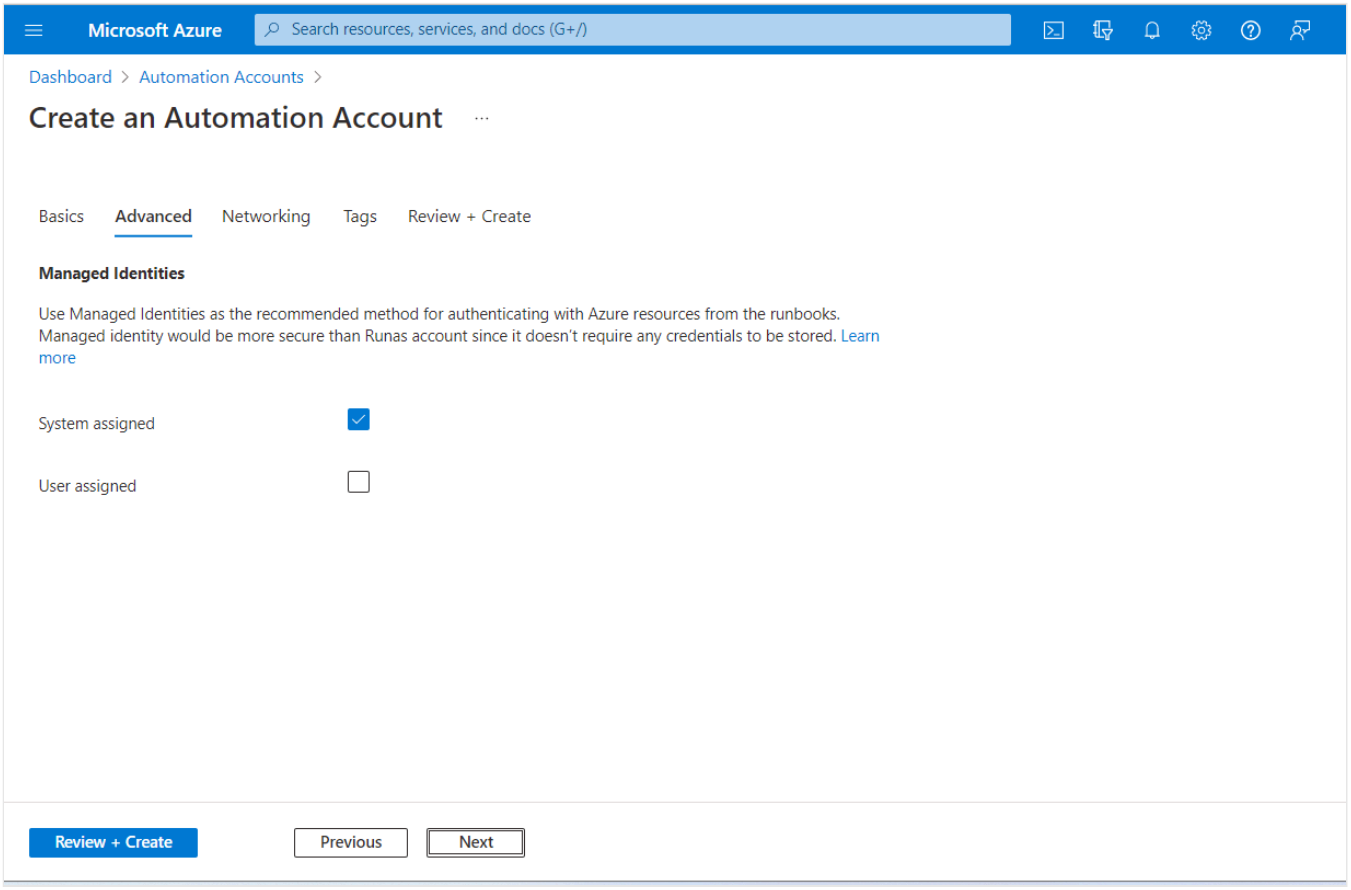
En la siguiente tabla se describen los campos de la ficha **Opciones avanzadas**.

Expandir tabla

Campo	Obligatorio o opcional	Descripción
Asignada por el sistema	Opcional	Una identidad de Microsoft Entra que esté vinculada al ciclo de vida de la cuenta de Automation.
Asignada por el usuario	Opcional	Una identidad administrada representada como un recurso de Azure independiente que se administra por separado de los recursos que la usan.

Puede optar por habilitar las identidades administradas más adelante y la cuenta de Automation se crea sin una. Para habilitar una identidad administrada después de crear la cuenta, consulte Habilitación de la identidad administrada. Si selecciona ambas opciones, para la identidad asignada por el usuario, seleccione la opción **Agregar identidades asignadas por el usuario**. En la página **Seleccionar una identidad administrada asignada por el usuario**, seleccione una suscripción y agregue una o varias identidades asignadas por el usuario creadas en esa suscripción para asignarlas a la cuenta de Automation.

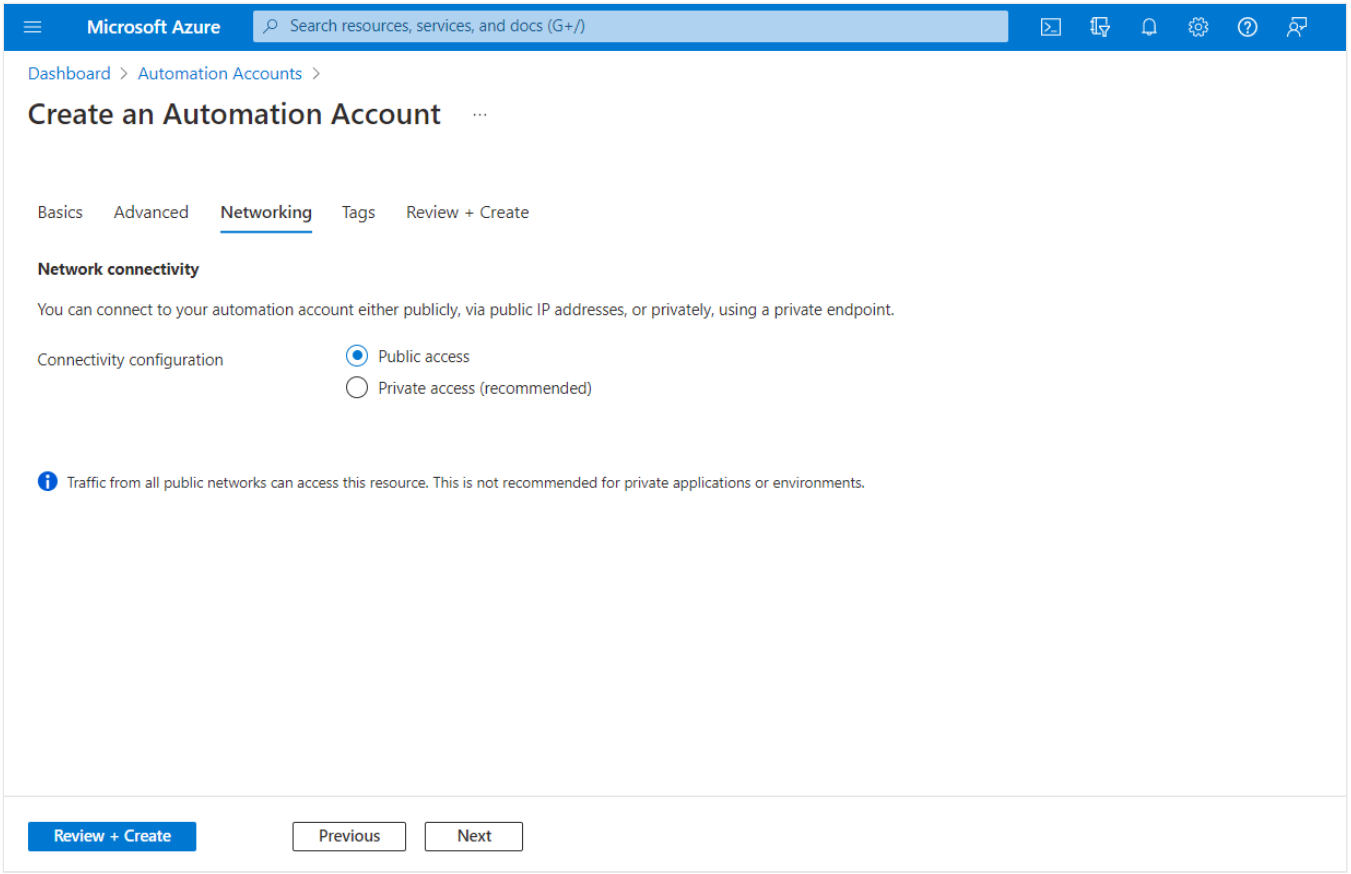
En la imagen siguiente se muestra una configuración estándar para una nueva cuenta de Automation.



Redes

En la pestaña **Redes**, puede configurar la conectividad con la cuenta de Automation, ya sea públicamente a través de IP públicas o de manera privada mediante un vínculo privado de Azure Automation. Private Link de Azure Automation conecta uno o más puntos de conexión privados (y, por tanto, las redes virtuales en las que se encuentran) al recurso de la cuenta de Automation.

En la imagen siguiente se muestra una configuración estándar para una nueva cuenta de Automation.



Etiquetas

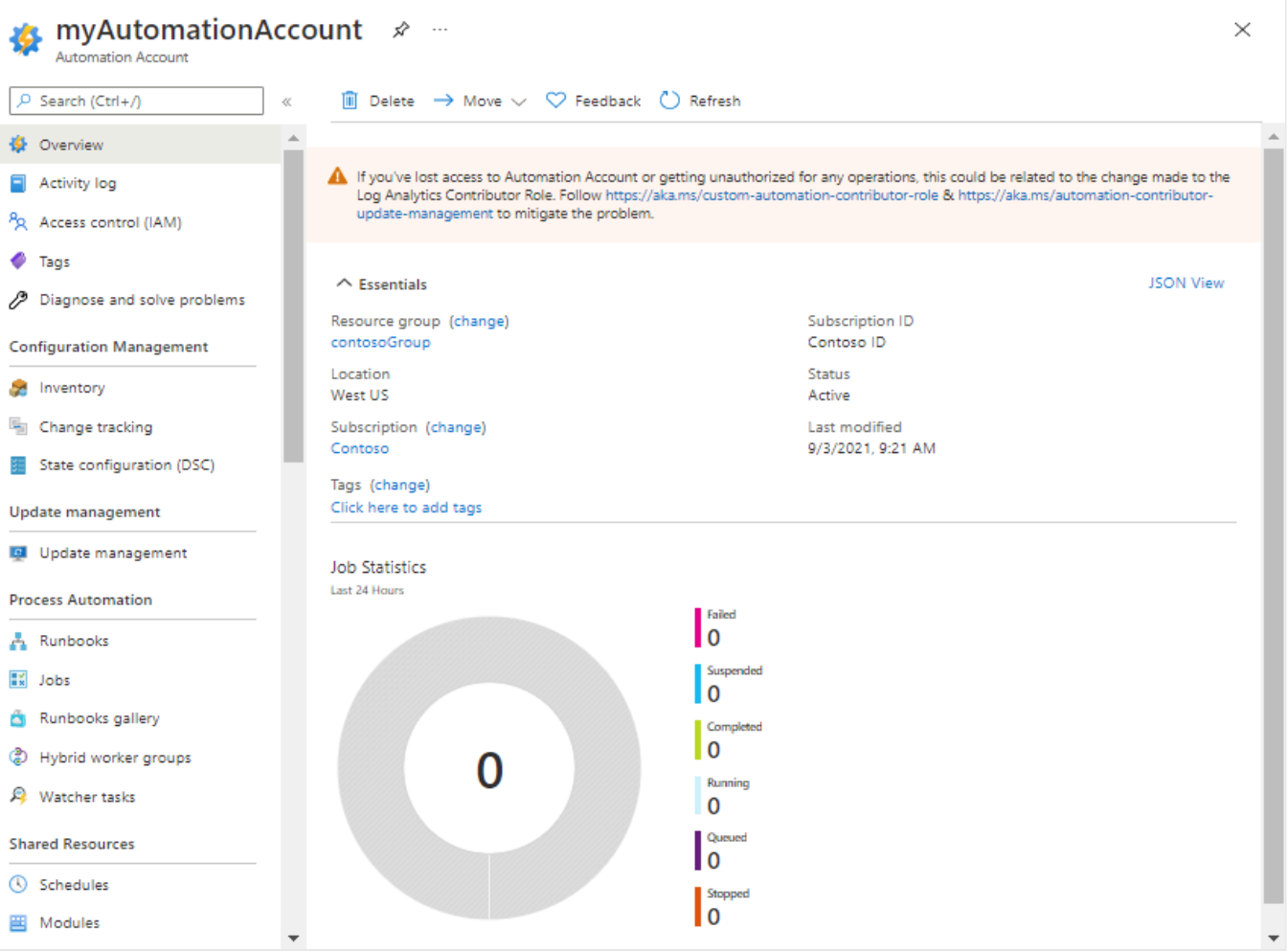
En la pestaña **Etiquetas**, puede especificar etiquetas de Resource Manager para ayudar a organizar los recursos de Azure. Para obtener más información, consulte Etiquetado de recursos, grupos de recursos y suscripciones para una organización lógica.

Revisar y crear

Al ir a la pestaña **Revisar y crear**, Azure ejecuta la validación en la configuración de la cuenta de Automation que ha elegido. Si se supera la validación, puede continuar con la creación de la cuenta de Automation.

En cambio, si se produce un error en la validación, el portal indica qué configuración debe modificarse.

Revise la nueva cuenta de Automation.



Una vez que se crea la cuenta de Automation, se también varios recursos automáticamente. Una vez creados, estos runbooks se pueden eliminar de forma segura, en caso de que no se desee conservarlos. Las identidades administradas pueden utilizarse para autenticarse con la cuenta en un runbook y deben conservarse a menos que se cree otra o que ya no sean necesarias. Las claves de acceso de Automation también se crean durante la creación de la cuenta de Automation. La siguiente tabla resume los recursos de la cuenta.

Expandir tabla

Recurso	Descripción
AzureAutomationTutorialWithIdentityGraphical	Runbook gráfico de ejemplo que muestra cómo realizar la autenticación mediante la identidad administrada. El runbook obtiene todos los recursos de Resource Manager.
AzureAutomationTutorialWithIdentity	Runbook de PowerShell de ejemplo que muestra cómo realizar la autenticación mediante la identidad

Recurso	Descripción
	administrada. El runbook obtiene todos los recursos de Resource Manager.

> **Nota:**
>
> Los runbooks del tutorial no se han actualizado para autenticarse mediante una identidad administrada. Revise la sección Uso de la identidad asignada por el sistema o Uso de la identidad asignada por el usuario para obtener información sobre cómo conceder a la identidad administrada acceso a los recursos y configurar los runbooks para que se autentiquen mediante cualquier tipo de identidad administrada.

Administración de claves de cuenta de Automation

Al crear una cuenta de Automation, Azure genera dos claves de acceso de 512 bits para esa cuenta. Estas claves son claves de acceso compartido que se usan como claves de registro para registrar nodos DSC . Estas claves solo se usan al registrar nodos DSC. Las máquinas existentes configuradas como nodos DSC o instancias de Hybrid Worker no se verán afectadas después de la rotación de estas claves.

Visualización de claves de cuenta de Automation

Para ver y copiar las claves de acceso de la cuenta de Automation, siga estos pasos:

1. En Azure Portal, vaya a su cuenta de Automation.
2. En **Configuración de la cuenta**, seleccione **Claves** para ver las claves de acceso principal y secundaria de la cuenta de Automation. Puede usar cualquiera de las dos claves para acceder a su cuenta de Automation. Sin embargo, se recomienda usar la primera clave y reservar la segunda.

Rotación manual de las claves de acceso

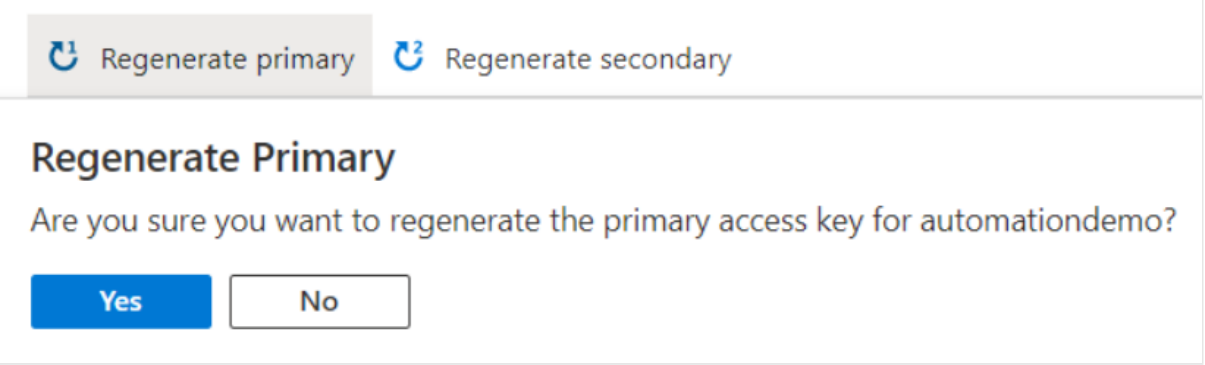
Se recomienda rotar las claves de acceso periódicamente para proteger la cuenta de Automation. Como tiene dos claves de acceso, puede rotarlas mediante Azure Portal o el cmdlet de Azure PowerShell.

Elija un cliente:

Azure Portal

Siga estos pasos:

1. Vaya a su cuenta de Automation en Azure Portal.
2. En **Configuración de la cuenta**, seleccione **Claves**.
3. Seleccione **Regenerar principal** para volver a generar la clave de acceso principal de la cuenta de Automation.
4. Seleccione **Regenerar secundaria** para volver a generar la clave de acceso secundaria.



Visualización de la dirección URL de registro

El nodo DSC se registra con el servicio State Configuration mediante la dirección URL de registro y se autentica mediante una clave de acceso de registro junto con las claves de acceso de la cuenta de Automation.

Pasos siguientes

- Para empezar a trabajar con runbooks de PowerShell, consulte Tutorial: Creación de un runbook de PowerShell.
- Para empezar a trabajar con runbooks de flujo de trabajo de PowerShell, consulte Creación de un runbook de flujo de trabajo de PowerShell.
- Para empezar a trabajar con runbooks de Python 3, consulte Tutorial: Creación de un runbook de Python 3.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.

Last updated on 17/11/2025

Creación de una cuenta de Azure Automation con una plantilla de Resource Manager

Artículo • 15/09/2024

Azure Automation ofrece un servicio de automatización y configuración basado en la nube que admite una administración coherente en los entornos de Azure y que no son de Azure. En este artículo se muestra cómo implementar una plantilla de Azure Resource Manager que crea una cuenta de Automation. El uso de una plantilla de ARM requiere menos pasos que otros métodos de implementación. La plantilla JSON especifica valores predeterminados para los parámetros que es probable que se utilicen como configuración estándar en el entorno. Puede almacenar la plantilla en una cuenta de Azure Storage para el acceso compartido en la organización. Para más información sobre el trabajo con plantillas, consulte Implementación de recursos con plantillas de ARM y la CLI de Azure.

Una plantilla de Azure Resource Manager es un archivo de notación de objetos JavaScript (JSON) que define tanto la infraestructura como la configuración de un proyecto. La plantilla usa sintaxis declarativa. Se describe la implementación deseada sin escribir la secuencia de comandos de programación para crear la implementación.

La plantilla de ejemplo realiza los siguientes pasos:

- Automatiza la creación de un área de trabajo de Log Analytics en Azure Monitor.
- Automatiza la creación de una cuenta de Azure Automation.
- El vínculo de la cuenta de Automation al área de trabajo de Log Analytics.
- Agrega runbooks de Automation de ejemplo a la cuenta.

Si no tiene una suscripción a Azure, cree una cuenta gratuita antes de empezar.

Requisitos previos

Si no está familiarizado con Azure Automation y Azure Monitor, es importante que comprenda los siguientes detalles de configuración. Pueden ayudarle a evitar errores al intentar crear, configurar y usar un área de trabajo de Log Analytics vinculada a la nueva cuenta de Automation.

- Revise los detalles adicionales para conocer totalmente las opciones de configuración del área de trabajo, como el modo de control de acceso, el plan de tarifa, la retención y el nivel de reserva de capacidad.

- Revise las asignaciones de área de trabajo para especificar las regiones admitidas en línea o en un archivo de parámetros. Solo en determinadas regiones se puede vincular un área de trabajo de Log Analytics y una cuenta de Automation en la suscripción.
- Si no está familiarizado con Azure Monitor Logs y aún no ha implementado un área de trabajo, revise la guía de diseño del área de trabajo. Este documento le ayudará a obtener información sobre el control de acceso y a comprender las estrategias de implementación de diseño que recomendamos para su organización.

Revisión de la plantilla

La plantilla usada en este artículo forma parte de las plantillas de inicio rápido de Azure.

JSON

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "workspaceName": {
      "type": "string",
      "metadata": {
        "description": "Workspace name"
      }
    },
    "sku": {
      "type": "string",
      "defaultValue": "pergb2018",
      "allowedValues": [
        "pergb2018",
        "Free",
        "Standalone",
        "PerNode",
        "Standard",
        "Premium"
      ],
      "metadata": {
        "description": "Pricing tier: perGB2018 or legacy tiers (Free, Standalone, PerNode, Standard or Premium), which are not available to all customers."
      }
    },
    "dataRetention": {
      "type": "int",
      "defaultValue": 30,
```

```
      "minValue": 7,
      "maxValue": 730,
      "metadata": {
        "description": "Number of days to retain data."
      }
    },
    "location": {
      "type": "string",
      "defaultValue": "[resourceGroup().location]",
      "metadata": {
        "description": "Specifies the location in which to create the
workspace."
      }
    },
    "automationAccountName": {
      "type": "string",
      "metadata": {
        "description": "Automation account name"
      }
    },
    "sampleGraphicalRunbookName": {
      "type": "String",
      "defaultValue": "AzureAutomationTutorial"
    },
    "sampleGraphicalRunbookDescription": {
      "type": "String",
      "defaultValue": "An example runbook that gets all the Resource Manager
resources by using the Run As account (service principal)."
    },
    "samplePowerShellRunbookName": {
      "type": "String",
      "defaultValue": "AzureAutomationTutorialScript"
    },
    "samplePowerShellRunbookDescription": {
      "type": "String",
      "defaultValue": "An example runbook that gets all the Resource Manager
resources by using the Run As account (service principal)."
    },
    "samplePython2RunbookName": {
      "type": "String",
      "defaultValue": "AzureAutomationTutorialPython2"
    },
    "samplePython2RunbookDescription": {
      "type": "String",
      "defaultValue": "An example runbook that gets all the Resource Manager
resources by using the Run As account (service principal)."
    },
    "_artifactsLocation": {
      "type": "string",
      "defaultValue": "[deployment().properties.templateLink.uri]",
      "metadata": {
        "description": "URI to artifacts location"
      }
    },
    "_artifactsLocationSasToken": {
```

```
      "type": "securestring",
      "defaultValue": "",
      "metadata": {
        "description": "The sasToken required to access _artifactsLocation.
When the template is deployed using the accompanying scripts, a sasToken
will be automatically generated"
      }
    }
  },
  "resources": [
    {
      "type": "Microsoft.OperationalInsights/workspaces",
      "apiVersion": "2020-08-01",
      "name": "[parameters('workspaceName')]",
      "location": "[parameters('location')]",
      "properties": {
        "sku": {
          "name": "[parameters('sku')]"
        },
        "retentionInDays": "[parameters('dataRetention')]",
        "features": {
          "searchVersion": 1,
          "legacy": 0
        }
      }
    },
    {
      "type": "Microsoft.Automation/automationAccounts",
      "apiVersion": "2020-01-13-preview",
      "name": "[parameters('automationAccountName')]",
      "location": "[parameters('location')]",
      "dependsOn": [
        "[parameters('workspaceName')]"
      ],
      "identity": {
        "type": "SystemAssigned"
      },
      "properties": {
        "sku": {
          "name": "Basic"
        }
      },
      "resources": [
        {
          "type": "runbooks",
          "apiVersion": "2020-01-13-preview",
          "name": "[parameters('sampleGraphicalRunbookName')]",
          "location": "[parameters('location')]",
          "dependsOn": [
            "[parameters('automationAccountName')]"
          ],
          "properties": {
            "runbookType": "GraphPowerShell",
            "logProgress": "false",
            "logVerbose": "false",
```

```
            "description": "
[parameters('sampleGraphicalRunbookDescription')]",
            "publishContentLink": {
              "uri": "[uri(parameters('_artifactsLocation'),
concat('scripts/AzureAutomationTutorial.graphrunbook',
parameters('_artifactsLocationSasToken')))]",
              "version": "1.0.0.0"
            }
          }
        },
        {
          "type": "runbooks",
          "apiVersion": "2020-01-13-preview",
          "name": "[parameters('samplePowerShellRunbookName')]",
          "location": "[parameters('location')]",
          "dependsOn": [
            "[parameters('automationAccountName')]"
          ],
          "properties": {
            "runbookType": "PowerShell",
            "logProgress": "false",
            "logVerbose": "false",
            "description": "
[parameters('samplePowerShellRunbookDescription')]",
            "publishContentLink": {
              "uri": "[uri(parameters('_artifactsLocation'),
concat('scripts/AzureAutomationTutorial.ps1',
parameters('_artifactsLocationSasToken')))]",
              "version": "1.0.0.0"
            }
          }
        },
        {
          "type": "runbooks",
          "apiVersion": "2020-01-13-preview",
          "name": "[parameters('samplePython2RunbookName')]",
          "location": "[parameters('location')]",
          "dependsOn": [
            "[parameters('automationAccountName')]"
          ],
          "properties": {
            "runbookType": "Python2",
            "logProgress": "false",
            "logVerbose": "false",
            "description": "
[parameters('samplePython2RunbookDescription')]",
            "publishContentLink": {
              "uri": "[uri(parameters('_artifactsLocation'),
concat('scripts/AzureAutomationTutorialPython2.py',
parameters('_artifactsLocationSasToken')))]",
              "version": "1.0.0.0"
            }
          }
        }
      ]
```

```
    },
    {
      "type": "Microsoft.OperationalInsights/workspaces/linkedServices",
      "apiVersion": "2020-08-01",
      "name": "[concat(parameters('workspaceName'), '/' , 'Automation')]",
      "location": "[parameters('location')]",
      "dependsOn": [
        "[parameters('workspaceName')]",
        "[parameters('automationAccountName')]"
      ],
      "properties": {
        "resourceId": "
[resourceId('Microsoft.Automation/automationAccounts',
parameters('automationAccountName'))]"
      }
    }
  ]
}
```

Recursos de Azure definidos en la plantilla:

- **Microsoft.OperationalInsights/workspaces**: crea un área de trabajo de Azure Log Analytics.
- **Microsoft.Automation/AutomationAccounts**: crea una cuenta de Azure Automation.
- **Microsoft.Automation/automationAccounts/runbooks**: crea un runbook de cuenta de Azure Automation.

Implementación de la plantilla

1. Seleccione el botón **Implementar en Azure** siguiente para iniciar sesión en Azure y abrir la plantilla de ARM.

   Deploy to Azure

2. Escriba o seleccione los siguientes valores:

Expandir tabla

Propiedad	Descripción
Subscription	Seleccione la suscripción de Azure en la lista desplegable.
Resource group	En la lista desplegable, seleccione el grupo de recursos existente o seleccione **Crear**.
Region	Este valor se rellenará automáticamente.

Propiedad	Descripción
Nombre del área de trabajo	Especifique un nombre para el nuevo área de trabajo de Log Analytics.
SKU	El valor predeterminado es el plan de tarifa por GB publicado en el modelo de precios de abril de 2018. Si quiere crear o configurar un área de trabajo de Log Analytics en una suscripción que ha elegido el modelo de precios de abril de 2018, el único plan de tarifa válido de Log Analytics es `PerGB2018`.
Retención de datos	Su valor predeterminado es 30 días.
Location	El valor se rellenará automáticamente con la ubicación usada para el grupo de recursos.
Nombre de la cuenta de Automation	Escriba un nombre para la nueva cuenta de Automation.
Nombre de ejemplo de un runbook gráfico	Déjelo tal cual.
Descripción de ejemplo de un runbook gráfico	Déjelo tal cual.
Nombre de ejemplo de un runbook de PowerShell	Déjelo tal cual.
Descripción de ejemplo del runbook de PowerShell	Déjelo tal cual.
Nombre de ejemplo de Python2Runbook	Déjelo tal cual.
Descripción de ejemplo de Python2Runbook	Déjelo tal cual.
_artifacts Location (Ubicación de _artefactos)	Déjelo tal cual.* URI a la ubicación de artefactos.
_artifacts Location Sas Token (Token de Sas de ubicación de _artefactos)	déjelo en blanco. SasToken necesario para acceder a `_artifactsLocation`. Cuando la plantilla se implementa mediante los scripts complementarios, se genera automáticamente un valor de `sasToken`.

*Cuando intenta ejecutar la plantilla de ARM desde PowerShell, la CLI o la característica Plantillas del portal, si el parámetro `_artifactsLocation` no está

configurado correctamente, recibirá un mensaje de error parecido al siguiente:

`"message": "Deployment template validation failed: 'The template resource '_artifactsLocation' at line '96' and column '31' is not valid: The language expression property 'templateLink' doesn't exist, available properties are 'template, templateHash, parameters, mode, debugSetting, provisioningState'.. Please see https://aka.ms/arm-template-expressions for usage details.'."`

Para evitar este error, cuando se ejecute desde la característica Plantillas del portal, especifique el siguiente valor para el parámetro `_artifactsLocation`: `https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/quickstarts/microsoft.automation/101-automation/azuredeploy.json`.

Si se ejecuta desde PowerShell, incluya el parámetro y su valor `-TemplateUri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/quickstarts/microsoft.automation/101-automation/azuredeploy.json`.

Si se ejecuta desde PowerShell, incluya el parámetro y su valor `--template-uri https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/quickstarts/microsoft.automation/101-automation/azuredeploy.json`.

Para obtener más referencias acerca del uso de PowerShell o la CLI, consulte Creación de una cuenta de Azure Automation (microsoft.com) en la sección **Uso de la plantilla**.

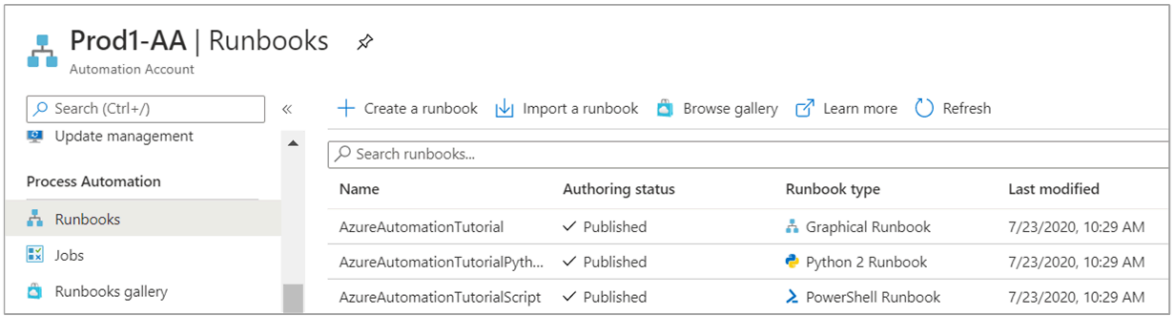
3. Seleccione **Revisar y crear** y, a continuación, **Crear**. La implementación puede tardar unos minutos en finalizar. Cuando se completa, la salida es similar a la siguiente imagen:

```
DeploymentName          : CreateAutomationAcct
ResourceGroupName       : Prod1
ProvisioningState       : Succeeded
Timestamp               : 4/15/2020 3:49:59 PM
Mode                    : Incremental
TemplateLink            :
Parameters              :
                          Name                                Type                        Value
                          ==================================  ========================    ==========
                          workspaceName                       String                      Prod1-LA
                          sku                                 String                      pergb2018
                          dataRetention                       Int                         30
                          immediatePurgeDataOn30Days          Bool                        False
                          location                            String                      eastus
                          automationAccountName               String                      Prod1-AA
                          automationAccountLocation           String                      eastus2
                          sampleGraphicalRunbookName          String                      AzureAutomationTutorial
                          sampleGraphicalRunbookDescription   String                       An example runbook which gets
```

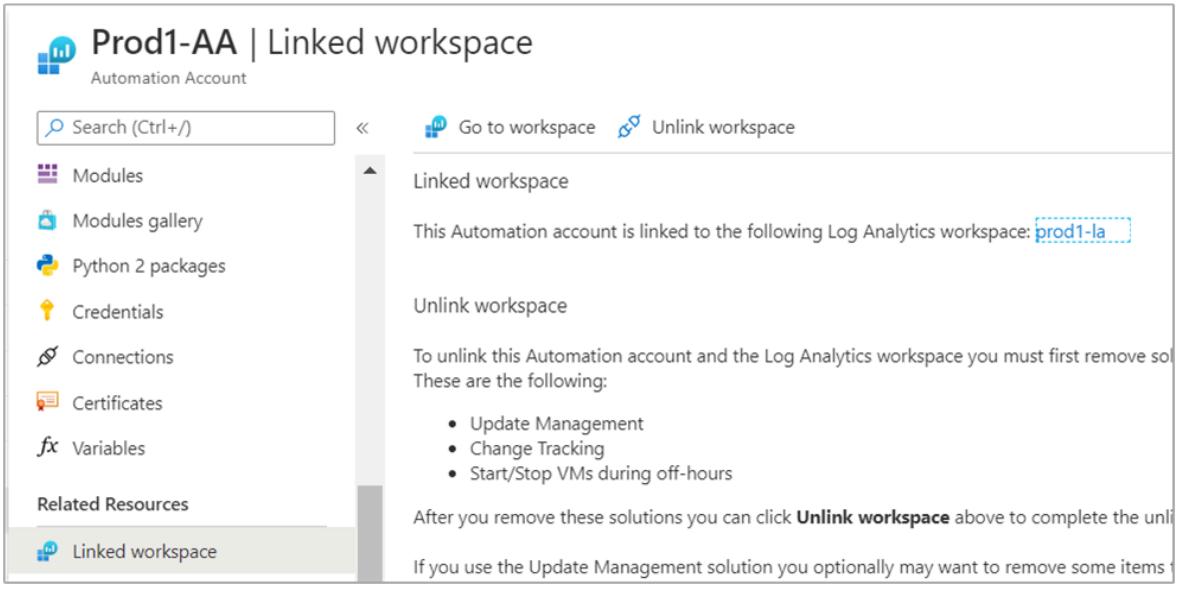
Revisión de los recursos implementados

1. Una vez que completada la implementación, recibirá una notificación de **Implementación correcta** con el vínculo **Ir al recurso**. En la página del **Grupo de recursos**, se muestran los nuevos recursos. En la lista, seleccione la nueva cuenta de Automation.

2. En la parte izquierda, en **Automatización de procesos**, seleccione **Runbooks**. En la página **Runbooks**, se muestran tres runbooks de ejemplo creados con la cuenta de Automation.



3. En la parte izquierda, en **Related Resources** (Recursos relacionados), seleccione **Linked workspace** (Área de trabajo vinculada). En la página **Linked workspace** (Área de trabajo vinculada), se muestra el área de trabajo de Log Analytics que especificó anteriormente como vinculada a la cuenta de Automation.



Pasos siguientes

Configure las opciones de diagnóstico para que la cuenta de Automation envíe el estado del trabajo de runbook y los flujos de trabajo al área de trabajo de Log Analytics vinculada.

Comentarios

¿Le ha resultado útil esta página? Sí No

Proporcionar comentarios sobre el producto | Obtener ayuda en Microsoft Q&A

Deshabilitación de la autenticación local en Automation

> **Importante**
>
> Cuando se deshabilita la autenticación local, esto afecta al inicio de un runbook mediante un webhook, la sincronización automática del control del código fuente y la Automation Desired State Configuration. Para obtener más información, vea Alternativas disponibles.

Azure Automation proporciona compatibilidad con la autenticación de Microsoft Entra para todos los puntos de conexión públicos del servicio Automation. Esta mejora de seguridad crítica quita las dependencias de certificado y proporciona a las organizaciones el control para deshabilitar los métodos de autenticación local. Esta característica proporciona una integración fluida cuando se requiere un control centralizado y la administración de identidades y credenciales de recursos mediante Microsoft Entra ID.

Azure Automation proporciona una característica opcional para "Deshabilitar la autenticación local" en el nivel de cuenta de Automation mediante la directiva de Azure Configurar la cuenta de Azure Automation para deshabilitar la autenticación local. De forma predeterminada, este indicador se establece en false en la cuenta, por lo que puede usar tanto la autenticación local como la autenticación de Microsoft Entra. Si decide deshabilitar la autenticación local, el servicio Automation solo acepta la autenticación basada en Microsoft Entra ID.

En Azure Portal, puede recibir un mensaje de advertencia en la página de aterrizaje de la cuenta de Automation seleccionada si la autenticación está deshabilitada. Para confirmar si la directiva de autenticación local está habilitada, use el cmdlet de PowerShell Get-AzAutomationAccount y compruebe la propiedad `DisableLocalAuth`. Un valor de `true` significa que la autenticación local está deshabilitada.

La deshabilitación de la autenticación local no tiene efecto inmediato. Espere unos minutos para que el servicio bloquee las solicitudes de autenticación futuras.

> **Nota:**
>
> - Actualmente, la compatibilidad de PowerShell con la nueva versión de API (22-06-2021) o la marca `-DisableLocalAuth` no está disponible. Sin embargo, puede usar la

API REST con esta versión de API para actualizar la marca. Aquí se muestra cómo hacerlo con el Invoke-AzRestMethod cmdlet .

Azure PowerShell

```
$SubscriptionId = "your-subscription-id"
$ResourceGroupName = "your-resource-group-name"
$AutomationAccountName = "your-automation-account-name"
$RequestUri =
"https://management.azure.com/subscriptions/$SubscriptionId/resourceGroups/$Res
ourceGroupName/providers/Microsoft.Automation/automationAccounts/$($AutomationA
ccountName)?api-version=2023-11-01"
$Payload = @{
    properties = @{
        disableLocalAuth = $true
    }
} | ConvertTo-Json
Invoke-AzRest -Method Patch -Uri $RequestUri -Payload $Payload

# Verify
$AutomationAccount = (Invoke-AzRest -Method Get -Uri $RequestUri).Content |
ConvertFrom-Json
"disableLocalAuth = $($AutomationAccount.properties.disableLocalAuth)"
```

Nueva habilitación de la autenticación local

Para volver a habilitar la autenticación local, cambie las propiedades de `disableLocalAuth = $true` a `disableLocalAuth = $false` en el script de PowerShell anterior y, a continuación, use el mismo Invoke-AzRestMethod cmdlet. Espere unos minutos para que el servicio acepte el cambio para permitir solicitudes de autenticación local.

Compatibilidad

En la tabla siguiente se describen los comportamientos o características cuyo funcionamiento se impide al deshabilitar la autenticación local.

Expandir tabla

Escenario	Alternativa
Llamada a un runbook con un webhook.	Ejecute un trabajo de libro de ejecución usando una plantilla de Azure Resource Manager, que utiliza autenticación de Microsoft Entra.
Uso de Desired State	Utilice la Configuración de invitados de Azure Policy.

Escenario	Alternativa
Configuration de Automation.	
Uso de agentes híbridos de runbooks basados en agentes.	Use agentes híbridos de runbooks basados en extensiones.

Pasos siguientes

- Introducción a la autenticación de cuentas de Azure Automation

---

Last updated on 05/04/2026

Utilización de una identidad administrada asignada por el sistema para la cuenta de Azure Automation

Artículo • 20/01/2025

En este artículo, se muestra cómo agregar una identidad administrada asignada por el sistema para una cuenta de Azure Automation y cómo usarla para acceder a otros recursos. Para más información sobre cómo funcionan las identidades administradas con Azure Automation, consulte Identidades administradas (versión preliminar).

Si no tiene una suscripción a Azure, cree una cuenta gratuita antes de empezar.

Requisitos previos

- Una cuenta de Azure Automation Para obtener instrucciones, consulte Creación de una cuenta de Azure Automation.
- La versión más reciente de los módulos Az de PowerShell: Az.Accounts, Az.Resources, Az.Automation, Az.KeyVault.
- Un recurso de Azure al que desea acceder desde el runbook de Automation. Este recurso debe tener un rol definido para la identidad administrada, lo que ayuda al runbook de Automation a autenticar el acceso al recurso. Para agregar roles, debe ser propietario del recurso en el inquilino de Microsoft Entra correspondiente.
- No hay ningún requisito de versión mínima para Hybrid Runbook Worker basado en extensiones, y todas las versiones funcionarán. Las versiones mínimas necesarias para Hybrid Worker basado en agente son:
  - Hybrid Runbook Worker de Windows: versión 7.3.1125.0
  - Hybrid Runbook Worker de Linux: versión 1.7.4.0

  Para comprobar las versiones:
  - Hybrid Runbook Worker de Windows: vaya a la ruta de instalación `C:\ProgramFiles\Microsoft Monitoring Agent\Agent\AzureAutomation\.`, y la carpeta *Azure Automation* incluirá una subcarpeta con el número de versión como nombre de la subcarpeta.
  - Hybrid Runbook Worker de Linux: vaya a la ruta de acceso `vi/opt/microsoft/omsconfig/modules/nxOMSAutomationWorker/VERSION.`, y el archivo *VERSION* tendrá el número de versión de Hybrid Worker.

- Para asignar un rol de Azure debe tener un permiso `Microsoft.Authorization/roleAssignments/write`, como Administrador de acceso de usuario o Propietario.

Habilitación de una identidad administrada asignada por el sistema para la cuenta de Azure Automation

Una vez habilitada, las siguientes propiedades se asignarán a la identidad administrada asignada por el sistema.

Expandir tabla

Propiedad (JSON)	Value	Descripción
principalid	<principal-ID>	Identificador único global (GUID) del objeto de entidad de servicio para la identidad administrada por el sistema que representa la cuenta de Automation en el inquilino de Microsoft Entra. Este GUID aparece a veces como "Id. de objeto" u objectID.
tenantid	<Azure-AD-tenant-ID>	Identificador único global (GUID) que representa el inquilino de Microsoft Entra del que es ahora miembro la cuenta de Automation. Dentro del inquilino de Microsoft Entra, la entidad de servicio tiene el mismo nombre que la cuenta de Automation.

Puede agregar una identidad administrada asignada por el sistema para una cuenta de Azure Automation mediante Azure Portal, PowerShell, la API REST de Azure o una plantilla de ARM. Para los ejemplos que implican PowerShell, inicie sesión primero en Azure de forma interactiva con el cmdlet Connect-AzAccount y siga las instrucciones.

PowerShell

```
# Sign in to your Azure subscription
$sub = Get-AzSubscription -ErrorAction SilentlyContinue
if(-not($sub))
{
    Connect-AzAccount
}

# If you have multiple subscriptions, set the one to use
# Select-AzSubscription -SubscriptionId "<SUBSCRIPTIONID>"
```

A continuación, inicialice un conjunto de variables que se usarán a lo largo de los ejemplos. Revise los valores siguientes y, a continuación, ejecute.

PowerShell

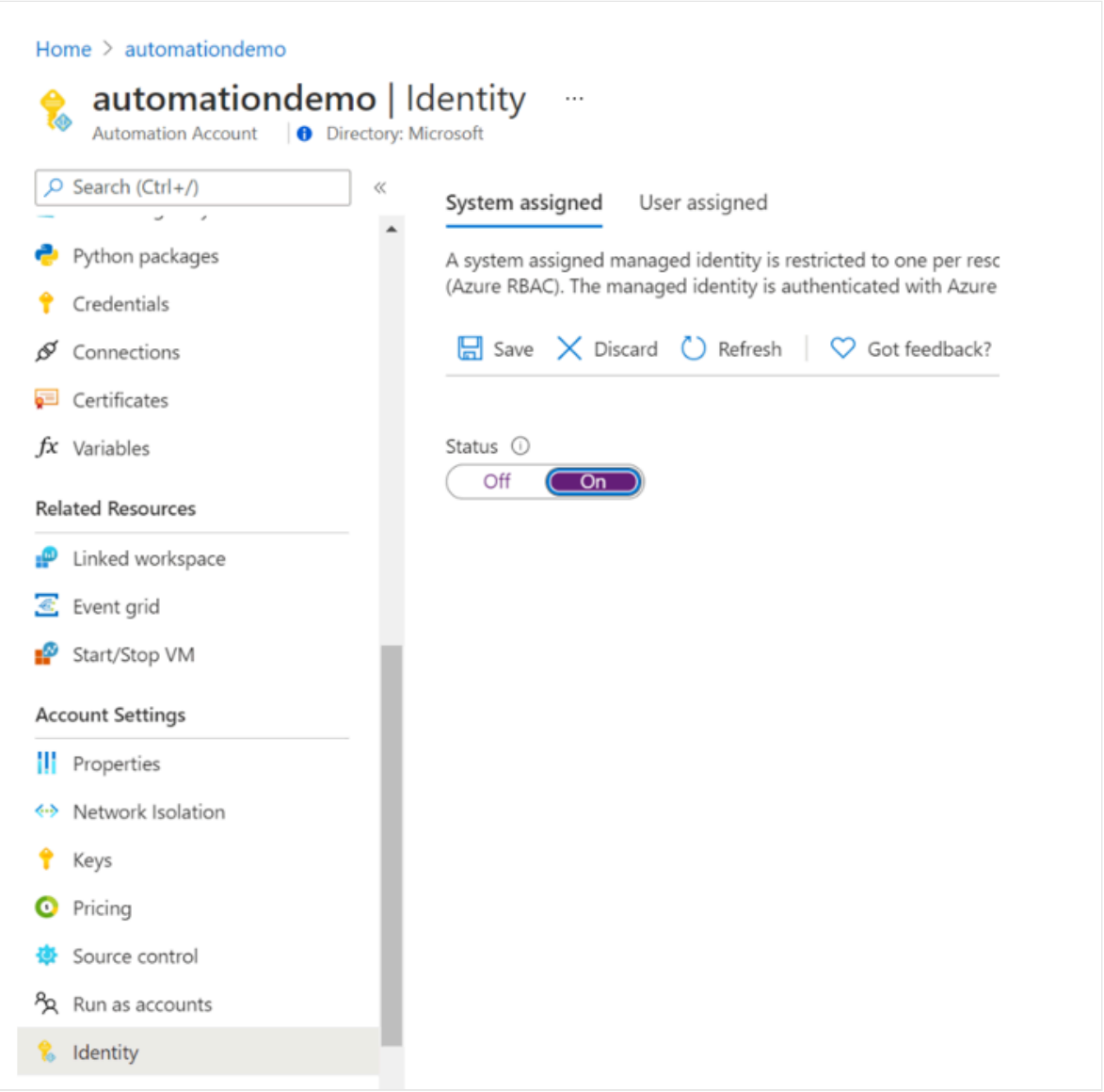
```
$subscriptionID = "subscriptionID"
$resourceGroup = "resourceGroupName"
$automationAccount = "automationAccountName"
```

> **Importante**
>
> La nueva identidad de nivel de cuenta de Automation invalida las identidades asignadas por el sistema de nivel de máquina virtual anteriores que se describen en Uso de la autenticación de runbook con identidades administradas. Si ejecuta trabajos híbridos en máquinas virtuales de Azure que usan la identidad asignada por el sistema de la máquina virtual para acceder a los recursos del runbook, la identidad de la cuenta de Automation se usará para los trabajos híbridos.
>
> Si desea seguir usando la identidad administrada de la máquina virtual, no debe habilitar la identidad de nivel de cuenta de Automation. Si ya la ha habilitado, puede deshabilitar la identidad administrada por el sistema de la cuenta de Automation. Consulte Deshabilitación de la identidad administrada de la cuenta de Azure Automation.

Habilitación de Azure Portal

Lleve a cabo los siguiente pasos:

1. Inicie sesión en Azure Portal.
2. En Azure Portal, vaya a su cuenta de Automation.
3. En **Configuración de la cuenta**, seleccione **Identidad**.
4. Establezca la opción **Asignado por el sistema** en **Activado** y presione **Guardar**. Cuando se le solicite confirmación, seleccione **Sí**.



La cuenta de Automation ahora podrá usar la identidad asignada por el sistema, registrada con Microsoft Entra ID y representada mediante un identificador de objeto.

Habilitar mediante PowerShell

Use el cmdlet de PowerShell Set-AzAutomationAccount para agregar las identidades administradas asignadas por el sistema.

PowerShell

```
$output = Set-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount `
    -AssignSystemIdentity

$output
```

La salida debe tener una apariencia similar a la siguiente:

```
SubscriptionId        : SubscriptionID
ResourceGroupName     : ContosoLab
AutomationAccountName : myAutomationAccount
Location              : westus
State                 : Ok
Plan                  : Basic
CreationTime          : 7/9/2021 12:18:26 AM +00:00
LastModifiedTime      : 7/9/2021 4:37:06 PM +00:00
LastModifiedBy        :
Tags                  : {}
Identity              : Microsoft.Azure.Management.Automation.Models.Identity
Encryption            : Microsoft.Azure.Management.Automation.Models.EncryptionProperties
PublicNetworkAccess   :
```

Para obtener una salida adicional, modifique el ejemplo para especificar: `$output.identity | ConvertTo-Json`.

Habilitación mediante una API REST

A continuación, se proporcionan los pasos de ejemplo y la sintaxis.

Sintaxis

La sintaxis del cuerpo siguiente permite una identidad administrada asignada por el sistema en una cuenta de Automation mediante el método HTTP **PATCH**. Sin embargo, esta sintaxis quitará las identidades administradas asignadas por el sistema existentes asociadas a la cuenta de Automation.

JSON

```
{
 "identity": {
   "type": "SystemAssigned"
  }
}
```

Si hay varias identidades asignadas por el usuario definidas, para conservarlas y quitar solo la identidad asignada por el sistema, debe especificar cada identidad asignada por

el usuario mediante una lista delimitada por comas. En el ejemplo siguiente, se usa el método HTTP **PATCH**.

JSON

```
{
  "identity" : {
    "type": "SystemAssigned, UserAssigned",
    "userAssignedIdentities": {
        "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/resourceGroupName/providers/Microsoft.ManagedIde
ntity/userAssignedIdentities/cmkID": {},
        "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/resourceGroupName/providers/Microsoft.ManagedIde
ntity/userAssignedIdentities/cmkID2": {}
    }
  }
}
```

La sintaxis de la API es la siguiente:

HTTP

```
PATCH https://management.azure.com/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/resource-group-
name/providers/Microsoft.Automation/automationAccounts/automation-account-
name?api-version=2020-01-13-preview
```

Ejemplo

Lleve a cabo los pasos siguientes.

1. Copie y pegue la sintaxis del cuerpo en un archivo denominado `body_sa.json`. Guarde el archivo en la máquina local o en una cuenta de almacenamiento de Azure.

2. Actualice el valor de variable siguiente y, a continuación, ejecútelo.

   PowerShell

   ```
   $file = "path\body_sa.json"
   ```

3. Este ejemplo usa el cmdlet de PowerShell Invoke-RestMethod para enviar una solicitud PATCH a la cuenta de Automation.

PowerShell

```
# build URI
$URI =
"https://management.azure.com/subscriptions/$subscriptionID/resourceGro
ups/$resourceGroup/providers/Microsoft.Automation/automationAccounts/$a
utomationAccount`?api-version=2020-01-13-preview"

# build body
$body = Get-Content $file

# obtain access token
$azContext = Get-AzContext
$azProfile =
[Microsoft.Azure.Commands.Common.Authentication.Abstractions.AzureRmPro
fileProvider]::Instance.Profile
$profileClient = New-Object -TypeName
Microsoft.Azure.Commands.ResourceManager.Common.RMProfileClient -
ArgumentList ($azProfile)
$token =
$profileClient.AcquireAccessToken($azContext.Subscription.TenantId)
$authHeader = @{
    'Content-Type'='application/json'
    'Authorization'='Bearer ' + $token.AccessToken
}

# Invoke the REST API
$response = Invoke-RestMethod -Uri $URI -Method PATCH -Headers
$authHeader -Body $body

# Review output
$response.identity | ConvertTo-Json
```

La salida debe tener una apariencia similar a la siguiente:

JSON

```
{
    "PrincipalId":  "aaaaaaaa-aaaa-aaaa-aaaa-aaaaaaaaaaaa",
    "TenantId":  "bbbbbbbb-bbbb-bbbb-bbbb-bbbbbbbbbbbb",
    "Type":  0,
    "UserAssignedIdentities":  null
}
```

Habilitación mediante una plantilla de ARM

A continuación, se proporcionan los pasos de ejemplo y la sintaxis.

Sintaxis de plantillas

La sintaxis de plantilla de ejemplo siguiente permite una identidad administrada asignada por el sistema en la cuenta de Automation. Sin embargo, esta sintaxis quitará las identidades administradas asignadas por el sistema existentes asociadas a la cuenta de Automation.

JSON

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "resources": [
    {
      "type": "Microsoft.Automation/automationAccounts",
      "apiVersion": "2020-01-13-preview",
      "name": "yourAutomationAccount",
      "location": "[resourceGroup().location]",
      "identity": {
        "type": "SystemAssigned"
        },
      "properties": {
        "sku": {
          "name": "Basic"
        }
      }
    }
  ]
}
```

Ejemplo

Lleve a cabo los pasos siguientes.

1. Revise la sintaxis de la plantilla anterior para usar la cuenta de Automation y guárdela en un archivo denominado `template_sa.json`.

2. Actualice el valor de variable siguiente y, a continuación, ejecútelo.

   PowerShell

   ```
   $templateFile = "path\template_sa.json"
   ```

3. Use el cmdlet de PowerShell New-AzResourceGroupDeployment para implementar la plantilla.

   PowerShell

```
New-AzResourceGroupDeployment `
    -Name "SystemAssignedDeployment" `
    -ResourceGroupName $resourceGroup `
    -TemplateFile $templateFile
```

El comando no producirá ninguna salida; sin embargo, puede usar el código siguiente para comprobarlo:

PowerShell

```
(Get-AzAutomationAccount `
-ResourceGroupName $resourceGroup `
-Name $automationAccount).Identity | ConvertTo-Json
```

La salida tendrá un aspecto similar al que se muestra en el ejemplo anterior de la API REST.

Asignación de un rol a una identidad administrada asignada por el sistema

Una cuenta de Automation puede utilizar su identidad administrada asignada por el sistema para obtener tokens para acceder a otros recursos protegidos por Microsoft Entra D, como Azure Key Vault. Estos tokens no representan a ningún usuario específico de la aplicación, sino que representan la aplicación que accede al recurso. En este caso, por ejemplo, el token representa una cuenta de Automation.

Para poder usar la identidad administrada asignada por el sistema para llevar a cabo cualquier acción en Azure, configure el acceso para esa identidad en el recurso de Azure en el que planee usar la identidad. Para completar esta tarea, asigne el rol adecuado a esa identidad en el recurso de Azure de destino.

Siga la entidad de seguridad con menos privilegios y asigne cuidadosamente solo los permisos necesarios para ejecutar el runbook. Por ejemplo, si la cuenta de Automation solo es necesaria para iniciar o detener una VM de Azure, los permisos asignados a la cuenta de ejecución o a la identidad administrada solo deben ser para iniciar o detener la VM. De manera similar, si un runbook lee el almacenamiento de blobs, asigne permisos de solo lectura.

En el ejemplo siguiente se usa Azure PowerShell para mostrar cómo asignar el rol Colaborador en la suscripción al recurso de Azure de destino. El rol Colaborador se usa como ejemplo y puede ser necesario o no en su caso.

PowerShell

```
New-AzRoleAssignment `
    -ObjectId <automation-Identity-object-id> `
    -Scope "/subscriptions/<subscription-id>" `
    -RoleDefinitionName "Contributor"
```

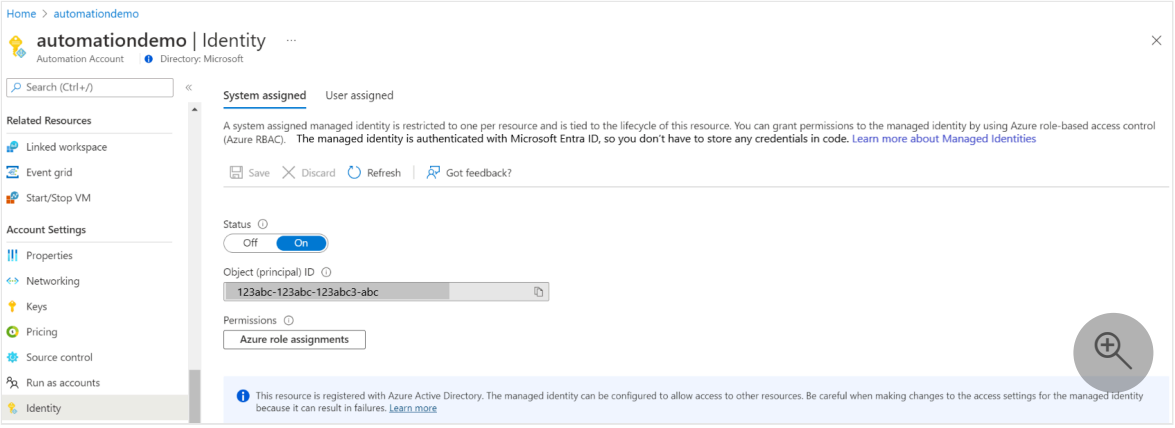
Comprobación de la asignación de roles a una identidad administrada por el sistema

Para comprobar un rol en una identidad administrada asignada por el sistema de la cuenta de Automation, siga estos pasos:

1. Inicie sesión en Azure Portal.

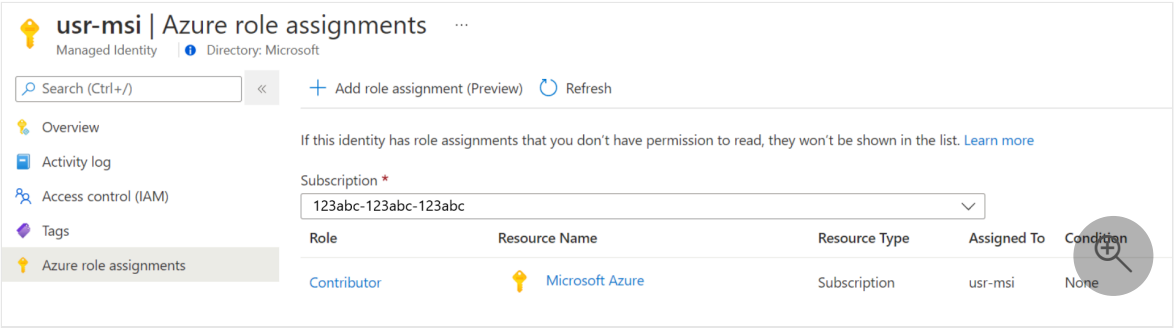
2. Vaya a su cuenta de Automation.

3. En **Configuración de la cuenta**, seleccione **Identidad**.



4. En **Permisos**, haga clic en **Asignaciones de roles de Azure**.

   Si los roles ya están asignados a la identidad administrada asignada por el sistema seleccionada, puede ver una lista de asignaciones de roles. En esta lista se incluyen todas las asignaciones de roles que puede leer.



5. Para cambiar la suscripción, haga clic en la lista desplegable **Suscripción** y seleccione la suscripción adecuada.

6. Haga clic en **Agregar asignación de roles (versión preliminar)**.

7. En la lista desplegable, seleccione el conjunto de recursos al que se aplica la asignación de roles: **Suscripción**, **Grupo de recursos**, **Rol** y **Ámbito**.
   Si no tiene la asignación de roles, puede ver los permisos de escritura del ámbito seleccionado como mensaje insertado.

8. En la lista desplegable **Rol**, seleccione un rol como *Colaborador de máquina virtual*.

9. Haga clic en **Save**(Guardar).

Transcurridos unos minutos, se asigna el rol a la identidad administrada en el ámbito seleccionado.

Autenticación del acceso con identidad administrada asignada por el sistema

Después de habilitar la identidad administrada para la cuenta de Automation y de conceder acceso a una identidad al recurso de destino, puede especificar esa identidad en los runbooks para los recursos que admiten la identidad administrada. Para compatibilidad con las identidades, use el cmdlet `Connect-AzAccount`. Consulte Connect-AzAccount en la referencia de PowerShell.

PowerShell

```
# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process
```

```
# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# Set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext
```

> ⓘ **Nota**
>
> Si su organización sigue usando los cmdlets de AzureRM en desuso, puede usar `Connect-AzureRMAccount -Identity`.

Generación de un token de acceso sin usar cmdlets de Azure

Para los puntos de conexión HTTP, asegúrese de lo siguiente.

- El encabezado de metadatos debe estar presente y establecido en "true".
- Se debe pasar un recurso junto con la solicitud, como parámetro de consulta para una solicitud GET y como datos de formulario para una solicitud POST.
- Establezca el valor de la variable de entorno IDENTITY_HEADER en X-IDENTITY-HEADER.
- El tipo de contenido de la solicitud Post debe ser "application/x-www-form-urlencoded".

Obtención del token de acceso para la identidad administrada asignada por el sistema mediante HTTP Get

PowerShell

```
$resource= "?resource=https://management.azure.com/"
$url = $env:IDENTITY_ENDPOINT + $resource
$Headers = New-Object "System.Collections.Generic.Dictionary[[String],
[String]]"
$Headers.Add("X-IDENTITY-HEADER", $env:IDENTITY_HEADER)
$Headers.Add("Metadata", "True")
$accessToken = Invoke-RestMethod -Uri $url -Method 'GET' -Headers $Headers
Write-Output $accessToken.access_token
```

Obtención del token de acceso para la identidad asignada por el sistema mediante HTTP Post

PowerShell

```
$url = $env:IDENTITY_ENDPOINT
$headers = New-Object "System.Collections.Generic.Dictionary[[String],
[String]]"
$headers.Add("X-IDENTITY-HEADER", $env:IDENTITY_HEADER)
$headers.Add("Metadata", "True")
$body = @{resource='https://management.azure.com/' }
$accessToken = Invoke-RestMethod $url -Method 'POST' -Headers $headers -
ContentType 'application/x-www-form-urlencoded' -Body $body
Write-Output $accessToken.access_token
```

Uso de identidades administradas asignadas por el sistema para acceder a Azure Key Vault en Azure PowerShell

Para obtener más información, consulte Get-AzKeyVaultSecret.

PowerShell

```
Write-Output "Connecting to azure via  Connect-AzAccount -Identity"
Connect-AzAccount -Identity
Write-Output "Successfully connected with Automation account's Managed
Identity"
Write-Output "Trying to fetch value from key vault using MI. Make sure you
have given correct access to Managed Identity"
$secret = Get-AzKeyVaultSecret -VaultName '<KVname>' -Name '<KeyName>'

$ssPtr =
[System.Runtime.InteropServices.Marshal]::SecureStringToBSTR($secret.SecretV
alue)
try {
  $secretValueText =
[System.Runtime.InteropServices.Marshal]::PtrToStringBSTR($ssPtr)
    Write-Output $secretValueText
} finally {
    [System.Runtime.InteropServices.Marshal]::ZeroFreeBSTR($ssPtr)
}
```

Uso de una identidad administrada asignada por el sistema en un runbook de Python

Python

```
#!/usr/bin/env python3
import os
import requests
```

```
# printing environment variables
endPoint = os.getenv('IDENTITY_ENDPOINT')+"?
resource=https://management.azure.com/"
identityHeader = os.getenv('IDENTITY_HEADER')
payload={}
headers = {
  'X-IDENTITY-HEADER': identityHeader,
  'Metadata': 'True'
}
response = requests.request("GET", endPoint, headers=headers, data=payload)
print(response.text)
```

Uso de una identidad administrada asignada por el sistema para acceder a SQL Database

Para más información sobre el aprovisionamiento del acceso a una base de datos de Azure SQL, consulte Aprovisionamiento del administrador de Microsoft Entra (SQL Database).

PowerShell

```
$queryParameter = "?resource=https://database.windows.net/"
$url = $env:IDENTITY_ENDPOINT + $queryParameter
$Headers = New-Object "System.Collections.Generic.Dictionary[[String],
[String]]"
$Headers.Add("X-IDENTITY-HEADER", $env:IDENTITY_HEADER)
$Headers.Add("Metadata", "True")
$content =[System.Text.Encoding]::Default.GetString((Invoke-WebRequest -
UseBasicParsing -Uri $url -Method 'GET' -Headers
$Headers).RawContentStream.ToArray()) | ConvertFrom-Json
$Token = $content.access_token
echo "The managed identities for Azure resources access token is $Token"
$SQLServerName = "<ServerName>"    # Azure SQL logical server name
$DatabaseName = "<DBname>"     # Azure SQL database name
Write-Host "Create SQL connection string"
$conn = New-Object System.Data.SqlClient.SQLConnection
$conn.ConnectionString = "Data
Source=$SQLServerName.database.windows.net;Initial
Catalog=$DatabaseName;Connect Timeout=30"
$conn.AccessToken = $Token
Write-host "Connect to database and execute SQL script"
$conn.Open()
$ddlstmt = "CREATE TABLE Person( PersonId INT IDENTITY PRIMARY KEY,
FirstName NVARCHAR(128) NOT NULL)"
Write-host " "
Write-host "SQL DDL command"
$ddlstmt
$command = New-Object -TypeName System.Data.SqlClient.SqlCommand($ddlstmt,
$conn)
Write-host "results"
```

```
$command.ExecuteNonQuery()
$conn.Close()
```

Migración de cuentas de ejecución existentes a identidad administrada

Azure Automation proporcionaba la autenticación para administrar los recursos de Azure Resource Manager o los recursos implementados en el modelo de implementación clásica con la cuenta de ejecución. Para cambiar de una cuenta de ejecución a una identidad administrada para la autenticación del runbook, siga estos pasos.

1. Habilite una identidad administrada asignada por el sistema, una identidad administrada asignada por el usuario o ambos tipos de identidades administradas.
2. Conceda a la identidad administrada los privilegios para los recursos de Azure que coincidan con los que se asignaron a la cuenta de ejecución.
3. Actualice los runbooks para autenticarse mediante la identidad administrada.
4. Modifique los runbooks para usar la identidad administrada. Para compatibilidad con las identidades, use el cmdlet `Connect-AzAccount`. Consulte Connect-AzAccount en la referencia de PowerShell.
    - Si usa módulos de AzureRM, actualice `AzureRM.Profile` a la versión más reciente y reemplace mediante el cmdlet `Add-AzureRMAccount` por `Connect-AzureRMAccount -Identity`.
    - Si usa módulos Az, actualice a la versión más reciente siguiendo los pasos descritos en el artículo Actualización de los módulos de Azure PowerShell.

Pasos siguientes

- Si los runbooks no se completan correctamente, revise Solución de problemas de identidad administrada de Azure Automation.
- Si necesita deshabilitar una identidad administrada, vea Deshabilitación de la identidad administrada de la cuenta de Azure Automation.
- Para información general sobre la seguridad de l cuenta de Azure Automation, consulte Introducción a la autenticación de cuentas de Automation.

Comentarios

¿Le ha resultado útil esta página? Sí No

Proporcionar comentarios sobre el producto | Obtener ayuda en Microsoft Q&A

Uso de una identidad administrada asignada por el usuario para una cuenta de Azure Automation

En este artículo se muestra cómo agregar una identidad administrada asignada por el usuario para una cuenta de Azure Automation y cómo usarla para acceder a otros recursos. Para obtener más información sobre cómo funcionan las identidades administradas con Azure Automation, consulte Administradas identidades.

> **Nota:**
>
> No es posible usar una identidad administrada asignada por el usuario en una instancia de Hybrid Runbook Worker cuando se ha creado una identidad administrada (asignada por el sistema o por el usuario) para la cuenta de Automation. Si la Identidad Administrada no se ha asignado a la Cuenta de Automatización, es posible utilizar la Identidad Administrada asignada por el Sistema o el Usuario de la máquina virtual en un Hybrid Runbook Worker que sea una máquina virtual de Azure con las identidades administradas asignadas.

Si no tiene una suscripción de Azure, cree una cuenta free antes de comenzar.

Requisitos previos

- Una cuenta de Azure Automation. Para obtener instrucciones, consulte Crear una cuenta de Azure Automation.
- La identidad administrada asignada por el usuario y los recursos de destino Azure que administra el runbook mediante esa identidad pueden estar en distintas suscripciones de Azure.
- La versión más reciente de los módulos de cuenta de Azure. Actualmente, es la versión 2.2.8. (Consulte Az.Accounts para obtener más información sobre esta versión).
- Un recurso de Azure al que desea acceder desde el runbook de Automation. Este recurso debe tener un rol definido para la identidad administrada asignada por el usuario, lo que ayuda al runbook de Automation a autenticar el acceso al recurso. Para agregar roles, debe ser propietario del recurso en la entidad de Microsoft Entra correspondiente.

- Para asignar un rol de Azure, debe tener permisos de `Microsoft.Authorization/roleAssignments/write`, como Administrador de acceso de usuario o Owner.

Agregar una identidad administrada asignada por el usuario a la cuenta de Azure Automation

Puede agregar una identidad administrada asignada por el usuario para una cuenta de Azure Automation mediante el portal de Azure, PowerShell, la API rest de Azure o la plantilla de ARM. Para ver los ejemplos relacionados con PowerShell, inicie sesión primero en Azure de forma interactiva mediante el cmdlet Connect-AzAccount y siga las instrucciones.

PowerShell

```
# Sign in to your Azure subscription
$sub = Get-AzSubscription -ErrorAction SilentlyContinue
if(-not($sub))
{
    Connect-AzAccount
}

# If you have multiple subscriptions, set the one to use
# Select-AzSubscription -SubscriptionId "<SUBSCRIPTIONID>"
```

A continuación, inicialice un conjunto de variables que se usarán a lo largo de los ejemplos. Revise los valores siguientes y, a continuación, ejecute:

PowerShell

```
$subscriptionID = "subscriptionID"
$resourceGroup = "resourceGroupName"
$automationAccount = "automationAccountName"
$userAssignedOne = "userAssignedIdentityOne"
$userAssignedTwo = "userAssignedIdentityTwo"
```

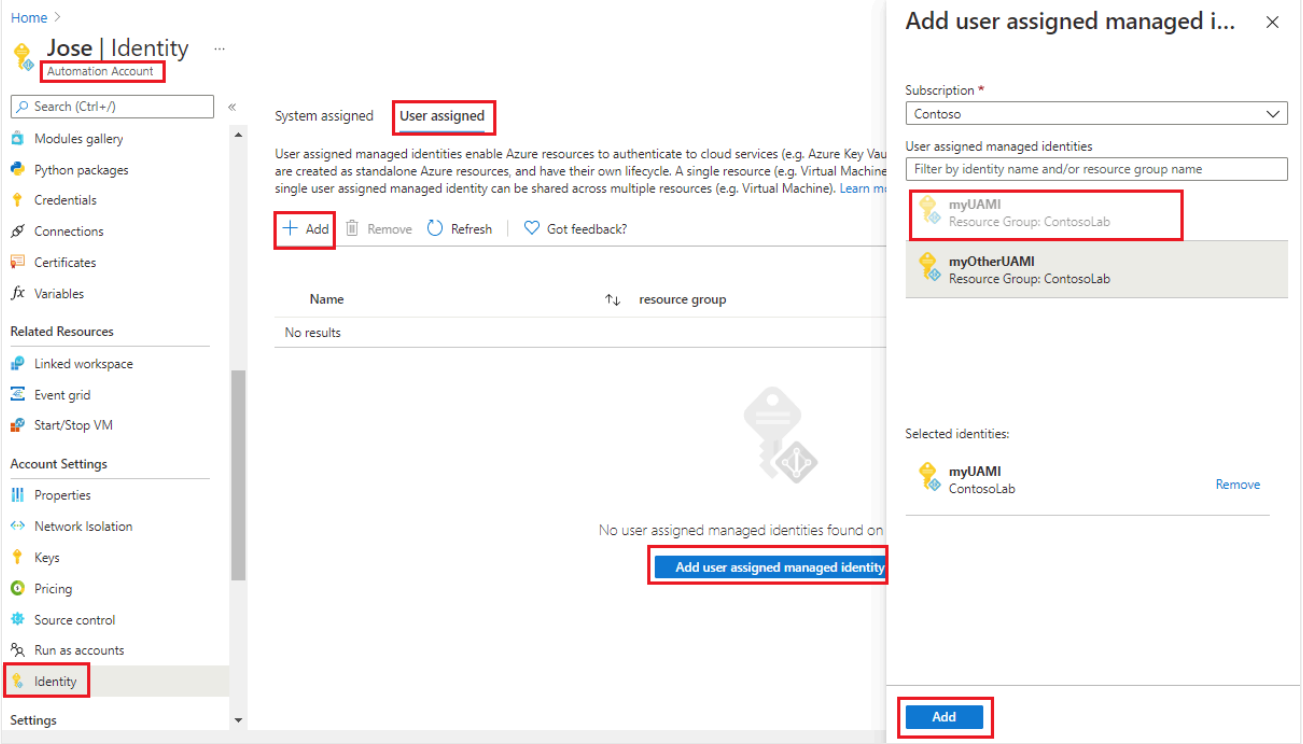
Adición mediante el portal de Azure

Lleve a cabo los siguiente pasos:

1. Inicie sesión en el portal Azure.
2. En el portal de Azure, vaya a la cuenta de Automation.
3. En **Configuración de la cuenta**, seleccione **Identidad**.

4. Seleccione la pestaña **User assigned (Usuario asignado)** y, a continuación, seleccione **Agregar**.

5. Seleccione la identidad administrada asignada por el usuario existente y, a continuación, seleccione **Agregar**. A continuación, se le devolverá a la pestaña **Usuario asignado**.



Incorporación mediante PowerShell

Use el cmdlet set-AzAutomationAccount de PowerShell para agregar las identidades administradas asignadas por el usuario. En primer lugar, debe tener en cuenta si hay una identidad administrada asignada por el sistema existente. En el ejemplo siguiente, se agregan dos identidades administradas asignadas por el usuario existentes a una cuenta de Automation existente y se deshabilitará una identidad administrada asignada por el sistema si existe una.

PowerShell

```
$output = Set-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount `
    -AssignUserIdentity
"/subscriptions/$subscriptionID/resourcegroups/$resourceGroup/providers/Microsoft.Managedidentity/userAssignedIdentities/$userAssignedOne", `

"/subscriptions/$subscriptionID/resourcegroups/$resourceGroup/providers/Microsoft.Managedidentity/userAssignedIdentities/$userAssignedTwo"

$output
```

Para mantener una identidad administrada asignada por el sistema existente, use lo siguiente:

PowerShell

```
$output = Set-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount `
    -AssignUserIdentity
"/subscriptions/$subscriptionID/resourcegroups/$resourceGroup/providers/Microsoft.Mana
gedIdentity/userAssignedIdentities/$userAssignedOne", `

"/subscriptions/$subscriptionID/resourcegroups/$resourceGroup/providers/Microsoft.Mana
gedIdentity/userAssignedIdentities/$userAssignedTwo" `
    -AssignSystemIdentity

$output
```

La salida debe tener una apariencia similar a la siguiente:

```
SubscriptionId        : SubscriptionID
ResourceGroupName     : ContosoLab
AutomationAccountName : myAutomationAccount
Location              : westus
State                 : Ok
Plan                  : Basic
CreationTime          : 7/9/2021 12:18:26 AM +00:00
LastModifiedTime      : 7/9/2021 4:37:06 PM +00:00
LastModifiedBy        :
Tags                  : {}
Identity              : Microsoft.Azure.Management.Automation.Models.Identity
Encryption            : Microsoft.Azure.Management.Automation.Models.EncryptionProperties
PublicNetworkAccess   :
```

Para obtener una salida adicional, ejecute: `$output.identity | ConvertTo-Json`.

Adición mediante una API REST

A continuación, se proporcionan los pasos de ejemplo y la sintaxis.

Sintaxis

La sintaxis del cuerpo del ejemplo siguiente habilita una identidad administrada asignada por el sistema si aún no está habilitada y asigna dos identidades administradas asignadas por el usuario existentes a la cuenta de Automation existente.

PATCH

JSON

```
{
  "identity": {
```

```
    "type": "SystemAssigned, UserAssigned",
    "userAssignedIdentities": {
      "/subscriptions/00a000aa-00a0-00aa-00aa-0a0aa000aa00/resourceGroups/resource-group-name/providers/Microsoft.ManagedIdentity/userAssignedIdentities/firstIdentity": {},
      "/subscriptions/00a000aa-00a0-00aa-00aa-0a0aa000aa00/resourceGroups/resource-group-name/providers/Microsoft.ManagedIdentity/userAssignedIdentities/secondIdentity": {}
    }
  }
}
```

La sintaxis de la API es la siguiente:

HTTP

```
https://management.azure.com/subscriptions/00a000aa-00a0-00aa-00aa-0a0aa000aa00/resourceGroups/resource-group-name/providers/Microsoft.Automation/automationAccounts/automation-account-name?api-version=2020-01-13-preview
```

Ejemplo

Lleve a cabo los siguiente pasos.

1. Revise la sintaxis del cuerpo anterior en un archivo llamado `body_ua.json`. Guarde el archivo en el equipo local o en una cuenta de almacenamiento de Azure.

2. Revise el valor de la variable siguiente y, a continuación, ejecute:

   PowerShell

   ```
   $file = "path\body_ua.json"
   ```

3. En este ejemplo se usa el cmdlet de PowerShell Invoke-RestMethod para enviar la solicitud PATCH a la cuenta de Automation.

   PowerShell

   ```
   # build URI
   $URI =
   "https://management.azure.com/subscriptions/$subscriptionID/resourceGroups/$resourceGroup/providers/Microsoft.Automation/automationAccounts/$automationAccount`?api-version=2020-01-13-preview"

   # build body
   $body = Get-Content $file
   ```

```
# obtain access token
$azContext = Get-AzContext
$azProfile =
[Microsoft.Azure.Commands.Common.Authentication.Abstractions.AzureRmProfileProvid
er]::Instance.Profile
$profileClient = New-Object -TypeName
Microsoft.Azure.Commands.ResourceManager.Common.RMProfileClient -ArgumentList
($azProfile)
$token = $profileClient.AcquireAccessToken($azContext.Subscription.TenantId)
$authHeader = @{
    'Content-Type'='application/json'
    'Authorization'='Bearer ' + $token.AccessToken
}

# Invoke the REST API
$response = Invoke-RestMethod -Uri $URI -Method PATCH -Headers $authHeader -Body
$body

# Review output
$response.identity | ConvertTo-Json
```

La salida debe tener una apariencia similar a la siguiente:

JSON

```
{
"type": "SystemAssigned, UserAssigned",
"principalId": "00a000aa-00a0-00aa-00aa-0a0aa000aa00",
"tenantId": "00a000aa-00a0-00aa-00aa-0a0aa000aa00",
"userAssignedIdentities":  {

"/subscriptions/ContosoID/resourcegroups/ContosoLab/providers/Microsoft.ManagedId
entity/userAssignedIdentities/ContosoUAMI1":  {
            "PrincipalId":  "00a000aa-00a0-00aa-00aa-0a0aa000aa00",
            "ClientId":  "00001111-aaaa-2222-bbbb-3333cccc4444"
                },

"/subscriptions/ContosoID/resourcegroups/ContosoLab/providers/Microsoft.ManagedId
entity/userAssignedIdentities/ContosoUAMI2":  {
            "PrincipalId":  "00a000aa-00a0-00aa-00aa-0a0aa000aa00",
            "ClientId":  "00001111-aaaa-2222-bbbb-3333cccc4444"
                }
    }
}
```

Adición mediante una plantilla de ARM

A continuación, se proporcionan los pasos de ejemplo y la sintaxis.

Sintaxis de plantillas

La sintaxis de la plantilla de ejemplo siguiente habilita una identidad administrada asignada por el sistema si aún no está habilitada y asigna dos identidades administradas asignadas por el usuario existentes a la cuenta de Automation existente.

JSON

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "automationAccountName": {
     "defaultValue": "YourAutomationAccount",
      "type": "String",
      "metadata": {
        "description": "Automation account name"
      }
    },
    "userAssignedOne": {
     "defaultValue": "userAssignedOne",
      "type": "String",
      "metadata": {
        "description": "User-assigned managed identity"
      }
      },
    "userAssignedTwo": {
     "defaultValue": "userAssignedTwo",
      "type": "String",
      "metadata": {
        "description": "User-assigned managed identity"
      }
      }
   },
  "resources": [
    {
      "type": "Microsoft.Automation/automationAccounts",
      "apiVersion": "2020-01-13-preview",
      "name": "[parameters('automationAccountName')]",
      "location": "[resourceGroup().location]",
      "identity": {
        "type": "SystemAssigned, UserAssigned",
        "userAssignedIdentities": {
          "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',parameters('userAssign
edOne'))]": {},
          "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',parameters('userAssign
edTwo'))]": {}
        }
      },
      "properties": {
        "sku": {
          "name": "Basic"
```

```
        },
        "encryption": {
          "keySource": "Microsoft.Automation",
          "identity": {}
        }
      }
    }
  ]
}
```

Ejemplo

Lleve a cabo los siguiente pasos.

1. Copie y pegue la plantilla en un archivo llamado `template_ua.json`. Guarde el archivo en el equipo local o en una cuenta de almacenamiento de Azure.

2. Revise el valor de la variable siguiente y, a continuación, ejecute:

   PowerShell

   ```
   $templateFile = "path\template_ua.json"
   ```

3. Use el cmdlet de PowerShell New-AzResourceGroupDeployment para implementar la plantilla.

   PowerShell

   ```
   New-AzResourceGroupDeployment `
       -Name "UserAssignedDeployment" `
       -ResourceGroupName $resourceGroup `
       -TemplateFile $templateFile `
       -automationAccountName $automationAccount `
       -userAssignedOne $userAssignedOne `
       -userAssignedTwo $userAssignedTwo
   ```

   El comando no producirá ninguna salida; sin embargo, puede usar el código siguiente para comprobarlo:

   PowerShell

   ```
   (Get-AzAutomationAccount `
   -ResourceGroupName $resourceGroup `
   -Name $automationAccount).Identity | ConvertTo-Json
   ```

   La salida tendrá un aspecto similar al que se muestra en el ejemplo anterior de la API REST.

Asignación de un rol a una identidad administrada asignada por el usuario

Una cuenta de Automation puede usar su identidad administrada asignada por el usuario para obtener tokens para acceder a otros recursos protegidos por Microsoft Entra ID, como Azure Key Vault. Estos tokens no representan a ningún usuario específico de la aplicación, sino que representan la aplicación que accede al recurso. En este caso, por ejemplo, el token representa una cuenta de Automation.

Para poder usar la identidad administrada asignada por el usuario para la autenticación, configure el acceso para esa identidad en el recurso de Azure en el que planea usar la identidad. Para completar esta tarea, asigne el rol adecuado a esa identidad en el recurso Azure de destino.

Siga la entidad de seguridad con menos privilegios y asigne cuidadosamente solo los permisos necesarios para ejecutar el runbook. Por ejemplo, si la cuenta de Automation solo es necesaria para iniciar o detener una máquina virtual de Azure, los permisos asignados a la cuenta de ejecución o la identidad administrada solo deben ser para iniciar o detener la máquina virtual. De manera similar, si un runbook lee del almacenamiento de blobs, asigne permisos de solo lectura.

En este ejemplo se usa Azure PowerShell para mostrar cómo asignar el rol Colaborador en la suscripción al recurso de destino Azure. El rol Colaborador se usa como ejemplo y puede ser necesario o no en su caso. Como alternativa, también puede asignar el rol al recurso de Azure de destino en el portal Azure.

PowerShell

```
New-AzRoleAssignment `
    -ObjectId <automation-Identity-object-id> `
    -Scope "/subscriptions/<subscription-id>" `
    -RoleDefinitionName "Contributor"
```

Comprobación de la asignación de roles a una identidad administrada por el usuario

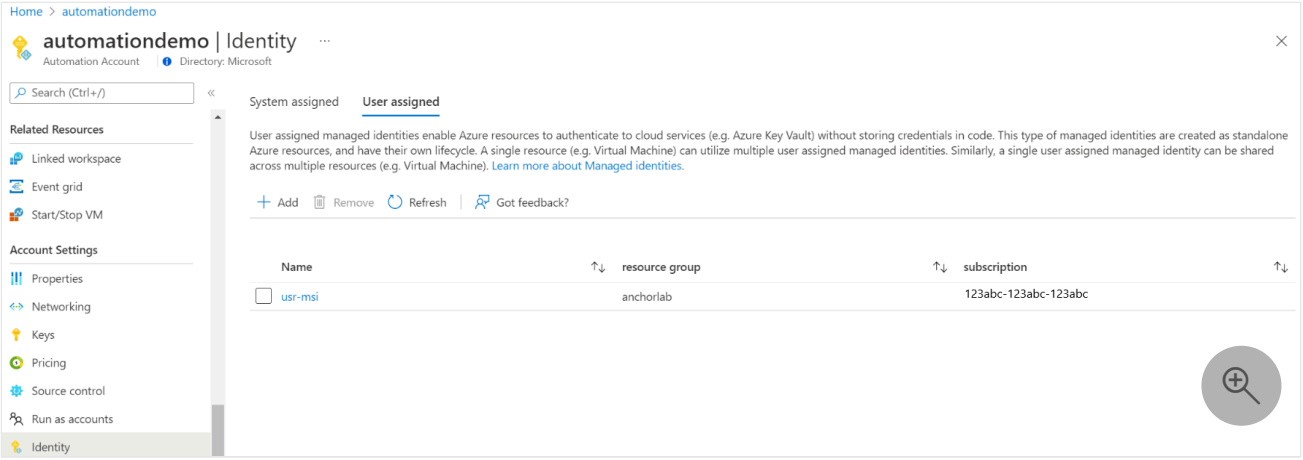
Para comprobar un rol en una identidad administrada asignada por el usuario de la cuenta de Automation, siga estos pasos:

1. Inicie sesión en el portal Azure.

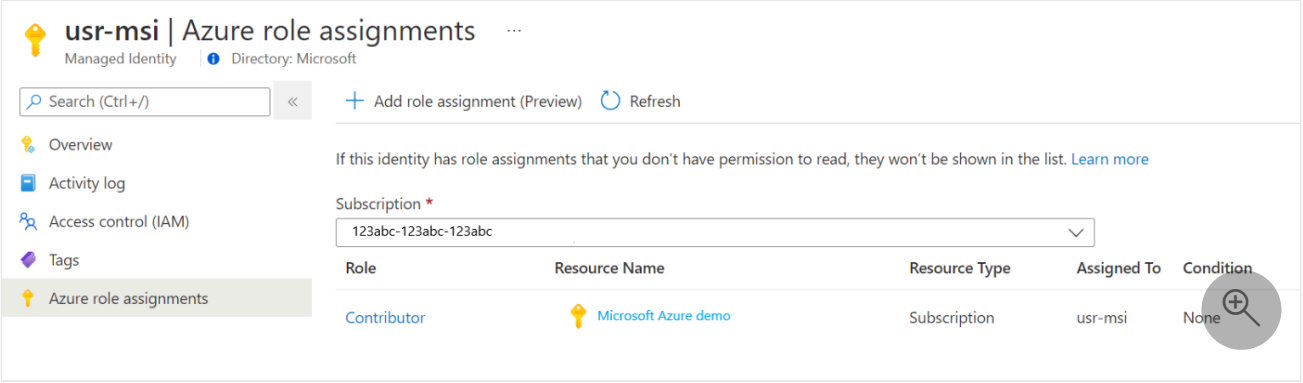
2. Vaya a su cuenta de Automation.

3. En **Configuración de la cuenta**, seleccione **Identidad**, **Usuario asignado**.

4. Haga clic en **Nombre de identidad asignado por el usuario**.



Si los roles ya están asignados a la identidad administrada asignada por el usuario seleccionada, puede ver una lista de asignaciones de roles. Esta lista incluye todas las asignaciones de roles que tiene permiso para leer.



5. Para cambiar la suscripción, haga clic en la lista desplegable **Suscripción** y seleccione la suscripción adecuada.

6. Haga clic en **Agregar asignación de roles (versión preliminar)**

7. En la lista desplegable, seleccione el conjunto de recursos que aplica la asignación de roles: **Suscripción**, **Grupo de recursos**, **Rol** y **Ámbito**.
   Si no tiene la asignación de roles, puede ver los permisos de escritura del ámbito seleccionado como mensaje insertado.

8. En la lista desplegable **Rol** , seleccione un rol como *Colaborador de máquina virtual*.

9. Haga clic en **Guardar**.

Transcurridos unos minutos, se asigna el rol a la identidad administrada en el ámbito seleccionado.

Autenticación del acceso con una identidad administrada asignada por el usuario

Después de habilitar la identidad administrada asignada por el usuario para la cuenta de Automation y de conceder acceso a una identidad al recurso de destino, puede especificar esa identidad en los runbooks para los recursos que admiten la identidad administrada. Para soporte de identidad, use el cmdlet Connect-AzAccount.

PowerShell

```
# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process

# Connect to Azure with user-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity -AccountId <user-assigned-identity-
ClientId>).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext
```

Generación de un token de acceso sin usar cmdlets de Azure

Para los puntos de conexión HTTP, asegúrese de lo siguiente.

- El encabezado de metadatos debe estar presente y establecido en "true".
- Se debe pasar un recurso junto con la solicitud, como parámetro de consulta para una solicitud GET y como datos de formulario para una solicitud POST.
- Se debe establecer el valor de la variable de entorno IDENTITY_HEADER en X-IDENTITY-HEADER.
- El tipo de contenido de la solicitud Post debe ser `application/x-www-form-urlencoded`.

Obtención de un token de acceso para la identidad administrada asignada por el usuario mediante HTTP Get

PowerShell

```
$resource= "?resource=https://management.azure.com/"
$client_id="&client_id=<ClientId of USI>"
$url = $env:IDENTITY_ENDPOINT + $resource + $client_id
$Headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"
$Headers.Add("Metadata", "True")
$headers.Add("X-IDENTITY-HEADER", $env:IDENTITY_HEADER)
$accessToken = Invoke-RestMethod -Uri $url -Method 'GET' -Headers $Headers
Write-Output $accessToken.access_token
```

Obtención de un token de acceso para la identidad administrada asignada por el usuario mediante HTTP Post

PowerShell

```
$url = $env:IDENTITY_ENDPOINT
$headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"
$headers.Add("Metadata", "True")
$headers.Add("X-IDENTITY-HEADER", $env:IDENTITY_HEADER)
$body = @{'resource'='https://management.azure.com/'
'client_id'='<ClientId of USI>'}
$accessToken = Invoke-RestMethod $url -Method 'POST' -Headers $headers -
ContentType 'application/x-www-form-urlencoded' -Body $body
Write-Output $accessToken.access_token
```

Uso de la identidad administrada asignada por el usuario en Azure PowerShell

PowerShell

```
Write-Output "Connecting to azure via  Connect-AzAccount -Identity -
AccountId <ClientId of USI>"
Connect-AzAccount -Identity -AccountId <ClientId of USI>
Write-Output "Successfully connected with Automation account's Managed Identity"
```

```
Write-
Output "Trying to fetch value from key vault using User Assigned Managed identity. Mak
e sure you have given correct access to Managed Identity"
$secret = Get-AzKeyVaultSecret -VaultName '<KVname>' -Name '<KeyName>'
$ssPtr = [System.Runtime.InteropServices.Marshal]::SecureStringToBSTR($secret.SecretVa
lue)
try {
  $secretValueText = [System.Runtime.InteropServices.Marshal]::PtrToStringBSTR($ssPtr)

    Write-Output $secretValueText
} finally {
    [System.Runtime.InteropServices.Marshal]::ZeroFreeBSTR($ssPtr)
}
```

Uso de la identidad administrada asignada por el usuario en Python Runbook

Python

```python
#!/usr/bin/env python3
import os
import requests

resource = "?resource=https://management.azure.com/"
client_id = "&client_id=<ClientId of USI>"
endPoint = os.getenv('IDENTITY_ENDPOINT')+ resource +client_id
identityHeader = os.getenv('IDENTITY_HEADER')
payload={}
headers = {
  'X-IDENTITY-HEADER': identityHeader,
  'Metadata': 'True'
}
response = requests.request("GET", endPoint, headers=headers, data=payload)
print(response.text)
```

Pasos siguientes

- Para solucionar problemas relacionados con la identidad administrada asignada por el usuario, consulte Troubleshoot Azure Automation problemas de identidad administrada.
- Si necesita deshabilitar una identidad administrada, consulte Disable la identidad administrada de la cuenta de Azure Automation.
- Para obtener información general sobre la seguridad de las cuentas de Azure Automation, consulte Información general sobre la autenticación de la cuenta de Automation.

Last updated on 20/04/2026

Deshabilitar la identidad administrada asignada por el sistema para la cuenta de Azure Automation

Puede deshabilitar una identidad administrada asignada por el sistema en Azure Automation mediante el portal de Azure o mediante la API REST.

Deshabilitación mediante el portal de Azure

Puede deshabilitar la identidad administrada asignada por el sistema desde el portal de Azure independientemente de cómo se configuró originalmente la identidad administrada asignada por el sistema.

1. Inicie sesión en el portal Azure.
2. Vaya a la cuenta de Automation y, en **Configuración de la cuenta**, seleccione **Identidad**.
3. En la pestaña **Asignada por el sistema**, en el botón **Estado**, seleccione **Desactivado** y, a continuación, seleccione **Guardar**. Cuando se le solicite confirmación, seleccione **Sí**.

La identidad administrada asignada por el sistema se deshabilitará y ya no tendrá acceso al recurso de destino.

Deshabilitación mediante la API REST

A continuación, se proporcionan los pasos de ejemplo y la sintaxis.

Cuerpo de la solicitud

El cuerpo de la solicitud siguiente deshabilita la identidad administrada asignada por el sistema y quita todas las identidades administradas asignadas por el usuario mediante el método HTTP **PATCH**.

JSON

```
{
 "identity": {
   "type": "None"
  }
```

```
}
```

Si hay varias identidades asignadas por el usuario definidas, para conservarlas y quitar solo la identidad asignada por el sistema, debe especificar cada identidad asignada por el usuario mediante una lista delimitada por comas. En el ejemplo siguiente, se usa el método HTTP **PATCH**.

JSON

```
{
"identity" : {
    "type": "UserAssigned",
    "userAssignedIdentities": {
        "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/resourceGroupName/providers/Microsoft.ManagedIdentity/user
AssignedIdentities/firstIdentity": {},
        "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/resourceGroupName/providers/Microsoft.ManagedIdentity/user
AssignedIdentities/secondIdentity": {}
        }
    }
}
```

A continuación, se muestra la URI de la solicitud de la API REST del servicio para enviar la petición PATCH.

HTTP

```
PATCH https://management.azure.com/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/resource-group-
name/providers/Microsoft.Automation/automationAccounts/automation-account-name?api-
version=2020-01-13-preview
```

Ejemplo

Lleve a cabo los pasos siguientes.

1. Copie y pegue el cuerpo de la solicitud, en función de la operación que quiera realizar, en un archivo denominado `body_remove_sa.json`. Guarde el archivo en el equipo local o en una cuenta de almacenamiento de Azure.

2. Inicie sesión en Azure interactivamente con el cmdlet Connect-AzAccount y siga las instrucciones.

   PowerShell

```
# Sign in to your Azure subscription
$sub = Get-AzSubscription -ErrorAction SilentlyContinue
if(-not($sub))
{
    Connect-AzAccount
}

# If you have multiple subscriptions, set the one to use
# Select-AzSubscription -SubscriptionId "<SUBSCRIPTIONID>"
```

3. Proporcione un valor adecuado para las variables y, después, ejecute el script.

PowerShell

```
$subscriptionID = "subscriptionID"
$resourceGroup = "resourceGroupName"
$automationAccount = "automationAccountName"
$file = "path\body_remove_sa.json"
```

4. Este ejemplo usa el cmdlet de PowerShell Invoke-RestMethod para enviar una solicitud PATCH a la cuenta de Automation.

PowerShell

```
# build URI
$URI =
"https://management.azure.com/subscriptions/$subscriptionID/resourceGroups/$resou
rceGroup/providers/Microsoft.Automation/automationAccounts/$automationAccount`?
api-version=2020-01-13-preview"

# build body
$body = Get-Content $file

# obtain access token
$azContext = Get-AzContext
$azProfile =
[Microsoft.Azure.Commands.Common.Authentication.Abstractions.AzureRmProfileProvid
er]::Instance.Profile
$profileClient = New-Object -TypeName
Microsoft.Azure.Commands.ResourceManager.Common.RMProfileClient -ArgumentList
($azProfile)
$token = $profileClient.AcquireAccessToken($azContext.Subscription.TenantId)
$authHeader = @{
    'Content-Type'='application/json'
    'Authorization'='Bearer ' + $token.AccessToken
}

# Invoke the REST API
Invoke-RestMethod -Uri $URI -Method PATCH -Headers $authHeader -Body $body

# Confirm removal
```

```
(Get-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount).Identity.Type
```

En función de la sintaxis que se utilice, la salida será `UserAssigned` o en blanco.

Pasos siguientes

- Para obtener más información sobre cómo habilitar identidades administradas en Azure Automation, consulte Enable and use managed identity for Automation.
- Para información general acerca de la seguridad de la cuenta de Automation, consulte Introducción a la autenticación de cuentas de Automation.
- Para solucionar problemas relacionados con la identidad administrada, consulte Troubleshoot Azure Automation problemas de identidad administrada.

Last updated on 20/04/2026

Eliminación de identidades administradas asignadas por el usuario de una cuenta de Azure Automation

Puede eliminar una identidad administrada asignada por el usuario de Azure Automation mediante Azure Portal, PowerShell, la API REST de Azure o una plantilla de Azure Resource Manager (ARM).

Eliminación mediante Azure Portal

Puede eliminar la identidad administrada asignada por el usuario desde Azure Portal independientemente de cómo la haya agregado originalmente.

1. Inicie sesión en Azure Portal.
2. Vaya a la cuenta de Automation y, en **Configuración de la cuenta**, seleccione **Identidad**.
3. Seleccione la pestaña **Usuario asignado**.
4. Seleccione la identidad administrada asignada por el usuario que se va a quitar de la lista.
5. Seleccione **Quitar**. Cuando se le solicite confirmación, seleccione **Sí**.

La identidad administrada asignada por el usuario se quitará y ya no tendrá acceso al recurso de destino.

Eliminación con PowerShell

Use el cmdlet de PowerShell Set-AzAutomationAccount para quitar todas las identidades administradas asignadas por el usuario y conservar una identidad administrada asignada por el sistema existente.

1. Inicie sesión en Azure de forma interactiva con el cmdlet Connect-AzAccount y siga las instrucciones.

PowerShell

```
# Sign in to your Azure subscription
$sub = Get-AzSubscription -ErrorAction SilentlyContinue
if(-not($sub))
{
    Connect-AzAccount
}
```

2. Proporcione un valor adecuado para las variables y, después, ejecute el script.

PowerShell

```
$resourceGroup = "resourceGroupName"
$automationAccount = "automationAccountName"
```

3. Ejecute Set-AzAutomationAccount.

PowerShell

```
# Removes all UAs, keeps SA
$output = Set-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount `
    -AssignSystemIdentity

$output.identity.Type
```

El resultado será `SystemAssigned`.

Eliminación mediante la API REST

Puede eliminar una identidad administrada asignada por el usuario de la cuenta de Automation mediante el siguiente ejemplo y llamada a la API REST.

Cuerpo de la solicitud

Escenario: la identidad administrada asignada por el sistema está habilitada o se va a habilitar. Se va a quitar una de las muchas identidades administradas asignadas por el usuario. En este ejemplo se quita una identidad administrada asignada por el usuario denominada `firstIdentity` mediante el método HTTP **PATCH**.

JSON

```
{
  "identity": {
    "type": "SystemAssigned, UserAssigned",
    "userAssignedIdentities": {
      "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e/resourceGroups/resource-group-name/providers/Microsoft.ManagedIdentity/userAssignedIdentities/firstIdentity": null
    }
  }
}
```

Escenario: la identidad administrada asignada por el sistema está habilitada o se va a habilitar. Todas las identidades administradas asignadas por el usuario se quitarán mediante el método HTTP **PUT**.

JSON

```
{
  "identity": {
    "type": "SystemAssigned"
  }
}
```

Escenario: la identidad administrada asignada por el sistema está deshabilitada o se va a deshabilitar. Se va a quitar una de las muchas identidades administradas asignadas por el usuario. En este ejemplo se quita una identidad administrada asignada por el usuario denominada `firstIdentity` mediante el método HTTP **PATCH**.

JSON

```
{
  "identity": {
    "type": "UserAssigned",
    "userAssignedIdentities": {
      "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e/resourceGroups/resource-group-
name/providers/Microsoft.ManagedIdentity/userAssignedIdentities/firstIdentity": null
    }
  }
}
```

Escenario: la identidad administrada asignada por el sistema está deshabilitada o se va a deshabilitar. Todas las identidades administradas asignadas por el usuario se quitarán mediante el método HTTP **PUT**.

JSON

```
{
  "identity": {
    "type": "None"
  }
}
```

A continuación, se muestra el identificador URI de solicitud de la API REST del servicio para enviar la solicitud PATCH.

HTTP

```
https://management.azure.com/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e/resourceGroups/resource-group-name/providers/Microsoft.Automation/automationAccounts/automation-account-name?api-version=2020-01-13-preview
```

Ejemplo

Lleve a cabo los pasos siguientes.

1. Copie y pegue el cuerpo de la solicitud, en función de la operación que quiera realizar, en un archivo denominado `body_remove_ua.json`. Realice las modificaciones necesarias y guarde el archivo en el equipo local o en una cuenta de Azure Storage.

2. Inicie sesión en Azure de forma interactiva con el cmdlet Connect-AzAccount y siga las instrucciones.

   PowerShell

   ```
   # Sign in to your Azure subscription
   $sub = Get-AzSubscription -ErrorAction SilentlyContinue
   if(-not($sub))
   {
       Connect-AzAccount -Subscription
   }
   ```

3. Proporcione un valor adecuado para las variables y, después, ejecute el script.

   PowerShell

   ```
   $subscriptionID = "subscriptionID"
   $resourceGroup = "resourceGroupName"
   $automationAccount = "automationAccountName"
   $file = "path\body_remove_ua.json"
   ```

4. Este ejemplo usa el cmdlet de PowerShell Invoke-RestMethod para enviar una solicitud PATCH a la cuenta de Automation.

   PowerShell

   ```
   # build URI
   $URI =
   "https://management.azure.com/subscriptions/$subscriptionID/resourceGroups/$resourceGroup/providers/Microsoft.Automation/automationAccounts/$automationAccount`?api-version=2020-01-13-preview"

   # build body
   $body = Get-Content $file
   ```

```
# obtain access token
$azContext = Get-AzContext
$azProfile =
[Microsoft.Azure.Commands.Common.Authentication.Abstractions.AzureRmProfileProv
ider]::Instance.Profile
$profileClient = New-Object -TypeName
Microsoft.Azure.Commands.ResourceManager.Common.RMProfileClient -ArgumentList
($azProfile)
$token = $profileClient.AcquireAccessToken($azContext.Subscription.TenantId)
$authHeader = @{
    'Content-Type'='application/json'
    'Authorization'='Bearer ' + $token.AccessToken
}

# Invoke the REST API
Invoke-RestMethod -Uri $URI -Method PATCH -Headers $authHeader -Body $body

# Confirm removal
(Get-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount).Identity.Type
```

En función de la sintaxis que se utilice, el resultado será `SystemAssignedUserAssigned`, `SystemAssigned`, `UserAssigned` o en blanco.

Eliminación mediante la plantilla de Azure Resource Manager

Si ha agregado la identidad administrada asignada por el usuario para la cuenta de Automation mediante una plantilla de Azure Resource Manager, puede quitar la identidad administrada asignada por el usuario modificando la plantilla y ejecutándola de nuevo.

Escenario: la identidad administrada asignada por el sistema está habilitada o se va a habilitar. Se va a quitar una de las dos identidades administradas asignadas por el usuario. Este fragmento de código de sintaxis quita **todas** las identidades administradas asignadas por el usuario, **excepto** la que se pasa como parámetro a la plantilla.

JSON

```
...
"identity": {
    "type": "SystemAssigned, UserAssigned",
    "userAssignedIdentities": {
        "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',parameters('userAssi
gnedOne'))]": {}
    }
```

```
},
...
```

Escenario: la identidad administrada asignada por el sistema está habilitada o se va a habilitar. Se quitarán todas las identidades administradas asignadas por el usuario.

JSON

```
...
"identity": {
    "type": "SystemAssigned"
},
...
```

Escenario: la identidad administrada asignada por el sistema está deshabilitada o se va a deshabilitar. Se va a quitar una de las dos identidades administradas asignadas por el usuario. Este fragmento de código de sintaxis quita **todas** las identidades administradas asignadas por el usuario, **excepto** la que se pasa como parámetro a la plantilla.

JSON

```
...
"identity": {
    "type": "UserAssigned",
    "userAssignedIdentities": {
        "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',parameters('userAssi
gnedOne'))]": {}
    }
},
...
```

Use el cmdlet Get-AzAutomationAccount para realizar la comprobación. En función de la sintaxis que se utilice, el resultado será `SystemAssignedUserAssigned`, `SystemAssigned` o `UserAssigned`.

PowerShell

```
(Get-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount).Identity.Type
```

Pasos siguientes

- Para más información sobre cómo habilitar las identidades administradas en Azure Automation, consulte Habilitación y uso de identidades administradas en Automation.

- Para información general acerca de la seguridad de la cuenta de Automation, consulte Introducción a la autenticación de cuentas de Automation.

Last updated on 17/11/2025

Eliminar una cuenta Run As de Azure Automation

Artículo • 30/04/2025

> **Importante**
>
> Las cuentas de ejecución de Azure Automation, incluidas las cuentas de ejecución clásicas, se retiraron el **30 de septiembre de 2023** y se reemplazaron por identidades administradas. Ya no se podrán crear o renovar cuentas de ejecución a través de Azure Portal. Para obtener más información, consulte migrar desde una cuenta de ejecución a una identidad administrada.

Las cuentas Run As en Azure Automation proporcionan autenticación para administrar recursos en el modelo de implementación de Azure Resource Manager o Azure clásico, utilizando runbooks de Automation y otras características de Automation. En este artículo se describe cómo eliminar una cuenta de Run As o una cuenta de Classic Run As. Al realizar esta acción, la cuenta de Automation se conserva.

Permisos para cuentas de ejecución y cuentas de ejecución clásicas

Para configurar, actualizar o eliminar una cuenta de ejecución y una cuenta de ejecución clásica, debe ser:

- Un propietario de la aplicación Microsoft Entra para la cuenta de ejecución.

  o

- Un miembro de uno de los siguientes roles de Microsoft Entra.
  - Administrador de aplicaciones
  - Administrador de aplicaciones en la nube

Puede eliminar la entidad de servicio; desde el portal de **Microsoft Entra ID** portal >**Registros de aplicaciones**> busque y seleccione el nombre de la cuenta de Automation y, en la página **Información general**, seleccione **Eliminar**.

Pasos siguientes

- Usar la identidad administrada asignada por el sistema.
- Uso de la identidad administrada asignada por el usuario.

Migración desde una cuenta de ejecución existente a identidades administradas

> **ⓘ Importante**
>
> Las cuentas de ejecución de Azure Automation, incluidas las cuentas de ejecución clásicas, se retiraron el **30 de septiembre de 2023** y se reemplazaron por identidades administradas. Ya no se podrán crear o renovar cuentas de ejecución a través de Azure Portal.

Para obtener más información sobre la cadencia de migración y la escala de tiempo de soporte técnico para la creación de cuentas de ejecución y la renovación de certificados, consulte las preguntas más frecuentes.

Las cuentas de ejecución de Azure Automation proporcionan autenticación para administrar recursos implementados mediante Azure Resource Manager o el modelo de implementación clásica. Cada vez que se crea una cuenta de ejecución, se registra una aplicación de Microsoft Entra y se genera un certificado autofirmado. El certificado es válido durante un mes. La renovación del certificado cada mes antes de que expire mantiene la cuenta de Automation funcionando, pero agrega sobrecarga.

Ahora puede configurar las cuentas de Automation para que usen una identidad administrada, que es la opción predeterminada al crear una cuenta de Automation. Con esta característica, una cuenta de Automation puede autenticarse en recursos de Azure sin necesidad de intercambiar credenciales. Una identidad administrada quita la sobrecarga de renovar el certificado o administrar la entidad de servicio.

Una identidad administrada puede ser asignada por el sistema o asignada por el usuario. Cuando se crea una nueva cuenta de Automation, se habilita una identidad administrada asignada por el sistema.

Requisitos previos

Antes de migrar desde una cuenta de ejecución o una cuenta de ejecución clásica a una identidad administrada:

1. Cree una identidad administrada asignada por el sistema o asignada por el usuario, o cree ambos tipos. Para más información sobre las diferencias entre ellas, consulte Tipos de identidad administrada.

> **Nota:**
>
> - Las identidades asignadas por el usuario solo se admiten para trabajos en la nube. No es posible usar la identidad administrada por el usuario de la cuenta de Automation en una instancia de Hybrid Runbook Worker. Para usar trabajos híbridos, debe crear identidades asignadas por el sistema.
> - Hay dos maneras de usar identidades administradas en scripts de Hybrid Runbook Worker: la identidad administrada asignada por el sistema de la cuenta de Automation *o* la identidad administrada de la máquina virtual (VM) de una máquina virtual de Azure que se ejecuta como Hybrid Runbook Worker.
> - La identidad administrada asignada por el usuario de la máquina virtual y la identidad administrada asignada por el sistema de la máquina virtual *no* funcionarán en una cuenta de Automation configurada con la identidad administrada de una cuenta de Automation. Al habilitar la identidad administrada de la cuenta de Automation, solo puede usar la identidad administrada asignada por el sistema de la cuenta de Automation y no la identidad administrada de la máquina virtual. Para más información, consulte Uso de la autenticación de runbook con identidades administradas.

2. Asigne el mismo rol a la identidad administrada para acceder a los recursos de Azure que coincidan con la cuenta de ejecución. Use este script para habilitar la identidad asignada por el sistema en una cuenta de Automation y asignar el mismo conjunto de permisos presentes en la cuenta de ejecución de Azure Automation a la identidad asignada por el sistema de la cuenta de Automation.

   Por ejemplo, si la cuenta de Automation solo es necesaria para iniciar o detener una VM de Azure, los permisos asignados a la cuenta de ejecución solo deben ser para iniciar o detener la VM. De manera similar, debe asignar permisos de solo lectura si un runbook lee de Azure Blob Storage. Para obtener más información, consulte Directrices de seguridad de Azure Automation.

3. Si usa cuentas de ejecución clásicas, asegúrese de haber migrado recursos implementados mediante el modelo de implementación clásica a Azure Resource Manager.

4. Use este script para averiguar qué cuentas de Automation usan una cuenta de ejecución. Si las cuentas de Azure Automation contienen una cuenta de ejecución, tiene asignado el rol de colaborador integrado de forma predeterminada. Puede usar el script

para comprobar las cuentas de ejecución de Azure Automation y determinar si su asignación de roles es la predeterminada o si se ha cambiado a otra definición de rol.

5. Use este script para averiguar si todos los runbooks de la cuenta de Automation usan la cuenta de ejecución.

Migración desde una cuenta de ejecución de Automation a una identidad administrada

Para migrar de una cuenta de ejecución clásica o una cuenta de ejecución de Automation a una identidad administrada para la autenticación del runbook, siga estos pasos:

1. Cambie el código del runbook para usar una identidad administrada.

   Se recomienda probar la identidad administrada para comprobar si el runbook funciona según lo previsto mediante la creación de una copia del runbook de producción. Actualice el código del runbook de prueba para autenticarse mediante la identidad administrada. Este método garantiza que no invalide `AzureRunAsConnection` en el runbook de producción ni interrumpa la instancia de Automation existente. Después de asegurarse de que el código del runbook se ejecuta según lo previsto a través de la identidad administrada, actualice el runbook de producción para usar la identidad administrada.

   Para compatibilidad con las identidades administradas, use el cmdlet `Connect-AzAccount`. Para más información sobre este cmdlet, consulte Connect-AzAccount en la referencia de PowerShell.

   - Si usa módulos `Az`, actualice a la versión más reciente siguiendo los pasos descritos en el artículo Actualización de los módulos de Azure PowerShell.
   - Si usa módulos de AzureRM, actualice `AzureRM.Profile` a la versión más reciente y reemplácelo mediante el cmdlet `Add-AzureRMAccount` por `Connect-AzureRMAccount -Identity`.

   Para comprender los cambios en el código de runbook que son necesarios para poder usar identidades administradas, use los scripts de muestra.

2. Cuando esté seguro de que el runbook se ejecuta correctamente mediante identidades administradas, puede eliminar de forma segura la cuenta de ejecución si ningún otro runbook usa esa cuenta.

Muestras de scripts

Los ejemplos siguientes de scripts de runbook capturan los recursos de Resource Manager mediante la cuenta de ejecución (entidad de servicio) y la identidad administrada. Observaría la diferencia en el código de runbook al principio del runbook, donde se autentica en el recurso.

Identidad administrada asignada por el sistema

> **Nota:**
>
> Habilite los permisos de RBAC adecuados para la identidad del sistema de esta cuenta de Automation. De lo contrario, se podría producir un error en el runbook.

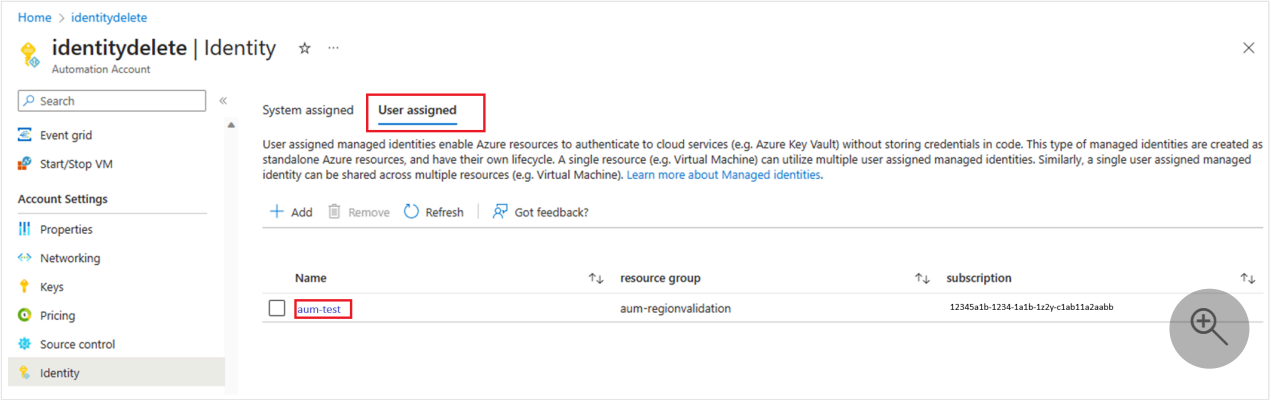
PowerShell

```
try
{
    "Logging in to Azure..."
    Connect-AzAccount -Identity
}
catch {
    Write-Error -Message $_.Exception
    throw $_.Exception
}

#Get all Resource Manager resources from all resource groups
$ResourceGroups = Get-AzResourceGroup

foreach ($ResourceGroup in $ResourceGroups)
{
    Write-Output ("Showing resources in resource group " +
$ResourceGroup.ResourceGroupName)
    $Resources = Get-AzResource -ResourceGroupName
$ResourceGroup.ResourceGroupName
    foreach ($Resource in $Resources)
    {
        Write-Output ($Resource.Name + " of type " +  $Resource.ResourceType)
    }
    Write-Output ("")
}
```

Ver el identificador de cliente de la identidad asignada por el usuario

1. En la cuenta de Automation, en **Configuración de la cuenta**, seleccione **Identidad**.

2. En la pestaña **Usuario asignado**, seleccione la identidad asignada por el usuario.



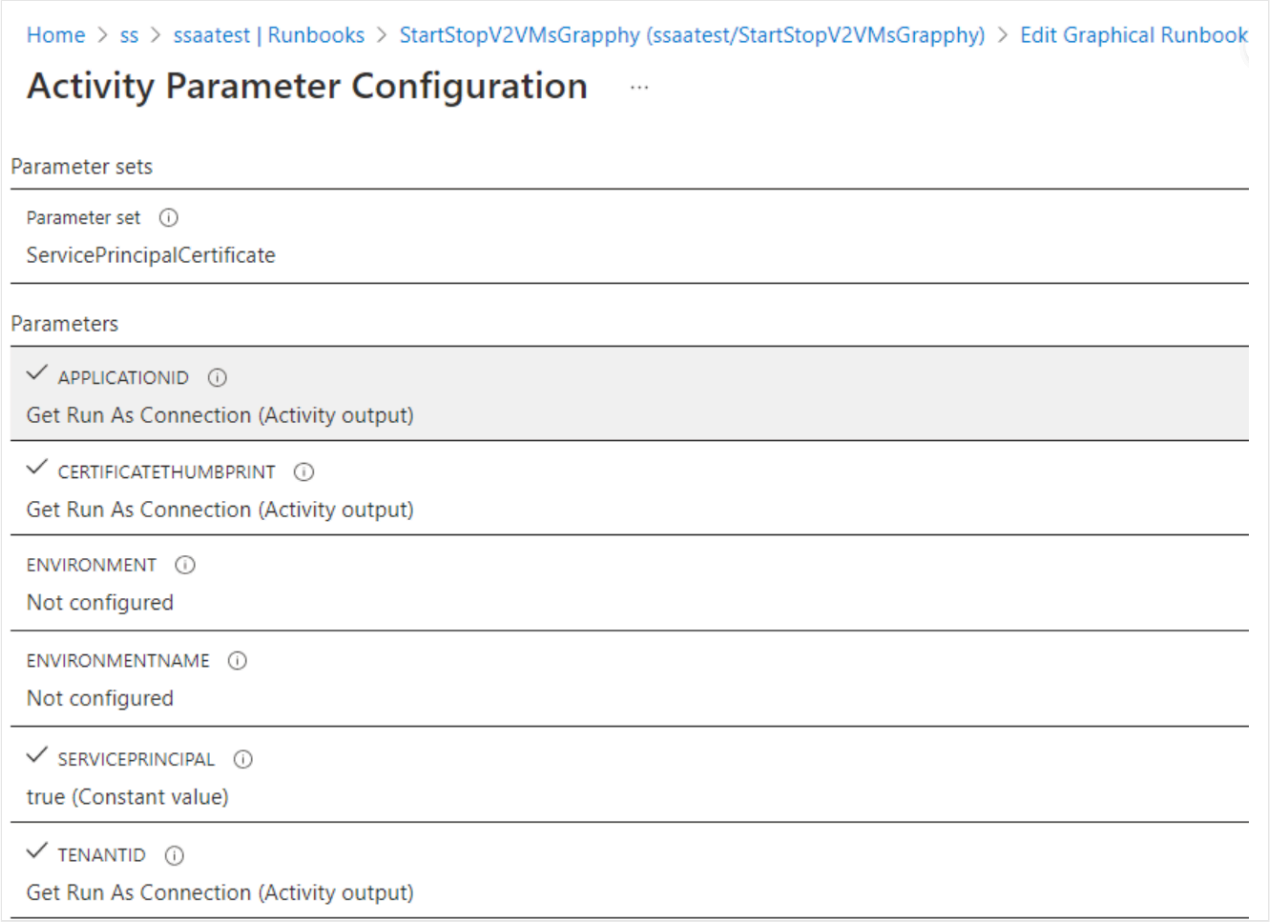
3. Vaya a **Información general**>**Essentials** para ver el **id. de cliente**.

Runbooks gráficos

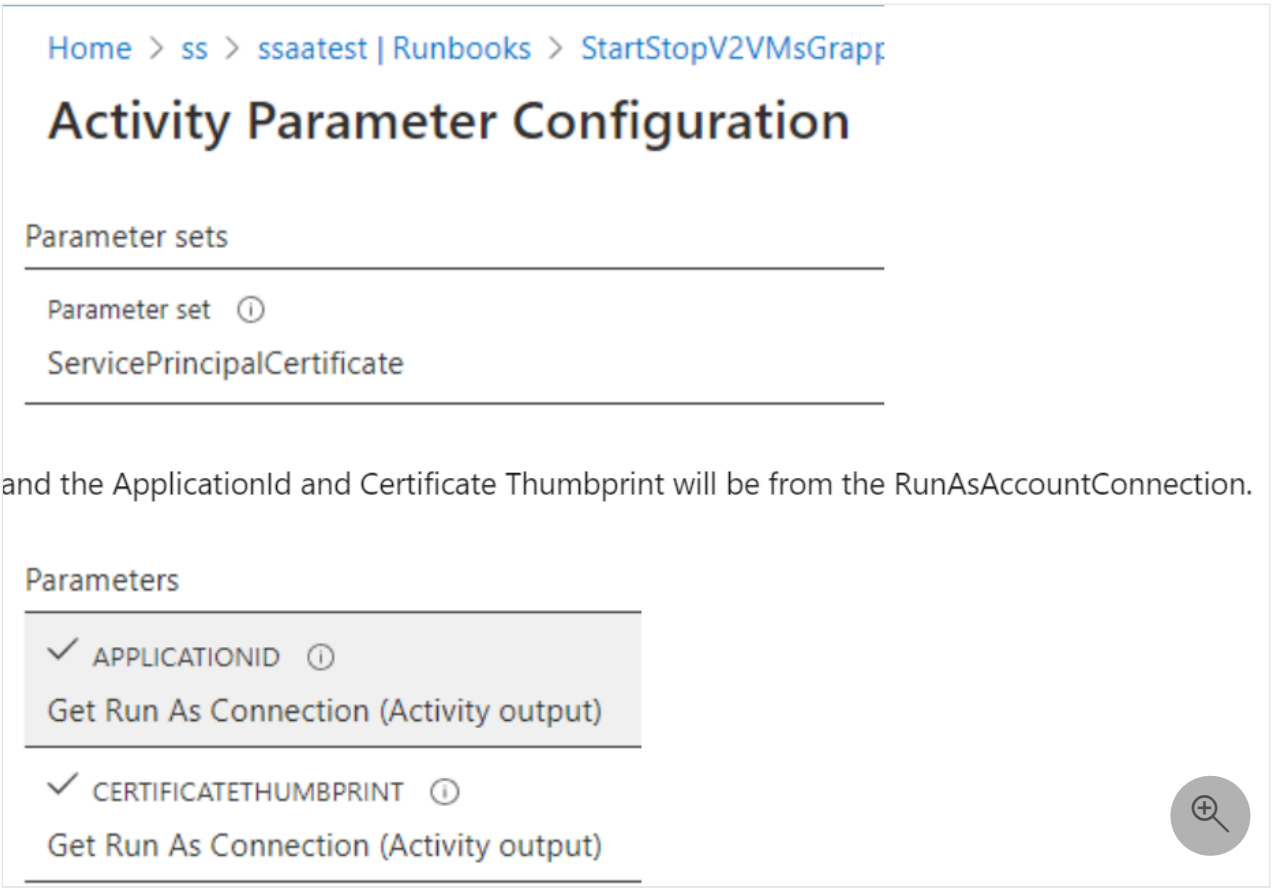
Compruebe si se usa una cuenta de ejecución en runbooks gráficos.

1. Compruebe cada una de las actividades del runbook para ver si usa la cuenta de ejecución cuando llama a cualquier cmdlet o alias de inicio de sesión, como `Add-AzRmAccount/Connect-AzRmAccount/Add-AzAccount/Connect-AzAccount`.

2. Examine los parámetros que usa el cmdlet.



Para su uso con la cuenta de ejecución, el cmdlet usa el parámetro `ServicePrincipalCertificate` establecido en `ApplicationId`. `CertificateThumbprint` será de `RunAsAccountConnection`.

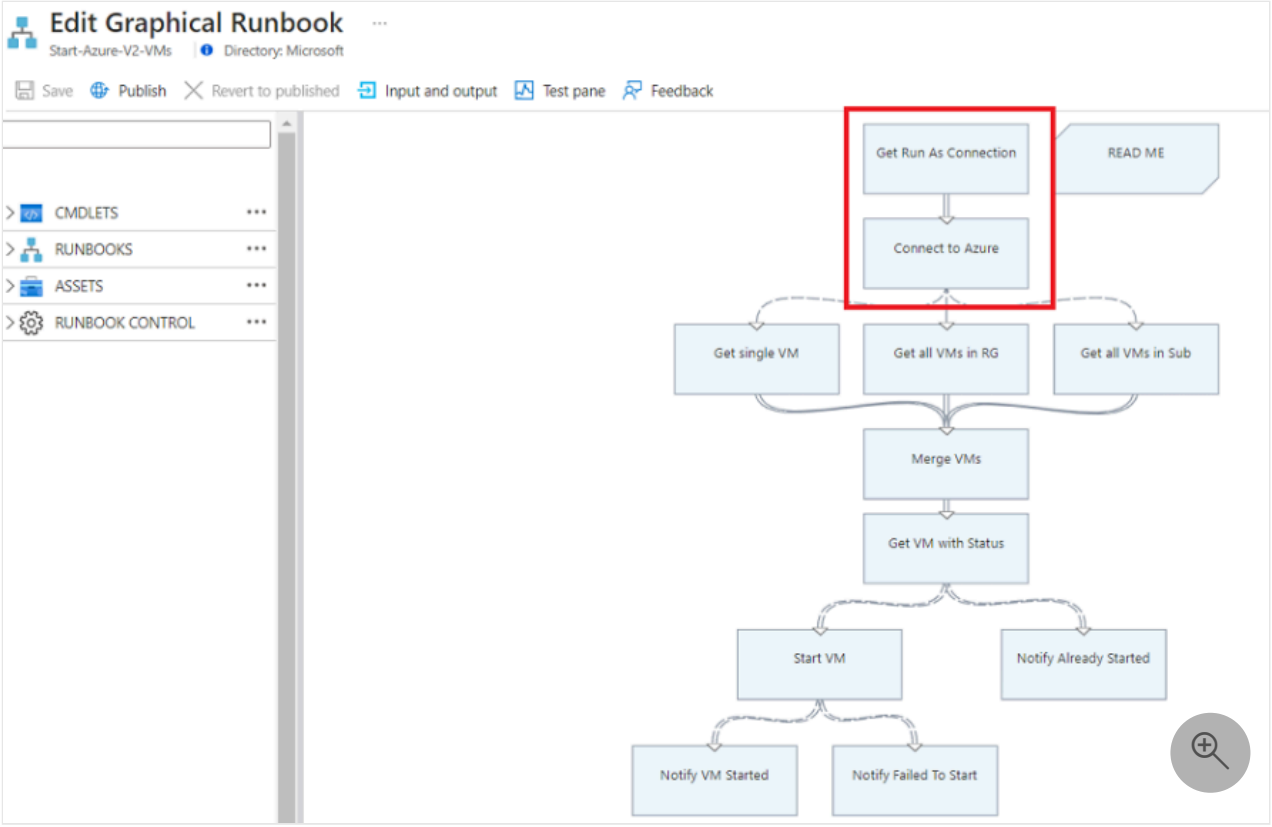


Edición de un runbook gráfico para usar una identidad administrada

Debe probar la identidad administrada para comprobar que el runbook gráfico funciona según lo previsto. Cree una copia del runbook de producción para usar la identidad administrada y, a continuación, actualice el código del runbook gráfico de prueba para autenticarse mediante la identidad administrada. Para agregar esta funcionalidad a un runbook gráfico, agregue el cmdlet `Connect-AzAccount`.

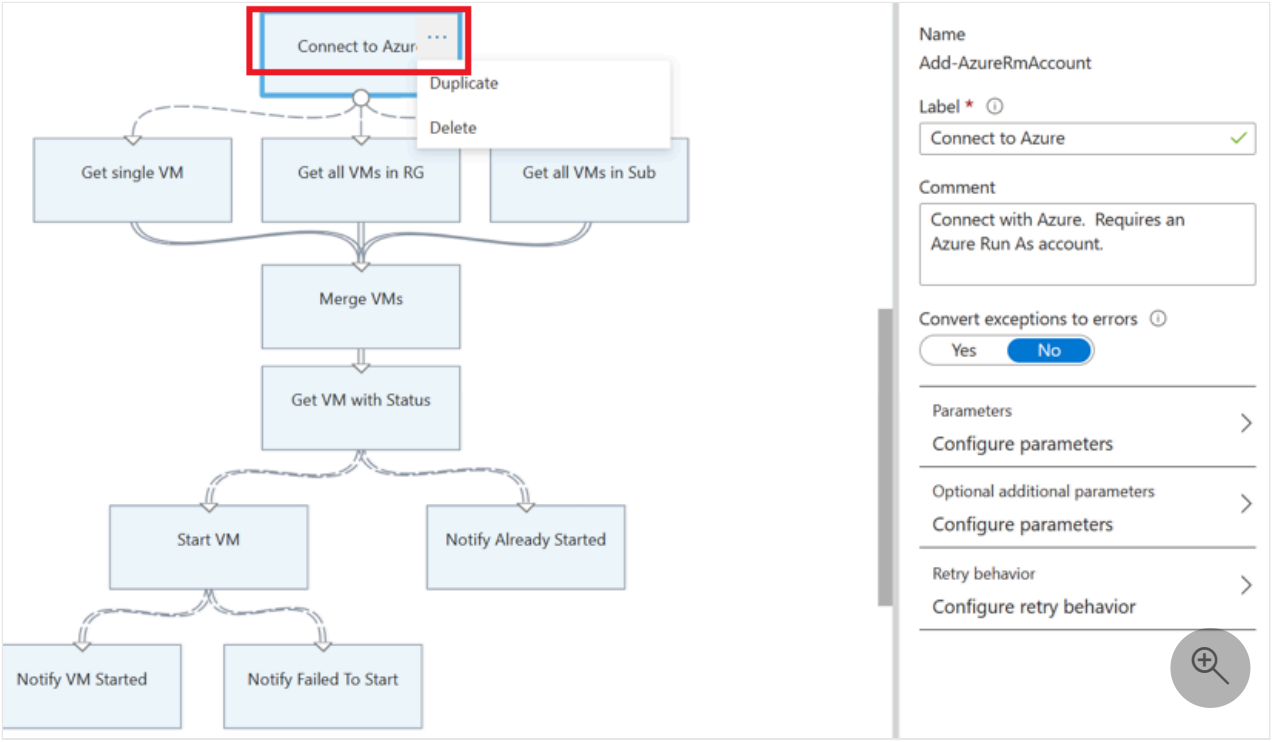
En los pasos siguientes se incluye un ejemplo para mostrar cómo un runbook gráfico que usa una cuenta de ejecución puede usar identidades administradas:

1. Inicie sesión en Azure Portal.
2. Abra la cuenta de Automation y seleccione **Automatización de procesos**>**Runbooks**.
3. Seleccionar un runbook. Por ejemplo, seleccione el runbook **Iniciar máquinas virtuales de Azure V2** de la lista y, a continuación, seleccione **Editar**, o bien vaya a **Examinar galería** y seleccione **Iniciar máquinas virtuales de Azure V2**.

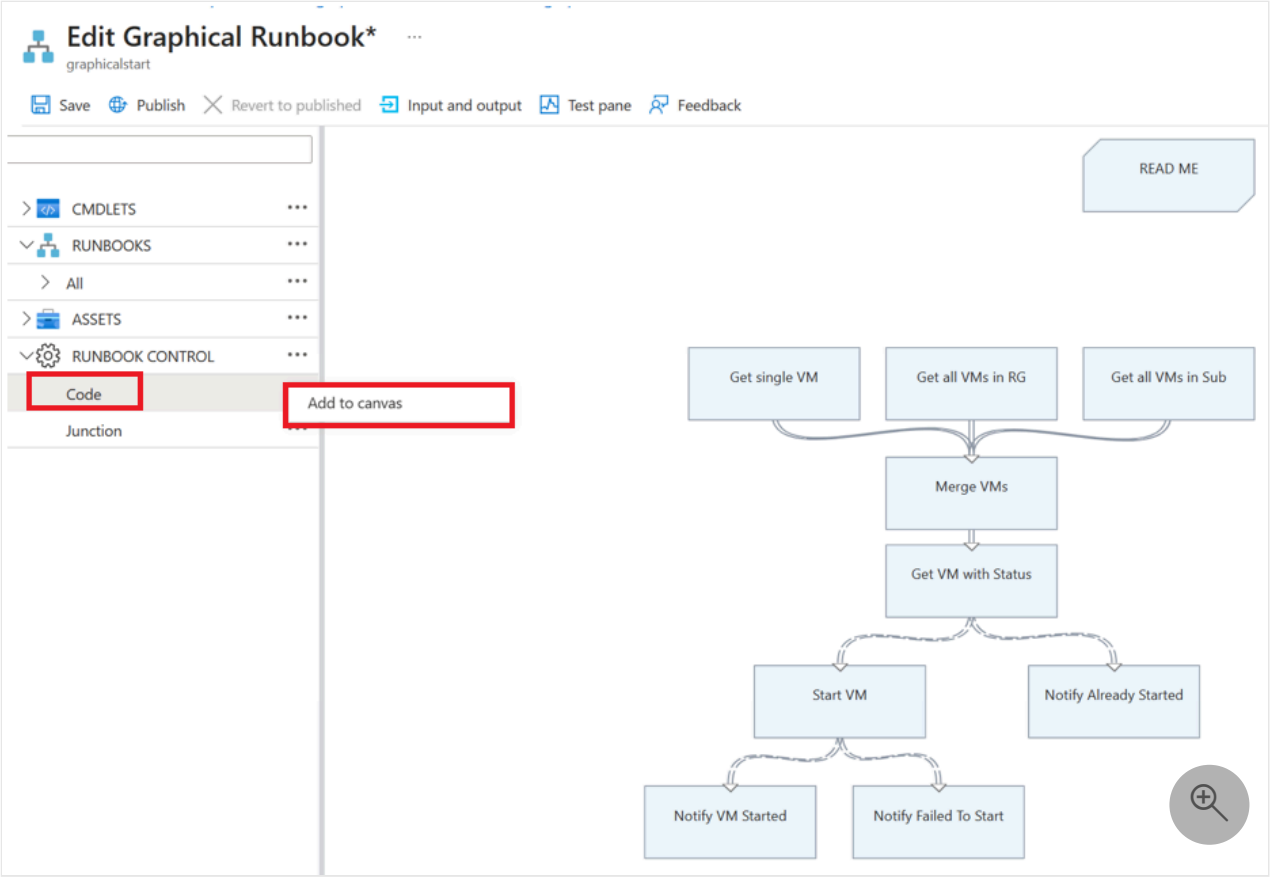


4. Reemplace la conexión de ejecución que usa `AzureRunAsConnection` y el recurso de conexión que usa internamente el cmdlet de PowerShell `Get-AutomationConnection` por el cmdlet `Connect-AzAccount`.

5. Seleccione **Eliminar** para eliminar la actividades `Get Run As Connection` y `Connect to Azure`.



6. En el panel izquierdo, en **CONTROL DE RUNBOOK**, seleccione **Código** y, a continuación, seleccione **Agregar al lienzo**.



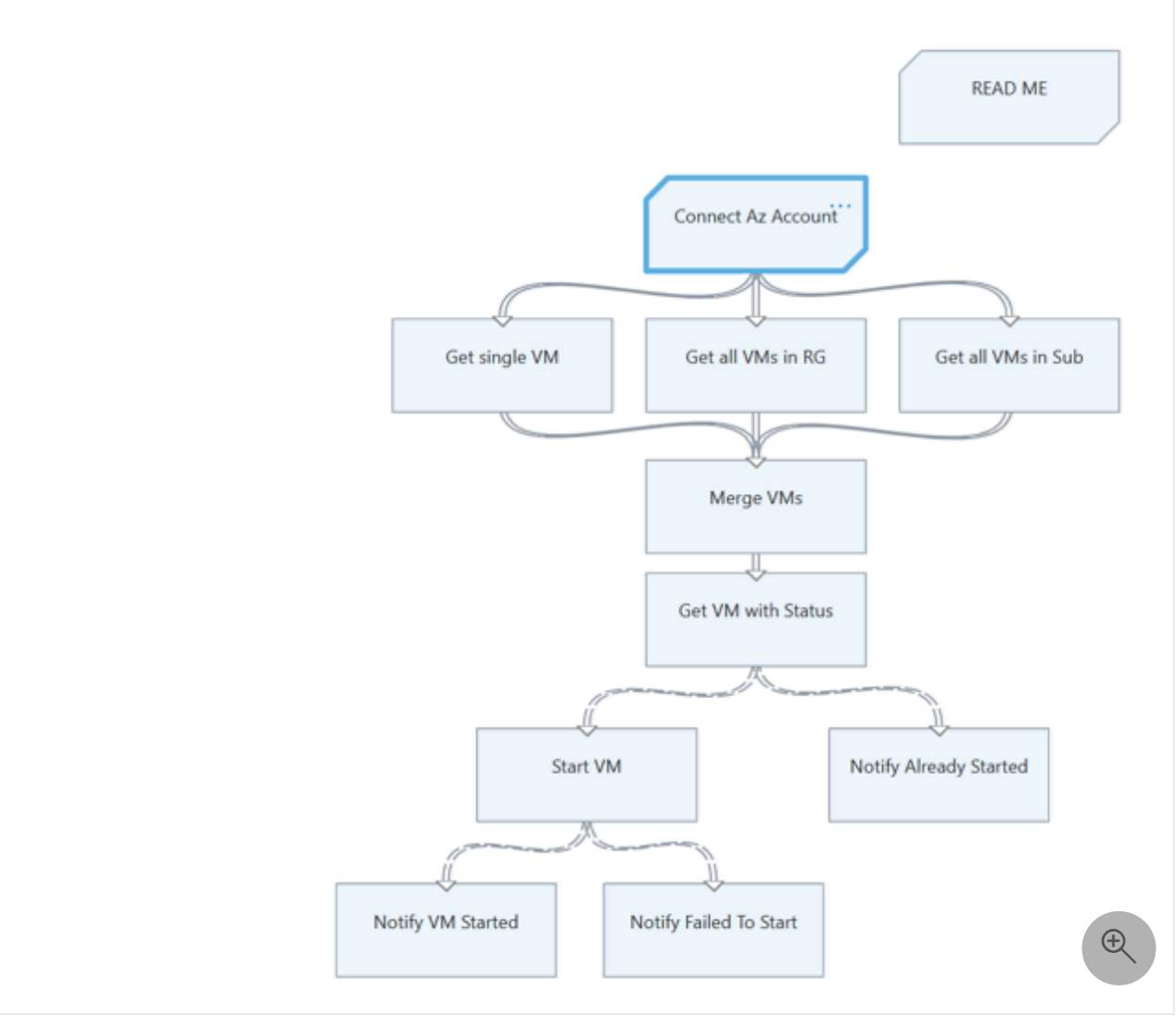
7. Edite la actividad de código, asigne cualquier nombre de etiqueta adecuado y seleccione **Lógica de actividad de creador**.

8. En la página **Editor de código**, escriba el siguiente código de PowerShell y seleccione **Aceptar**.

PowerShell

```
try
{
   Write-Output ("Logging in to Azure...")
   Connect-AzAccount -Identity
}
catch {
   Write-Error -Message $_.Exception
   throw $_.Exception
}
```

9. Conecte la nueva actividad a las actividades que se conectaron anteriormente mediante **Conectar a Azure** y guarde el runbook.



Por ejemplo, en el runbook **Iniciar máquinas virtuales de Azure V2** de la galería de runbooks, deberá reemplazar las actividades `Get Run As Connection` y `Connect to Azure` por la actividad de código que usa el cmdlet `Connect-AzAccount`, tal y como se describió anteriormente. Para más información, consulte el nombre del runbook de muestra **AzureAutomationTutorialWithIdentityGraphical** que se crea con la cuenta de Automation.

> ⓘ **Nota:**

Los módulos de AzureRM PowerShell se retirarán el **29 de febrero de 2024**. Si usa módulos de AzureRM PowerShell en runbooks gráficos, deberá actualizarlos para que usen módulos de Az PowerShell. **Más información**.

Pasos siguientes

- Revise las preguntas más frecuentes para migrar a identidades administradas
- Si los runbooks no se finalizan correctamente, revise Solución de problemas de identidad administrada de Azure Automation.
- Para obtener más información sobre las identidades administradas asignadas por el sistema, consulte Uso de una identidad administrada asignada por el sistema de una cuenta de Automation.
- Para obtener más información sobre las identidades administradas asignadas por el usuario, consulte Uso de una identidad administrada asignada por el usuario de una cuenta de Automation.
- Para obtener información sobre la seguridad de las cuentas de Azure Automation, vea Información general sobre la autenticación de cuentas de Azure Automation.

Last updated on 17/11/2025

Uso de la extensión de Azure Automation para Visual Studio Code

08/10/2025

En este artículo se explica cómo usar Visual Studio, el cual puede utilizarse para crear y administrar runbooks. Es posible realizar todas las operaciones de administración de runbook, como, crear y editar runbooks, desencadenar un trabajo, realizar un seguimiento de la salida de trabajos recientes, vincular una programación, la administración de recursos y la depuración local.

Prerrequisitos

Los siguientes elementos son necesarios para completar los pasos descritos en este artículo:

- Una suscripción de Azure. Si no tiene una suscripción de Azure, cree una cuenta gratuita.
- Código de Visual Studio.
- Los módulos de PowerShell y los paquetes de Python usados por runbook deben instalarse localmente en la máquina para ejecutar el runbook localmente.

Instalación y configuración de la extensión de Azure Automation

Después de cumplir los requisitos previos, puede instalar la extensión de Azure Automation para Visual Studio Code siguiendo estos pasos:

1. Abre Visual Studio Code.
2. En la barra de menús, vaya a **Ver**>**extensiones**.
3. En el cuadro de búsqueda, escriba **Azure Automation**.
4. Seleccione **Azure Automation** en los resultados de búsqueda y, a continuación, seleccione **Instalar**.
5. Seleccione **Volver a cargar** cuando sea necesario.

Conexión a una cuenta de Azure

Para ver todos los recursos de la cuenta de Automation, debe conectarse a su cuenta de Azure. Siga los pasos para conectarse a Azure desde Visual Studio Code:

1. Puede iniciar sesión en Azure desde la extensión de Azure Automation o la paleta de comandos.

    - Para iniciar sesión desde la extensión de Azure Automation: seleccione **Iniciar sesión en Azure**.

      O

    - Para iniciar sesión desde la paleta de comandos: en la barra de menús, vaya a **Ver** > **paleta de comandos** y escriba **Azure:Iniciar** sesión.

2. Siga las instrucciones de inicio de sesión para iniciar sesión en Azure. Después de conectarse, encontrará el nombre de la cuenta de Azure en la barra de estado de Visual Studio Code.

Selección de suscripciones

Al iniciar sesión por primera vez, la extensión carga solo los recursos de suscripción predeterminados y las cuentas de Automation. Para agregar o quitar suscripciones, siga estos pasos:

1. Puede usar la paleta de comandos o el pie de página de la ventana para iniciar el comando de suscripción.

    - Para iniciar sesión desde la paleta de comandos: en la barra de menús, vaya a **Ver** > **paleta de comandos** y escriba **Azure: Seleccione Suscripciones**.

      O

    - Para iniciar sesión desde el pie de página de la ventana: en el pie de página de la ventana, seleccione el segmento que coincida con **Azure: su cuenta**.

2. Use el filtro para buscar las suscripciones por nombre.

3. Active o desactive cada suscripción para agregarlas o quitarlas de la lista de suscripciones que muestra la extensión de Azure Automation.

4. Seleccione **Aceptar** después de completar la adición o eliminación de las suscripciones.

Uso de la extensión de Azure Automation

La extensión simplifica el proceso de creación y edición de runbooks. Ahora puede probarlos localmente sin iniciar sesión en Azure Portal. A continuación se enumeran las distintas acciones que puede realizar:

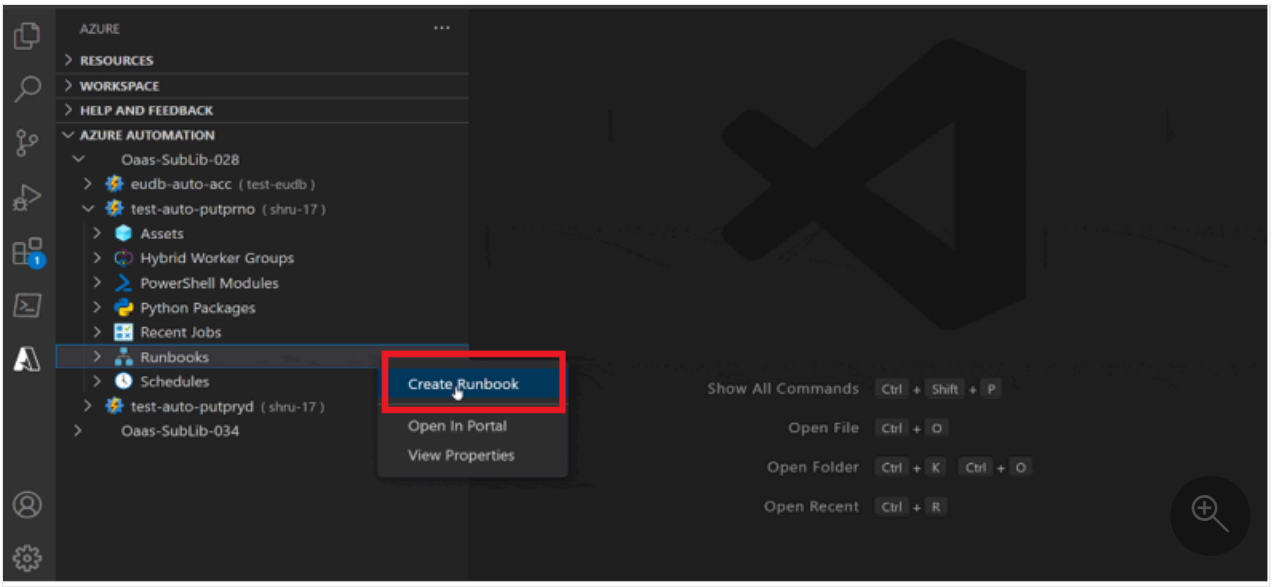
Crear un runbook

Crear un runbook en la cuenta de Automation. Siga estos pasos:

1. Inicie sesión en Azure desde la extensión de Azure Automation.

2. Seleccione **Runbooks**

3. Haga clic con el botón derecho y seleccione **Crear runbook** para crear un runbook en la cuenta de Automation.



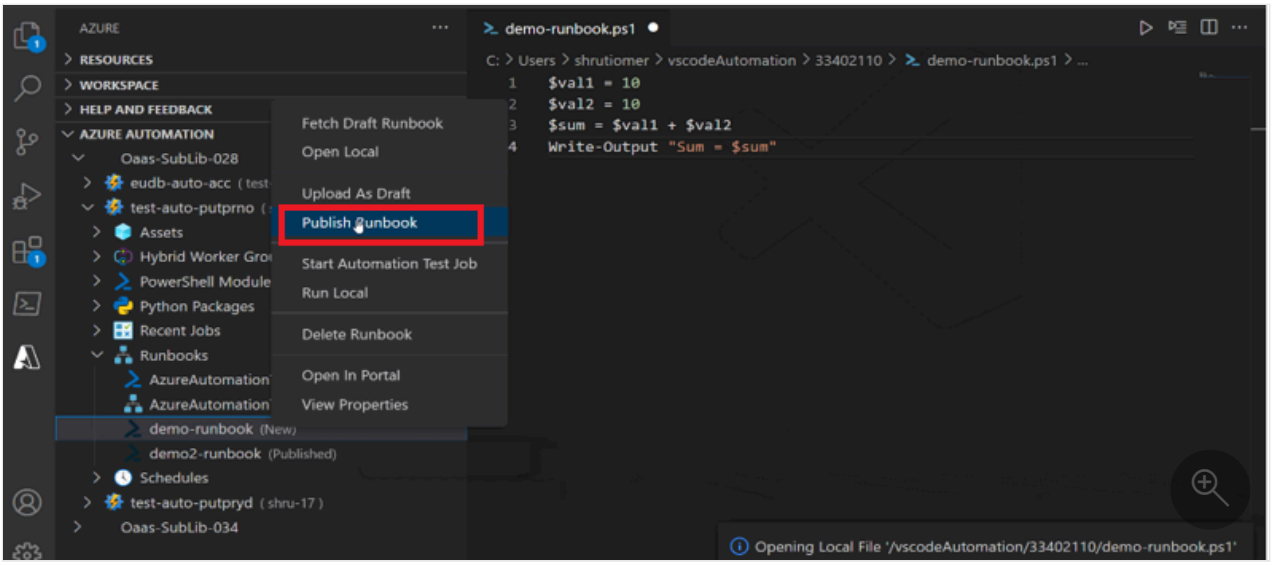
Publicación de un runbook

Para publicar un runbook en la cuenta de Automation. Siga estos pasos:

1. En la cuenta de Automation, seleccione el runbook.

2. Haga clic con el botón derecho y seleccione **Publicar runbook** para publicar el runbook.

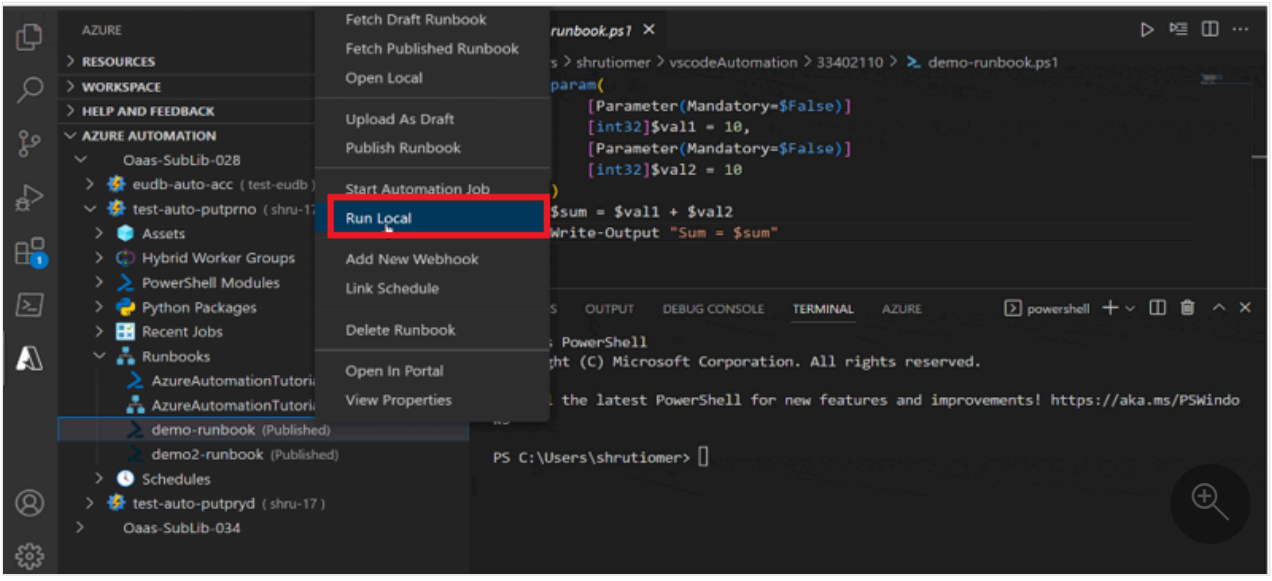
   Aparece una notificación de que el manual de ejecución se ha publicado correctamente.



Ejecución de la versión local del trabajo de Automation

Para ejecutar la versión local del trabajo de Automation, siga estos pasos:

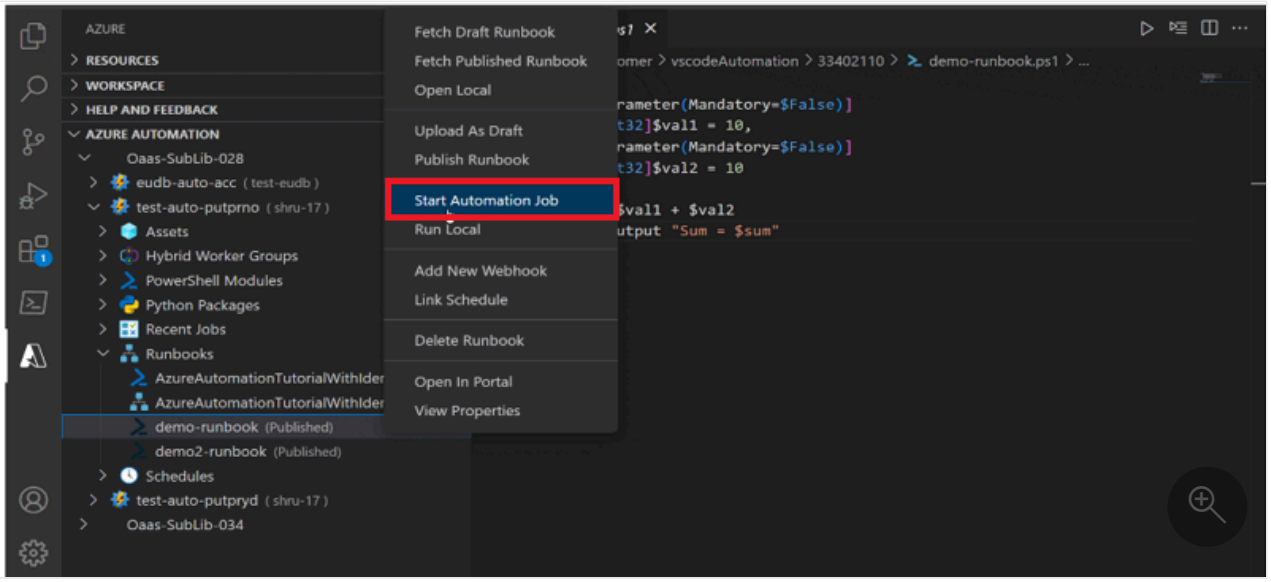
1. En la cuenta de Automation, seleccione el runbook.
2. Haga clic con el botón derecho y seleccione **Ejecutar local** para ejecutar la versión local del trabajo de Automation.



Ejecutar trabajo de automatización

Para ejecutar el trabajo de Automation, siga estos pasos:

1. En la cuenta de Automation, seleccione el runbook.
2. Haga clic con el botón derecho y seleccione **Iniciar trabajo de Automation** para ejecutar el trabajo de Automation.

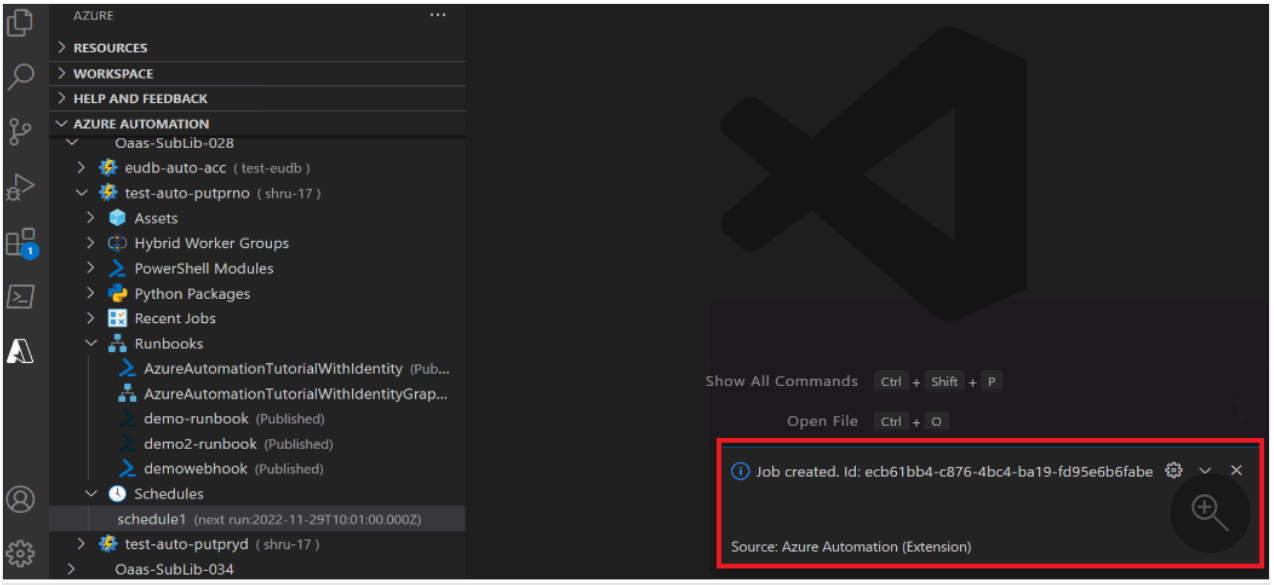


Adición de un nuevo webhook

Para agregar un webhook al runbook, siga estos pasos:

1. En la cuenta de Automation, seleccione el runbook.
2. Haga clic con el botón derecho y seleccione **Agregar nuevo webhook**.
3. Seleccione y copie el URI del webhook.
4. Utilice la paleta de comandos y seleccione **Azure Automation Trigger Webhook**.
5. Pegue el URI del webhook.

   Aparece una notificación que indica que JobId se ha creado correctamente.



Enlazar un horario

1. En la cuenta de Automation, vaya a **Programaciones** y seleccione la programación.

2. Vaya a **Runbooks** y seleccione el runbook.

3. Haga clic con el botón derecho y seleccione **Vincular calendario** y confirme el calendario.

4. En la lista desplegable, seleccione **Azure**.

   Aparece una notificación indicando que el horario está vinculado.

Administrar recursos

1. En la cuenta de Automation, vaya a **Recursos**>**Variables fx**.

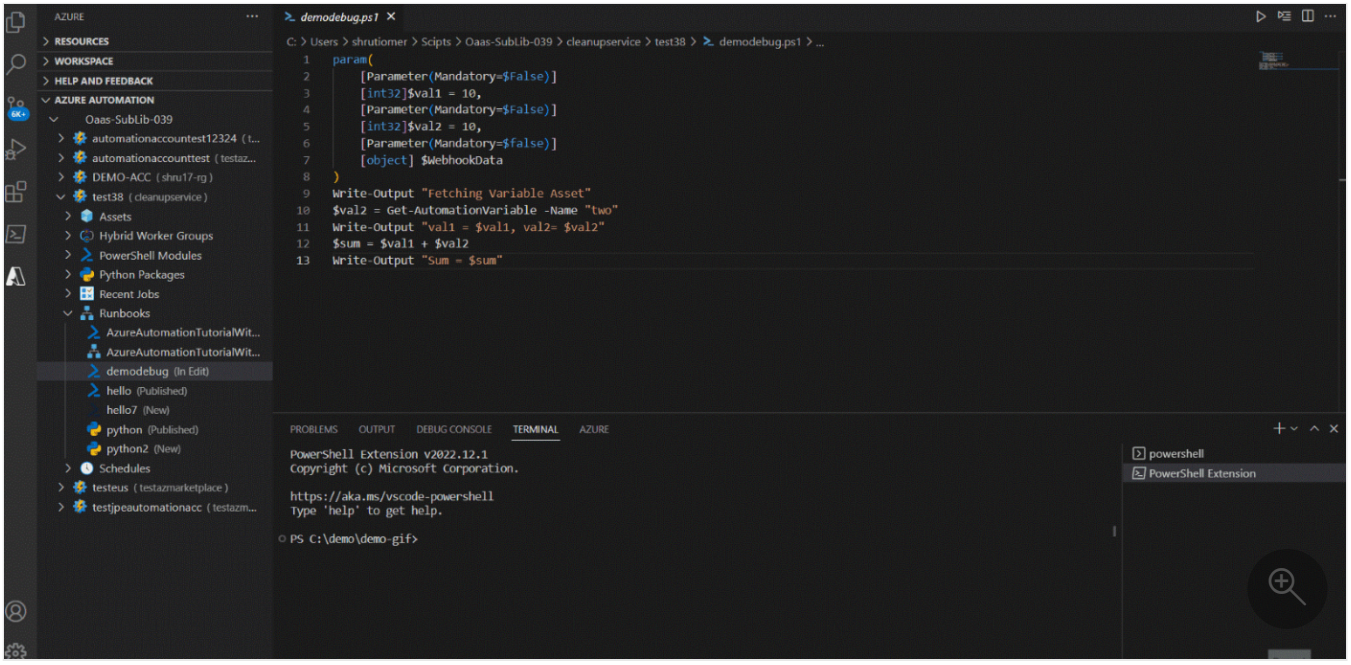
2. Haga clic con el botón derecho y seleccione **Crear o actualizar**.

3. Proporcione un nombre en el cuadro de texto.

   Aparece una notificación que indica que la variable se crea, puede ver la nueva variable en la opción **variables fx** .

Ejecución local en modo de depuración

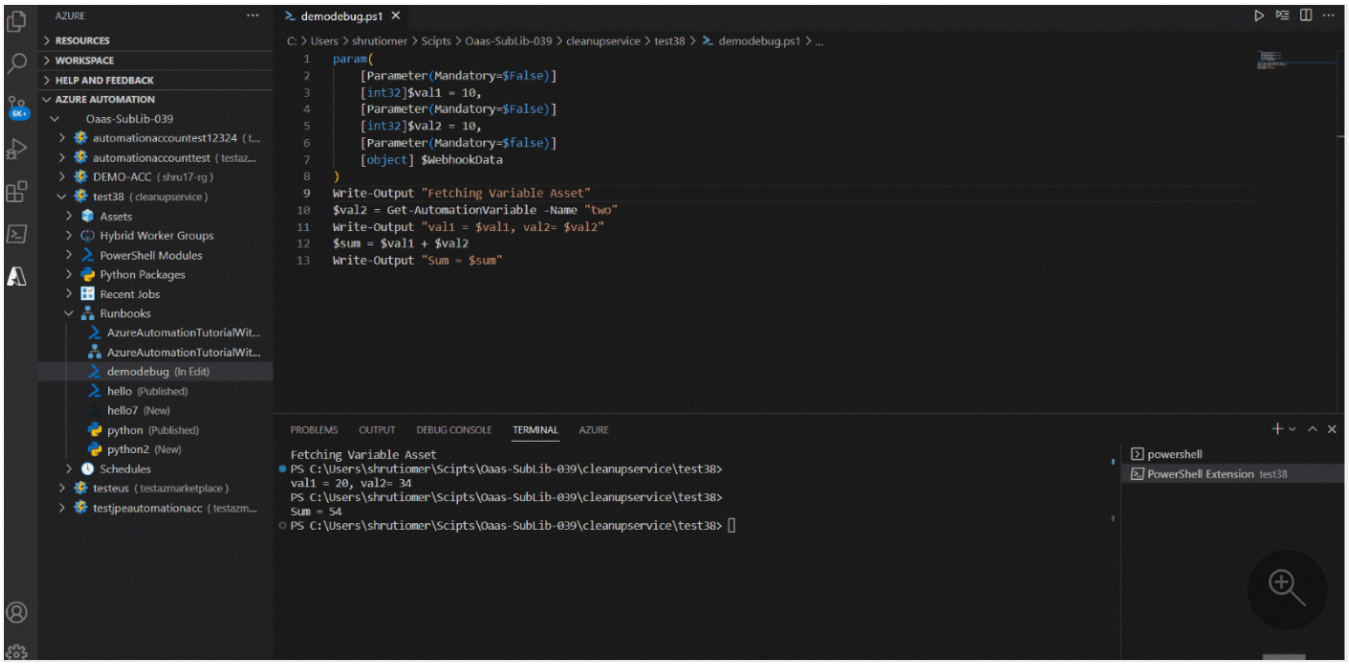
1. En la cuenta de Automation, vaya a **Runbooks** y seleccione un runbook.
2. En el panel de edición, agregue el punto de interrupción.
3. Haga clic con el botón derecho en el runbook y seleccione **Ejecutar localmente en modo de depuración**.



Comparación del runbook local

1. En la cuenta de Automatización, vaya a **Runbooks** y seleccione un runbook.
2. Haga clic con el botón derecho en el runbook y seleccione **Comparar runbook local**.
3. En el panel de edición, verá la información en dos diseños: copia de runbook y copia publicada o borrador.

> **Nota**
>
> Si el runbook estuviera en modo **InEdit**, tendrá que seleccionar una de las dos opciones para comparar: Comparar contenido publicado o Comparar contenido de borrador.



Pasos siguientes

- Para obtener información sobre las características clave y las limitaciones de la extensión de Azure Automation, consulte Creación de runbook mediante VS Code en Azure Automation.

Autenticación de runbooks con Amazon Web Services

Puede automatizar tareas comunes con recursos en Amazon Web Services (AWS) mediante runbooks de Automation en Azure. Puede automatizar muchas tareas en AWS mediante runbooks de Automation similares a los recursos de Azure. Asegúrese de que tiene la suscripción de Azure para autenticarse.

Obtención de credenciales y suscripciones de AWS

Asegúrese de obtener una suscripción de AWS y especificar un conjunto de credenciales de AWS para autenticar los runbooks que se ejecutan desde Azure Automation. Las credenciales específicas necesarias son la clave de acceso de AWS y la clave secreta. Consulte Uso de credenciales de AWS.

Configuración de la cuenta de Automation

Puede usar una cuenta de Automation existente para autenticarse con AWS. Como alternativa, puede dedicar una cuenta para runbooks que tengan como destino recursos de AWS. En este caso, cree una nueva cuenta de Automation.

Almacenamiento de credenciales de AWS

Debe almacenar las credenciales de AWS como recursos en Azure Automation. Consulte Administración de claves de acceso para su cuenta de AWS para obtener instrucciones sobre cómo crear la clave de acceso y la clave secreta. Cuando las claves estén disponibles, copie el identificador de clave de acceso y el identificador de clave secreta en un lugar seguro. Puede descargar el archivo de clave para almacenarlo de forma segura.

Crear recurso de credencial

Después de crear y copiar las claves de seguridad de AWS, debe crear un recurso Credencial con la cuenta de Automation. El recurso le permite almacenar de forma segura las claves de AWS y hacer referencia a ellas en los runbooks. Consulte Creación de un nuevo recurso de credenciales con Azure Portal.

Escriba la siguiente información de AWS en los campos proporcionados:

- **Nombre** - **AWScred** o un valor adecuado siguiendo los estándares de nomenclatura

- **Nombre de usuario:** identificador de acceso
- **Contraseña:** nombre de la clave secreta

Pasos siguientes

- Para obtener información sobre cómo crear runbooks para automatizar tareas en AWS, consulte Implementación de una máquina virtual de Amazon Web Services con un runbook.

Last updated on 30/03/2026

Uso de Microsoft Entra ID para autenticarse en Azure

El servicio Microsoft Entra ID habilita varias tareas administrativas, como la administración de usuarios, la administración de dominios y la configuración de inicio de sesión único. En este artículo se describe cómo usar Microsoft Entra ID dentro de Azure Automation como proveedor de autenticación para Azure.

Instalación de módulos de Microsoft Entra

Puede habilitar Microsoft Entra ID a través de los siguientes módulos de PowerShell:

- Azure Active Directory PowerShell para Graph. Azure Automation se incluye con el módulo Az. La funcionalidad incluye autenticación no interactiva para Azure mediante la autenticación basada en credenciales de usuario (OrgId) de Microsoft Entra. Consulte Azure AD 2.0.2.182 .
- Microsoft Entra ID para Windows PowerShell. Este módulo habilita las interacciones con Microsoft Online, incluida la Microsoft 365.

Instalación de la compatibilidad con PSCredential

Azure Automation usa la clase PSCredential para representar un recurso de credenciales. Los scripts recuperan los objetos `PSCredential` mediante el cmdlet `Get-AutomationPSCredential`. Para obtener más información, vea Credential assets in Azure Automation.

Asignación de un administrador de suscripción

Debe asignar un administrador para la suscripción de Azure. Esta persona tiene el rol de propietario para el ámbito de la suscripción. Consulte el control de acceso basado en roles en Azure Automation.

Cambiar la contraseña del usuario del Microsoft Entra

Para cambiar la contraseña del usuario Microsoft Entra:

1. Cierre sesión de Azure.

2. Haga que el administrador inicie sesión en Azure como acaba de crear el usuario Microsoft Entra, con el nombre de usuario completo (incluido el dominio) y una contraseña temporal.

3. Pida al administrador que cambie la contraseña cuando se le solicite.

Configuración de Azure Automation para administrar la suscripción de Azure

Para que Azure Automation comunicarse con Microsoft Entra ID, debe recuperar las credenciales asociadas a la conexión de Azure a Microsoft Entra ID. Algunos ejemplos de estas credenciales son el id. de inquilino, el id. de suscripción y otras similares. Para obtener más información sobre la conexión entre Azure y Microsoft Entra ID, consulte Conectar la organización a Microsoft Entra ID.

Creación de un recurso de credencial

Con las credenciales de Azure para Microsoft Entra disponibles, es el momento de crear un recurso de credenciales de Azure Automation para almacenar de forma segura las credenciales de Microsoft Entra para que los runbooks y los scripts de Desired State Configuration (DSC) puedan acceder a ellos. Puede hacerlo mediante el portal de Azure o los cmdlets de PowerShell.

Creación del recurso de credenciales en Azure portal

Puede usar el portal de Azure para crear el recurso de credenciales. Realice esta operación desde la cuenta de Automation mediante **credenciales** en **Recursos compartidos**. Consulte Recursos de credenciales en Azure Automation.

Creación del recurso de credenciales con Windows PowerShell

Para preparar un nuevo recurso de credenciales en Windows PowerShell, el script crea primero un objeto `PSCredential` mediante el nombre de usuario y la contraseña asignados. A continuación, el script usa este objeto para crear el recurso mediante una llamada al cmdlet New-AzAutomationCredential . Como alternativa, el script puede llamar al cmdlet Get-Credential para pedir al usuario que escriba un nombre y una contraseña. Consulte Recursos de credenciales en Azure Automation.

Gestión de recursos de Azure desde un runbook de Azure Automation

Puede administrar recursos de Azure desde libros de ejecución de Azure Automation mediante el activo de credenciales. A continuación se muestra un runbook de PowerShell de ejemplo que recopila el recurso de credenciales que se va a usar para detener e iniciar máquinas virtuales en una suscripción de Azure. En primer lugar, este runbook usa `Get-AutomationPSCredential` para recuperar la credencial que se va a usar para autenticarse en Azure. A continuación, llama al cmdlet Connect-AzAccount para conectarse a Azure mediante la credencial.

PowerShell

```
Workflow Workflowname
{
    Param
    (
        [Parameter(Mandatory=$true)][ValidateNotNullOrEmpty()]
        [String]
        $AzureSubscriptionId,
        [Parameter(Mandatory=$true)][ValidateNotNullOrEmpty()]
        [String]
        $AzureVMList="All",
        [Parameter(Mandatory=$true)][ValidateSet("Start","Stop")]
        [String]
        $Action
    )

    # Ensures you do not inherit an AzContext in your runbook
    Disable-AzContextAutosave -Scope Process

    # Connect to Azure with system-assigned managed identity
    $AzureContext = (Connect-AzAccount -Identity).context

    # set and store context
    $AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext

    # get credential
    $credential = Get-AutomationPSCredential -Name "AzureCredential"

    # Connect to Azure with credential
    $AzureContext = (Connect-AzAccount -Credential $credential -TenantId
$AzureContext.Subscription.TenantId).context

    # set and store context
    $AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription `
        -TenantId $AzureContext.Subscription.TenantId `
        -DefaultProfile $AzureContext

    if($AzureVMList -ne "All")
```

```
    {
        $AzureVMs = $AzureVMList.Split(",")
        [System.Collections.ArrayList]$AzureVMsToHandle = $AzureVMs
    }
    else
    {
        $AzureVMs = (Get-AzVM -DefaultProfile $AzureContext).Name
        [System.Collections.ArrayList]$AzureVMsToHandle = $AzureVMs
    }

    foreach($AzureVM in $AzureVMsToHandle)
    {
        if(!(Get-AzVM -DefaultProfile $AzureContext | ? {$_.Name -eq $AzureVM}))
        {
            throw " AzureVM : [$AzureVM] - Does not exist! - Check your inputs "
        }
    }

    if($Action -eq "Stop")
    {
        Write-Output "Stopping VMs";
        foreach -parallel ($AzureVM in $AzureVMsToHandle)
        {
            Get-AzVM -DefaultProfile $AzureContext | ? {$_.Name -eq $AzureVM} |
Stop-AzVM -DefaultProfile $AzureContext -Force
        }
    }
    else
    {
        Write-Output "Starting VMs";
        foreach -parallel ($AzureVM in $AzureVMsToHandle)
        {
            Get-AzVM -DefaultProfile $AzureContext | ? {$_.Name -eq $AzureVM} |
Start-AzVM -DefaultProfile $AzureContext
        }
    }
}
```

Uso de Microsoft Graph con PowerShell

Consulte Comenzar con el SDK de Microsoft Graph PowerShell

Pasos siguientes

- Para obtener más información sobre el uso de credenciales, consulte Administrar credenciales en Azure Automation.
- Para obtener información sobre los módulos, consulte Administrar módulos en Azure Automation.
- Si necesita iniciar un runbook, consulte Iniciar un runbook en Azure Automation.

- Para más información sobre PowerShell, consulte La documentación de PowerShell.

Last updated on 08/04/2026

Registros DNS para regiones de Azure utilizados por Azure Automation

El servicio Azure Automation utiliza muchos registros DNS (Sistema de nombres de dominio) para que las características se conecten al servicio. Si tiene una cuenta de Automation configurada para una región específica, puede restringir la comunicación con ese centro de datos regional. Es posible que necesite conocer estos registros para permitir que las siguientes características de Automation funcionen cuando se hospedan detrás de un firewall:

- Hybrid Runbook Worker
- State Configuration
- webhooks

> **Nota:**
>
> El registro de Hybrid Runbook Worker de Linux producirá un error con los nuevos registros a menos que sea de la versión 1.6.10.2 o posterior. Debe actualizar a una versión más reciente del agente de Log Analytics para Linux para que la máquina reciba una versión actualizada del rol de trabajo y use estos nuevos registros. Los equipos existentes seguirán funcionando sin problemas.

Registros DNS por región

En la tabla siguiente se proporciona el registro de DNS para cada región.

> **Nota:**
>
> Aunque la lista de los registros DNS de Automation que se proporcionan aquí se ha retirado, siguen siendo funcionales para darle tiempo de migrar a los nuevos registros que aparecen en Compatibilidad con el vínculo privado y evitar errores con los procesos de automatización.

Expandir tabla

Región	Registro de DNS	Código de almacén
Centro de Australia	ac-jobruntimedata-prod-su1.azure-automation.net ac-agentservice-prod-1.azure-automation.net	ac

Región	Registro de DNS	Código de almacén
Este de Australia	ae-jobruntimedata-prod-su1.azure-automation.net ae-agentservice-prod-1.azure-automation.net	ae
Sudeste de Australia	ase-jobruntimedata-prod-su1.azure-automation.net ase-agentservice-prod-1.azure-automation.net	ase
Centro de Canadá	cc-jobruntimedata-prod-su1.azure-automation.net cc-agentservice-prod-1.azure-automation.net	cc
Centro de la India	cid-jobruntimedata-prod-su1.azure-automation.net cid-agentservice-prod-1.azure-automation.net	cid
Este de EE. UU. 2	eus2-jobruntimedata-prod-su1.azure-automation.net eus2-agentservice-prod-1.azure-automation.net	eus2
Japón Oriental	jpe-jobruntimedata-prod-su1.azure-automation.net jpe-agentservice-prod-1.azure-automation.net	jpe
Norte de Europa	ne-jobruntimedata-prod-su1.azure-automation.net ne-agentservice-prod-1.azure-automation.net	ne
Centro-sur de EE. UU.	scus-jobruntimedata-prod-su1.azure-automation.net scus-agentservice-prod-1.azure-automation.net	scus
Sudeste de Asia	sea-jobruntimedata-prod-su1.azure-automation.net sea-agentservice-prod-1.azure-automation.net	sea
Norte de Suiza	stzn-jobruntimedata-prod-su1.azure-automation.net stzn-agentservice-prod-su1.azure-automation.net	stzn
Sur de Reino Unido 2	uks-jobruntimedata-prod-su1.azure-automation.net uks-agentservice-prod-1.azure-automation.net	uks
US Gov - Virginia	usge-jobruntimedata-prod-su1.azure-automation.us usge-agentservice-prod-1.azure-automation.us	usge
Centro-Oeste de EE. UU.	wcus-jobruntimedata-prod-su1.azure-automation.net wcus-agentservice-prod-1.azure-automation.net	wcus
Oeste de Europa	we-jobruntimedata-prod-su1.azure-automation.net we-agentservice-prod-1.azure-automation.net	we
Oeste de EE. UU. 2	wus2-jobruntimedata-prod-su1.azure-automation.net wus2-agentservice-prod-1.azure-automation.net	wus2

Compatibilidad con el vínculo privado

Para admitir el vínculo privado en Azure Automation, se han actualizado los registros DNS de cada centro de datos compatible. En lugar de direcciones URL específicas de la región, las direcciones URL son específicas de la cuenta de Automation.

Expandir tabla

Región	Registro de DNS	Código de almacén

Norte de Sudáfrica	`https://<accountId>.webhook.san.azure-automation.net`
	`https://<accountId>.agentsvc.san.azure-automation.net`
	`https://<accountId>.jrds.san.azure-automation.net`
		san

Este de Asia	`https://<accountId>.webhook.ea.azure-automation.net`
	`https://<accountId>.agentsvc.ea.azure-automation.net`
	`https://<accountId>.jrds.ea.azure-automation.net`
		c/u

Sudeste de Asia	`https://<accountId>.webhook.sea.azure-automation.net`
	`https://<accountId>.agentsvc.sea.azure-automation.net`
	`https://<accountId>.jrds.sea.azure-automation.net`
		sea

Centro de Australia	`https://<accountId>.webhook.ac.azure-automation.net`
	`https://<accountId>.agentsvc.ac.azure-automation.net`
	`https://<accountId>.jrds.ac.azure-automation.net`
		ac

Centro de Australia 2	`https://<accountId>.webhook.cbr2.azure-automation.net`
	`https://<accountId>.agentsvc.cbr2.azure-automation.net`
	`https://<accountId>.jrds.cbr2.azure-automation.net`
		cbr2

Sudeste de Australia	`https://<accountId>.webhook.ase.azure-automation.net`
	`https://<accountId>.agentsvc.ase.azure-automation.net`
	`https://<accountId>.jrds.ase.azure-automation.net`
		ase

Este de Australia	`https://<accountId>.webhook.ae.azure-automation.net` `https://<accountId>.agentsvc.ae.azure-`	ae

Región	Registro de DNS	Código de almacén
	automation.net https://<accountId>.jrds.ae.azure-automation.net	
Sur de Brasil	https://<accountId>.webhook.brs.azure-automation.net https://<accountId>.agentsvc.brs.azure-automation.net https://<accountId>.jrds.brs.azure-automation.net	brs
Sudeste de Brasil	https://<accountId>.webhook.brse.azure-automation.net https://<accountId>.agentsvc.brse.azure-automation.net https://<accountId>.jrds.brse.azure-automation.net	brse
Centro de Canadá	https://<accountId>.webhook.cc.azure-automation.net https://<accountId>.agentsvc.cc.azure-automation.net https://<accountId>.jrds.cc.azure-automation.net	cc
Este de China 2	https://<accountId>.webhook.sha2.azure-automation.cn https://<accountId>.agentsvc.sha2.azure-automation.cn https://<accountId>.jrds.sha2.azure-automation.cn	sha2
Norte de China	https://<accountId>.webhook.bjb.azure-automation.cn https://<accountId>.agentsvc.bjb.azure-automation.cn https://<accountId>.jrds.bjb.azure-automation.cn	bjb
Norte de China 2	https://<accountId>.webhook.bjs2.azure-automation.cn https://<accountId>.agentsvc.bjs2.azure-automation.cn https://<accountId>.jrds.bjs2.azure-automation.cn	bjs2
Norte de China 3	https://<accountId>.webhook.cnn3.azure-automation.cn https://<accountId>.agentsvc.cnn3.azure-automation.cn https://<accountId>.jrds.cnn3.azure-automation.cn	cnn3
Oeste de Europa	https://<accountId>.webhook.we.azure-automation.net	we

Región	Registro de DNS	Código de almacén
	`https://<accountId>.agentsvc.we.azure-automation.net` `https://<accountId>.jrds.we.azure-automation.net`	

Norte de Europa	`https://<accountId>.webhook.ne.azure-automation.net`
	`https://<accountId>.agentsvc.ne.azure-automation.net`
	`https://<accountId>.jrds.ne.azure-automation.net`
		ne

Centro de Francia	`https://<accountId>.webhook.fc.azure-automation.net`
	`https://<accountId>.agentsvc.fc.azure-automation.net`
	`https://<accountId>.jrds.fc.azure-automation.net`
		fc

Sur de Francia	`https://<accountId>.webhook.mrs.azure-automation.net`
	`https://<accountId>.agentsvc.mrs.azure-automation.net`
	`https://<accountId>.jrds.mrs.azure-automation.net`
		mrs

Centro-oeste de Alemania	`https://<accountId>.webhook.dewc.azure-automation.de`
	`https://<accountId>.agentsvc.dewc.azure-automation.de`
	`https://<accountId>.jrds.dewc.azure-automation.de`
		dewc

Centro de la India	`https://<accountId>.webhook.cid.azure-automation.net`
	`https://<accountId>.agentsvc.cid.azure-automation.net`
	`https://<accountId>.jrds.cid.azure-automation.net`
		cid

Sur de la India	`https://<accountId>.webhook.ma.azure-automation.net`
	`https://<accountId>.agentsvc.ma.azure-automation.net`
	`https://<accountId>.jrds.ma.azure-automation.net`
		ma

Japón Oriental	`https://<accountId>.webhook.jpe.azure-automation.net`
	`https://<accountId>.agentsvc.jpe.azure-automation.net`
	`https://<accountId>.jrds.jpe.azure-automation.net`
		jpe

Región	Registro de DNS	Código de almacén

Japón Occidental	`https://<accountId>.webhook.jpw.azure-automation.net`
	`https://<accountId>.agentsvc.jpw.azure-automation.net`
	`https://<accountId>.jrds.jpw.azure-automation.net`
		jpw

Centro de Corea del Sur	`https://<accountId>.webhook.kc.azure-automation.net`
	`https://<accountId>.agentsvc.kc.azure-automation.net`
	`https://<accountId>.jrds.kc.azure-automation.net`
		kc

Corea del Sur	`https://<accountId>.webhook.ps.azure-automation.net`
	`https://<accountId>.agentsvc.ps.azure-automation.net`
	`https://<accountId>.jrds.ps.azure-automation.net`
		ps

Este de Noruega	`https://<accountId>.webhook.noe.azure-automation.net`
	`https://<accountId>.agentsvc.noe.azure-automation.net`
	`https://<accountId>.jrds.noe.azure-automation.net`
		noe

Oeste de Noruega	`https://<accountId>.webhook.now.azure-automation.net`
	`https://<accountId>.agentsvc.now.azure-automation.net`
	`https://<accountId>.jrds.now.azure-automation.net`
		ahora

Oeste de Suiza	`https://<accountId>.webhook.stzw.azure-automation.net`
	`https://<accountId>.agentsvc.stzw.azure-automation.net`
	`https://<accountId>.jrds.stzw.azure-automation.net`
		stzw

Centro de Emiratos Árabes Unidos	`https://<accountId>.webhook.auh.azure-automation.net`
	`https://<accountId>.agentsvc.auh.azure-automation.net`
	`https://<accountId>.jrds.auh.azure-automation.net`
		auh

Norte de Emiratos Árabes Unidos	`https://<accountId>.webhook.uaen.azure-automation.net` `https://<accountId>.agentsvc.uaen.azure-`	uaen

Región	Registro de DNS	Código de almacén
	`automation.net` `https://<accountId>.jrds.uaen.azure-automation.net`	

Oeste de Reino Unido	`https://<accountId>.webhook.cw.azure-automation.net`
	`https://<accountId>.agentsvc.cw.azure-automation.net`
	`https://<accountId>.jrds.cw.azure-automation.net`
		cw

Sur de Reino Unido 2	`https://<accountId>.webhook.uks.azure-automation.net`
	`https://<accountId>.agentsvc.uks.azure-automation.net`
	`https://<accountId>.jrds.uks.azure-automation.net`
		uks

Centro de EE. UU.	`https://<accountId>.webhook.cus.azure-automation.net`
	`https://<accountId>.agentsvc.cus.azure-automation.net`
	`https://<accountId>.jrds.cus.azure-automation.net`
		cus

Este de EE. UU.	`https://<accountId>.webhook.eus.azure-automation.net`
	`https://<accountId>.agentsvc.eus.azure-automation.net`
	`https://<accountId>.jrds.eus.azure-automation.net`
		eus

Este de EE. UU. 2	`https://<accountId>.webhook.eus2.azure-automation.net`
	`https://<accountId>.agentsvc.eus2.azure-automation.net`
	`https://<accountId>.jrds.eus2.azure-automation.net`
		eus2

Centro-Norte de EE. UU	`https://<accountId>.webhook.ncus.azure-automation.net`
	`https://<accountId>.agentsvc.ncus.azure-automation.net`
	`https://<accountId>.jrds.ncus.azure-automation.net`
		ncus

Centro-sur de EE. UU.	`https://<accountId>.webhook.scus.azure-automation.net`
	`https://<accountId>.agentsvc.scus.azure-automation.net`
	`https://<accountId>.jrds.scus.azure-automation.net`
		scus

Centro-Oeste de EE. UU.	`https://<accountId>.webhook.wcus.azure-automation.net`	wcus

Región	Registro de DNS	Código de almacén
	`https://<accountId>.agentsvc.wcus.azure-automation.net` `https://<accountId>.jrds.wcus.azure-automation.net`	

Oeste de EE. UU.	`https://<accountId>.webhook.wus.azure-automation.net`
	`https://<accountId>.agentsvc.wus.azure-automation.net`
	`https://<accountId>.jrds.wus.azure-automation.net`
		wus

Oeste de EE. UU. 2	`https://<accountId>.webhook.wus2.azure-automation.net`
	`https://<accountId>.agentsvc.wus2.azure-automation.net`
	`https://<accountId>.jrds.wus2.azure-automation.net`
		wus2

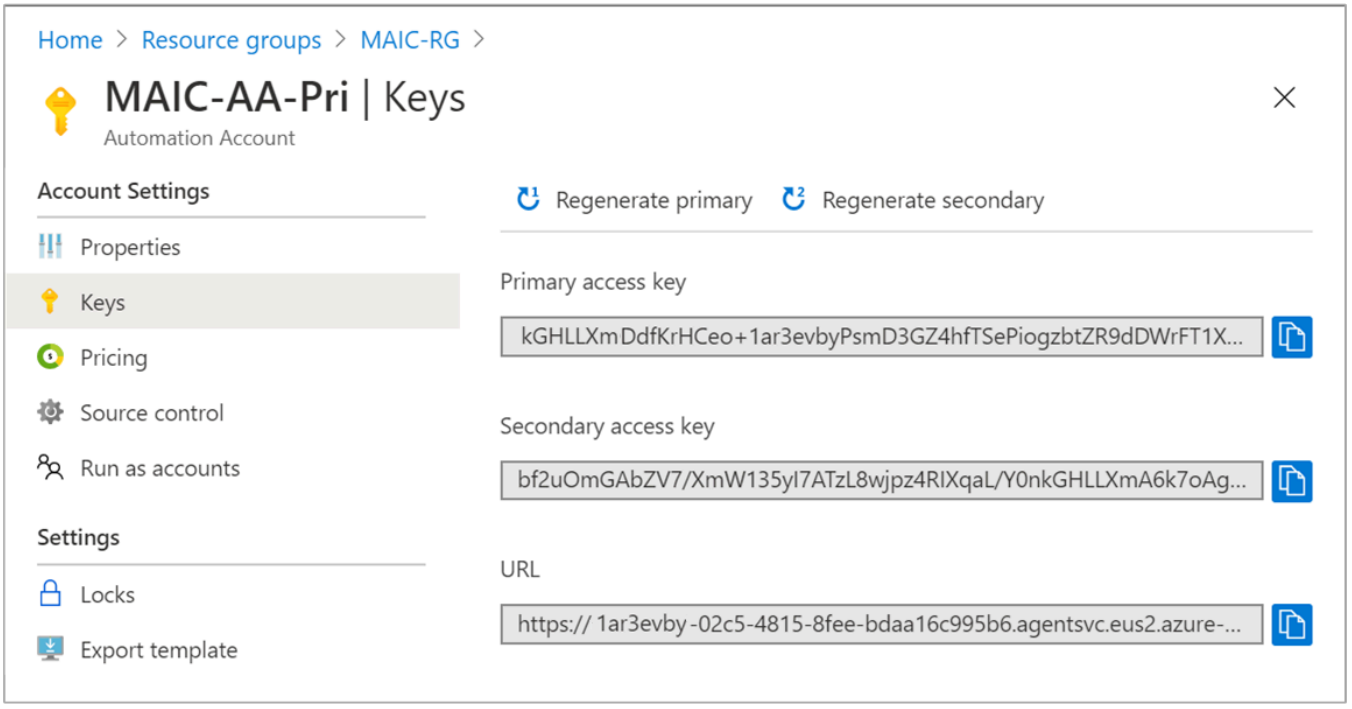
Oeste de EE. UU. 3	`https://<accountId>.webhook.usw3.azure-automation.net`
	`https://<accountId>.agentsvc.usw3.azure-automation.net`
	`https://<accountId>.jrds.usw3.azure-automation.net`
		usw3

US Gov - Virginia	`https://<accountId>.webhook.usge.azure-automation.us`
	`https://<accountId>.agentsvc.usge.azure-automation.us`
	`https://<accountId>.jrds.usge.azure-automation.us`
		usge

US Gov Texas	`https://<accountId>.webhook.ussc.azure-automation.us`
	`https://<accountId>.agentsvc.ussc.azure-automation.us`
	`https://<accountId>.jrds.ussc.azure-automation.us`
		ussc

US Gov: Arizona	`https://<accountId>.webhook.phx.azure-automation.us`
	`https://<accountId>.agentsvc.phx.azure-automation.us`
	`https://<accountId>.jrds.phx.azure-automation.us`
		phx

Reemplace `<accountId>` en el registro DNS por el GUID que representa el id. de la cuenta de Automation de la **dirección URL** del valor. Puede obtener el id. necesario de la sección **Claves** en **Configuración de la cuenta** de Azure Portal.



Copie el valor después de *accounts/* desde el campo **URL**: `https://<GUID>.agentsvc.<region>.azure-automation.net/accounts/<GUID>`

> **Nota:**
>
> Todos los registros DNS de Webhook y agentservice se han actualizado con los registros DNS de estilo nuevo para admitir Private Link. En el caso de los registros DNS de JRDS, se admiten los registros DNS de estilo antiguo y nuevo. Si no usa Private Link, verá los registros DNS de estilo antiguo, mientras que los que utilizan Private Link verán los registros DNS de estilo nuevo.

Se recomienda utilizar las direcciones mostradas al definir excepciones. Para obtener una lista de direcciones IP de regiones en lugar de nombres de regiones, descargue el archivo JSON desde el Centro de descarga de Microsoft para los siguientes entornos de nube:

- Rangos de direcciones IP y etiquetas de servicio de Azure: Azure público
- Rangos de direcciones IP y etiquetas de servicio de Azure: Azure Government
- Rangos de direcciones IP y etiquetas de servicio de Azure: Azure Alemania
- Intervalos de IP y etiquetas de servicio de Azure - Microsoft Azure operado por 21Vianet

El archivo de direcciones IP muestra los intervalos de direcciones IP que se usan en los centros de datos de Microsoft Azure. Incluye intervalos de proceso, SQL y almacenamiento, y refleja los intervalos implementados actualmente y los próximos cambios en los intervalos IP. Los nuevos intervalos que aparecen en el archivo no se utilizan en los centros de datos durante al menos una semana.

Es conveniente descargar el nuevo archivo de direcciones IP cada semana. A continuación, actualice el sitio para identificar correctamente los servicios que se ejecutan en Azure.

> **Nota:**
>
> Si utiliza Azure ExpressRoute, recuerde que el archivo de direcciones IP se usa para actualizar la publicidad del Protocolo de puerta de enlace de borde (BGP) del espacio de Azure la primera semana de cada mes.

Pasos siguientes

- Para aprender a solucionar problemas con las instancias de Hybrid Runbook Worker, consulte Solución de incidencias de Hybrid Runbook Worker.
- Para obtener información sobre cómo solucionar problemas relacionados con la configuración de estado, consulte Solución de problemas de State Configuration.

Last updated on 17/11/2025

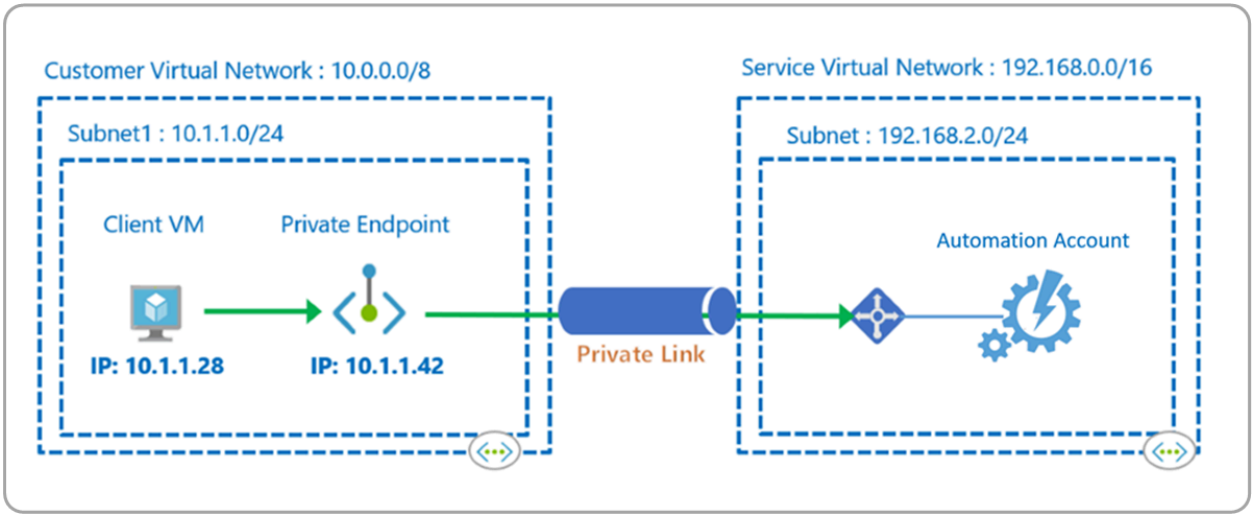
Uso de Azure Private Link para conectar redes a Azure Automation de forma segura

Artículo • 10/09/2024

Un punto de conexión privado de Azure es una interfaz de red que le conecta de forma privada y segura a un servicio con la tecnología de Azure Private Link. El punto de conexión privado usa una dirección IP privada de la red virtual para incorporar el servicio Automation de manera eficaz a su red virtual. El tráfico entre las máquinas de la red virtual y la cuenta de Automation atraviesa la red virtual y un vínculo privado de la red troncal de Microsoft, lo que elimina la exposición a la red pública de Internet.

Por ejemplo, tiene una red virtual en la que ha deshabilitado el acceso de salida a Internet. Sin embargo, quiere acceder a la cuenta de Automation de forma privada y usar características de Automation, como Webhooks, State Configuration y trabajos de runbook en Hybrid Runbook Worker. Además, quiere que los usuarios tengan acceso a la cuenta de Automation solo a través de la red virtual. La implementación de un punto de conexión privado consigue estos objetivos.

En este artículo se explica cuándo usar y cómo configurar un punto de conexión privado con su cuenta de Automation.



> **Nota**
>
> La compatibilidad de Private Link con Azure Automation solo está disponible en las nubes de Azure Commercial y Azure US Government.

Ventajas

Con Private Link puede:

- Conectarse de forma privada a Azure Automation sin necesidad de abrir ningún acceso a la red pública.
- Conectarse de forma privada al área de trabajo de Log Analytics de Azure Monitor sin necesidad de abrir ningún acceso a la red pública.

  > **Nota**
  >
  > Se necesitará un punto de conexión privado independiente para el área de trabajo de Log Analytics si la cuenta de Automation está vinculada a un área de trabajo de Log Analytics para reenviar los datos del trabajo y cuando se han habilitado características como Update Management, Seguimiento de cambios e inventario, State Configuration o Start/Stop VMs during off-hours. Para obtener más información sobre Private Link para Azure Monitor, vea Uso de Azure Private Link para conectar redes a Azure Monitor de forma segura.

- Asegurarse de que solo se accede a los datos de Automation a través de redes privadas autorizadas.
- Impedir la filtración de datos de las redes privadas mediante la definición del recurso de Azure Automation que se conecta a través del punto de conexión privado.
- Conectar de forma segura la red local privada a Azure Automation mediante ExpressRoute y Private Link.
- Mantener todo el tráfico dentro de la red troncal de Microsoft Azure.

Para más información, consulte Ventajas principales de Private Link.

Limitaciones

- En la implementación actual de Private Link, los trabajos en la nube de la cuenta de Automation no pueden acceder a los recursos de Azure que están protegidos mediante un punto de conexión privado. Por ejemplo, Azure Key Vault, Azure SQL, la cuenta de Azure Storage, etc. Para solucionar este problema, utilice una instancia de Hybrid Runbook Worker. Por lo tanto, se admiten VM locales para ejecutar

instancias de Hybrid Runbook Worker en una cuenta de Automation con Private Link habilitado.
- Deberá utilizar la versión más reciente del agente de Log Analytics para Windows o Linux.
- La puerta de enlace de Log Analytics no admite Private Link.
- Una alerta de Azure (métrica, registro y registro de actividad) no se puede usar para desencadenar un webhook de Automation cuando la cuenta de Automation está configurada con el **acceso público** establecido en **Deshabilitar**.

Funcionamiento

Private Link de Azure Automation conecta uno o más puntos de conexión privados (y, por tanto, las redes virtuales en las que se encuentran) al recurso de la cuenta de Automation. Estos puntos de conexión son máquinas que usan webhooks para iniciar un runbook, máquinas que hospedan el rol de Hybrid Runbook Worker y nodos de Desired State Configuration (DSC).

Después de crear puntos de conexión privados para Automation, cada una de las direcciones URL de Automation de acceso público se asigna a un punto de conexión privado de la red virtual. El usuario o la máquina pueden contactar directamente con las direcciones URL de Automation.

Escenario de webhook

Se pueden iniciar runbooks mediante una operación POST en la dirección URL del webhook. Por ejemplo, la dirección URL es similar a la siguiente: `https://<automationAccountId>.webhooks.<region>.azure-automation.net/webhooks?token=gzGMz4SMpqNo8gidqPxAJ3E%3d`

Escenario de Hybrid Runbook Worker

La característica Hybrid Runbook Worker de usuario de Azure Automation le permite ejecutar runbooks directamente en la máquina, sea de Azure o no, incluidos servidores registrados con otros habilitados para Azure Arc. En la máquina o servidor que hospeda el rol, puede ejecutar runbooks directamente en él y en los recursos del entorno para administrar esos recursos locales.

Hybrid Worker usa un punto de conexión de JRDS para iniciar o detener runbooks, descargarlos en el rol de trabajo y devolver el flujo de registro de trabajos al servicio Automation. Después de habilitar el punto de conexión de JRDS, la dirección URL debe tener el siguiente aspecto: `https://<automationaccountID>.jrds.`

`<region>.privatelink.azure-automation.net`. Esto garantiza la ejecución del runbook en el rol Hybrid Worker conectado a Azure Virtual Network lo cual permite ejecutar trabajos sin necesidad de abrir una conexión saliente a Internet.

> **Nota**
>
> Con la implementación actual de instancias de Private Link para Azure Automation, solo se admite la ejecución de trabajos en la instancia de Hybrid Runbook Worker conectada a una red virtual de Azure y no se admiten los trabajos en la nube.

Escenario de Hybrid Worker para Update Management

El rol Hybrid Runbook Worker del sistema admite un conjunto de runbooks ocultos que usa la característica Update Management y están diseñados para instalar las actualizaciones que el usuario especifique en máquinas Windows y Linux. Si habilita Update Management de Azure Automation, las máquinas conectadas al área de trabajo de Log Analytics se configurarán automáticamente como una instancia de Hybrid Runbook Worker del sistema.

Para comprender y configurar Update Management, consulte Acerca de Update Management. La característica Update Management tiene una dependencia sobre un área de trabajo de Log Analytics y, por tanto, es necesario vincular el área de trabajo a una cuenta de Automation. Un área de trabajo de Log Analytics almacena los datos recopilados por la solución y hospeda sus vistas y búsquedas de registros.

Si desea que las máquinas configuradas para Update Management se conecten a las áreas de trabajo de Automation y Log Analytics de forma segura a través de un canal de Private Link, debe habilitar Private Link para el área de trabajo de Log Analytics vinculada a la cuenta de Automation configurada con Private Link.

Puede controlar cómo acceder a un área de trabajo de Log Analytics desde fuera de los ámbitos de Private Link siguiendo los pasos que se indican en Configuración de Log Analytics. Si establece **Allow public network access for ingestion** (Permitir el acceso de la red pública para la ingesta) en **No**, las máquinas que se encuentren fuera de los ámbitos conectados no podrán cargar datos en esta área de trabajo. Si establece **Allow public network access for queries** (Permitir el acceso a la red pública para las consultas) en **No**, las máquinas que se encuentren fuera de los ámbitos no podrán acceder a los datos de esta área de trabajo.

Use un subrecurso de destino **DSCAndHybridWorker** para habilitar Private Link para los roles Hybrid Worker del usuario y el sistema.

> **Nota**
>
> Las máquinas que están hospedadas fuera de Azure que administra Update Management y que están conectadas a la red virtual de Azure a través de un emparejamiento privado de ExpressRoute, túneles VPN y redes virtuales emparejadas mediante puntos de conexión privados, admiten Private Link.

Escenario de State Configuration (agentsvc)

State Configuration proporciona un servicio de administración de configuración de Azure que permite escribir, administrar y compilar configuraciones de Desired State Configuration (DSC) de PowerShell para nodos de cualquier centro de datos en la nube o local.

El agente en la máquina se registra con el servicio DSC y, luego, usa el punto de conexión de servicio para extraer la configuración de DSC. El punto de conexión de servicio del agente tiene el siguiente aspecto: `https://<automationAccountId>.agentsvc.<region>.azure-automation.net`.

La dirección URL del punto de conexión privado y público sería la misma, sin embargo, cuando el vínculo privado estuviera habilitado, se asignaría a una dirección IP privada.

Planeación basada en la red

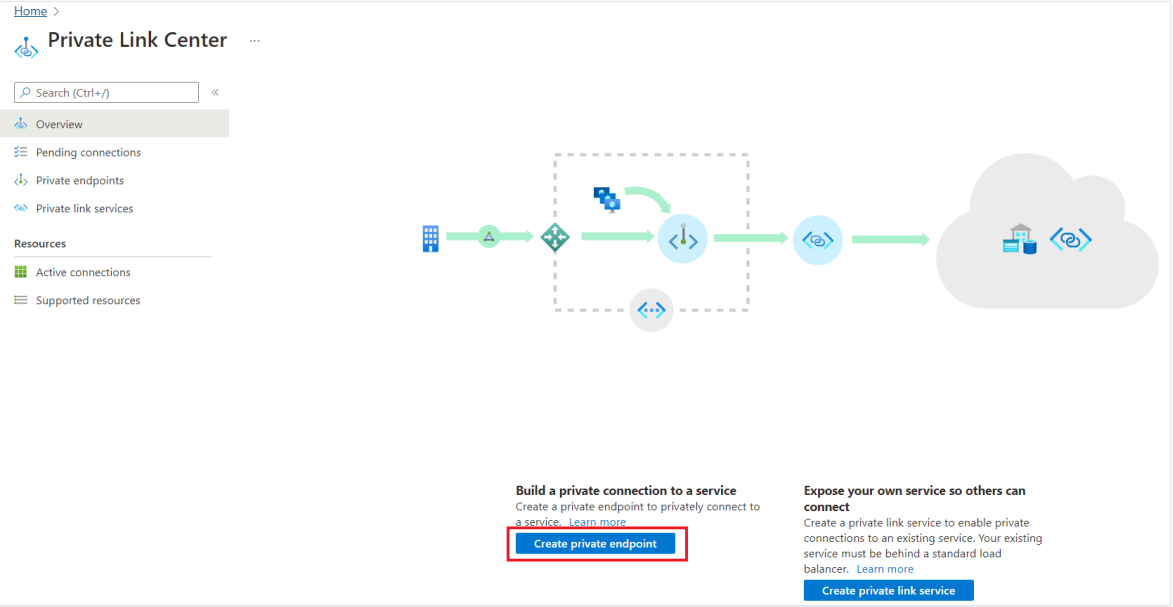
Antes de configurar los recursos de la cuenta de Automation, tenga en cuenta los requisitos de aislamiento de red. Evalúe el acceso de las redes virtuales a la red pública de Internet y las restricciones de acceso a su cuenta de Automation (incluida la configuración de un ámbito de grupo de Private Link en Registros de Azure Monitor si se ha integrado con su cuenta de Automation). Incluya también una revisión del servicio de automatización Registros DNS como parte del plan para asegurarse de que las características admitidas funcionen sin problemas.

Conexión a un punto de conexión privado

Siga los pasos que se indican a continuación para crear un punto de conexión privado para su cuenta de Automation.

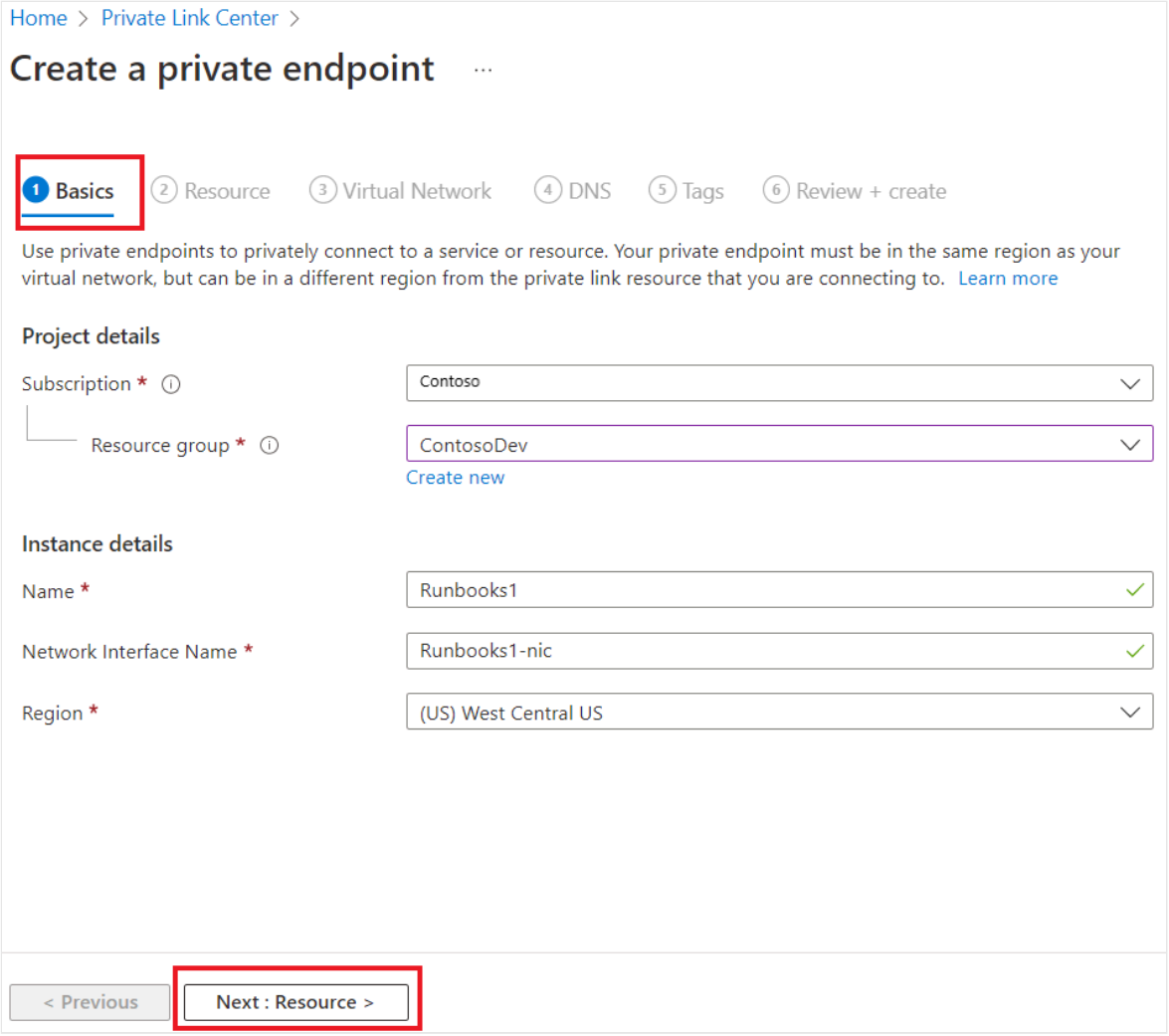
1. Vaya a Private Link Center en Azure Portal para crear un punto de conexión privado para conectar nuestra red.

2. En **Private Link Center**, seleccione **Crear un punto de conexión privado**.



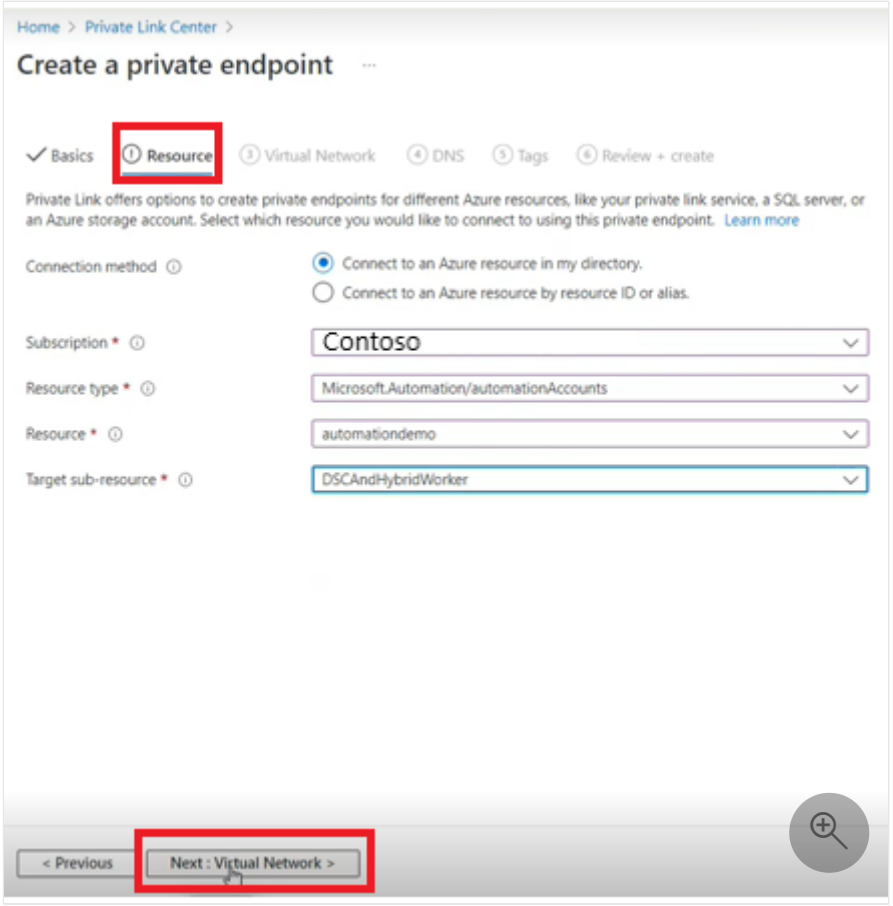
3. En **Datos básicos**, escriba la siguiente información:

   - **Suscripción**
   - **Grupos de recursos**
   - **Nombre**
   - **Nombre de la interfaz de red**
   - **Región** y seleccione **Siguiente: Recurso**.



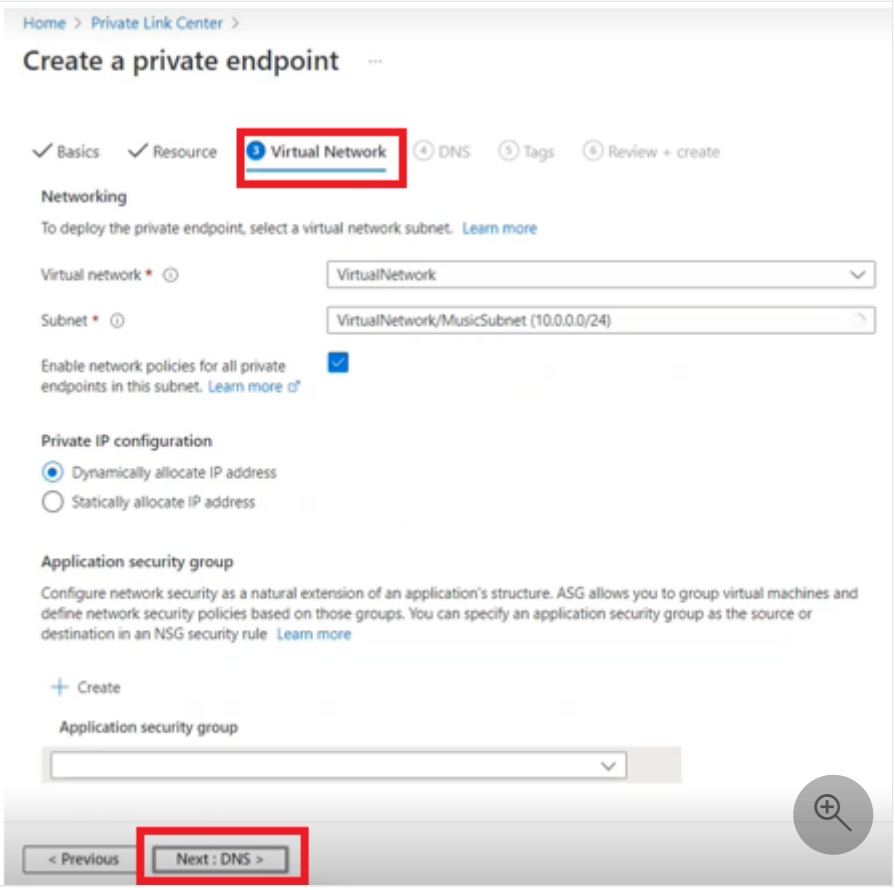
4. En **Recurso**, escriba los siguientes datos:

    - **Método de conexión**: seleccione la opción predeterminada, *Conectarse a un recurso de Azure en mi directorio*.
    - **Suscripción**
    - **Tipo de recurso**
    - **Recurso**.
    - El **Subrecurso de destino** puede ser *Webhook* o *DSCAndHybridWorker*, según el escenario. Seleccione **Siguiente: Virtual Network**.



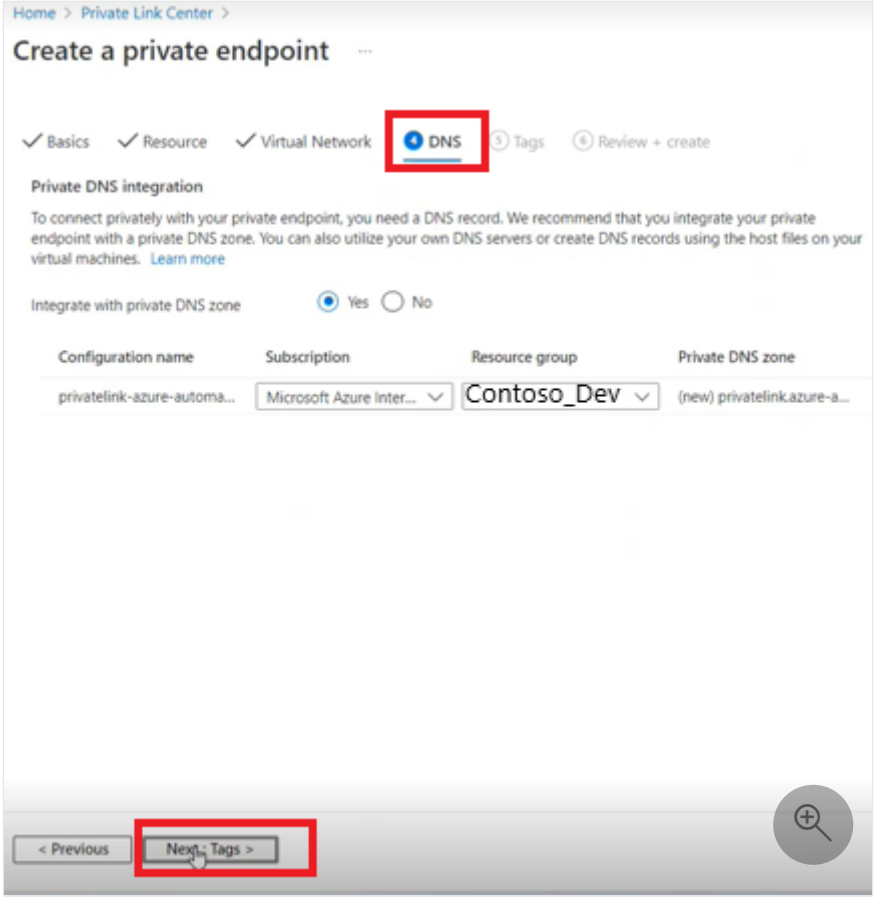
5. En **Virtual Network**, escriba los siguientes datos:

- **Red virtual**
- **Subred**
- Active la casilla **Enable network policies for all private endpoints in this subnet** (Habilitar directivas de red para todos los puntos de conexión privados de esta subred).
- Elija **Dynamically allocate IP address** (Asignar dirección IP dinámicamente) y seleccione **Siguiente: DNS**.



6. En **DNS**, los datos se rellenan según la información especificada en las pestañas **Datos básicos**, **Recurso** y **Virtual Network**, y se crea una zona de DNS privado. Escriba la siguiente información:

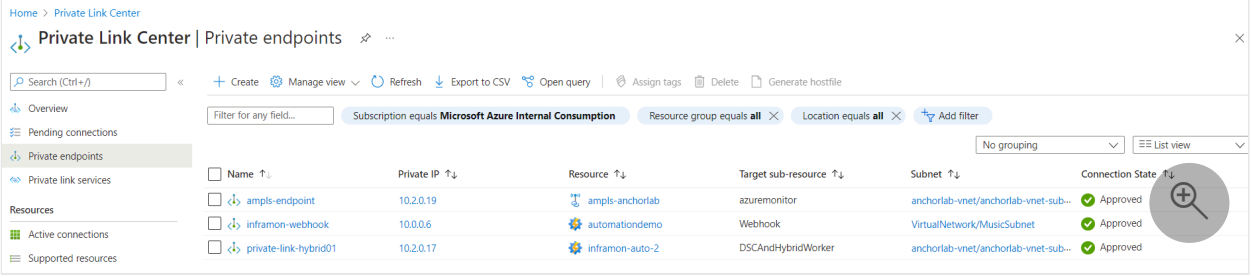
    - **Integrar con la zona DNS privada**
    - **Suscripción**
    - **Grupo de recursos** y seleccione **Siguiente: Etiquetas**



7. En **Etiquetas**, puede clasificar los recursos. Seleccione **Nombre** y **Valor**, y elija **Revisar y crear**.

Se le remitirá a la página **Revisar y crear**, donde Azure validará la configuración. Una vez aplicados los cambios del acceso de red público y de Private Link, pueden tardar hasta 35 minutos en surtir efecto.

En **Private Link Center**, seleccione **Puntos de conexión privados** para ver el recurso de Private Link.



Seleccione el recurso para ver todos los detalles. Esto crea un punto de conexión privado nuevo para la cuenta de Automation y le asigna una IP privada de la red virtual. El **estado de la conexión** se muestra como **aprobado**.

Del mismo modo, se crea un nombre de dominio completo (FQDN) único para State Configuration (agentsvc) y para el tiempo de ejecución del trabajo de Hybrid Runbook

Worker (jrds). Cada uno de ellos tiene asignado una IP independiente de la red virtual y el **estado de conexión** se muestra como **aprobado**.

Si el consumidor del servicio tiene permisos de RBAC de Azure en el recurso de Automation, podrá elegir el método de aprobación automático. En este caso, cuando el recurso de proveedor de Automation recibe la solicitud, no se requerirá ninguna acción del proveedor de servicios y la conexión se aprobará automáticamente.

Establecimiento de marcas de acceso de red

Se puede configurar una cuenta de Automation para denegar todas las configuraciones públicas y permitir solo las conexiones a través de puntos de conexión privados a fin de mejorar aún más la seguridad de la red. Si quiere permitir el acceso a la cuenta de Automation solo desde dentro de la red virtual y denegarlo desde la red pública de Internet, puede establecer la propiedad `publicNetworkAccess` en `$false`.

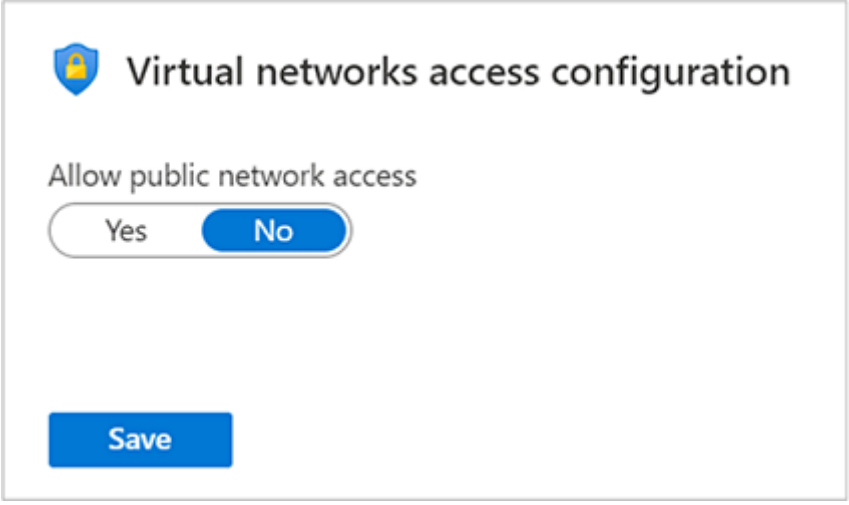
Cuando la opción **Public Network Access** (Acceso de red público) está establecida en `$false`, solo se permiten conexiones a través de puntos de conexión privados y todas las conexiones a través de puntos de conexión públicos se deniegan con un mensaje de error Sin autorización y el estado HTTP 401.

En el siguiente script de PowerShell se muestra cómo aplicar las operaciones `Get` y `Set` a la propiedad **Acceso a la red pública** en el nivel de cuenta de Automation:

PowerShell

```
$account = Get-AzResource -ResourceType
Microsoft.Automation/automationAccounts -ResourceGroupName "
<resourceGroupName>" -Name "<automationAccountName>" -ApiVersion "2020-01-
13-preview"
$account.Properties | Add-Member -Name 'publicNetworkAccess' -Type
NoteProperty -Value $false -Force
$account | Set-AzResource -Force -ApiVersion "2020-01-13-preview"
```

También puede controlar la propiedad de acceso a la red pública en Azure Portal. En la cuenta de Automation, seleccione **Aislamiento de red** en el panel izquierdo, en la sección **Configuración de cuenta**. Si la opción Public Network Access (Acceso a la red pública) está establecida en **No**, solo se permiten conexiones a través de puntos de conexión privados y todas las conexiones a través de puntos de conexión públicos se deniegan.



Configuración de DNS

Al conectarse a un recurso de vínculo privado mediante un nombre de dominio completo (FQDN) como parte de la cadena de conexión, es importante establecer correctamente la configuración de DNS para que se resuelva en la dirección IP privada asignada. Es posible que los servicios de Azure existentes ya tengan una configuración de DNS que usar al conectarse a través de un punto de conexión público. La configuración de DNS debe revisarse y actualizarse para conectarse con el punto de conexión privado.

La interfaz de red asociada con el punto de conexión privado contiene el conjunto completo de información necesaria para configurar el DNS, incluidos el FQDN y las direcciones IP privadas asignadas a un recurso de vínculo privado determinado.

Puede usar las siguientes opciones para establecer la configuración de DNS para los puntos de conexión privados:

- Use el archivo de host (solo se recomienda para realizar pruebas). Puede usar el archivo de host en una máquina virtual a fin de invalidar el uso del DNS en primer lugar para la resolución de nombres. La entrada del DNS debe tener un aspecto similar al ejemplo siguiente: `privatelinkFQDN.jrds.sea.azure-automation.net`.

- Use una zona DNS privada. Puede usar zonas DNS privadas para invalidar la resolución DNS de un punto de conexión privado determinado. Una zona DNS privada se puede vincular a la red virtual para resolver dominios específicos. Para permitir que el agente en la máquina virtual se comunique a través del punto de conexión privado, cree un registro de DNS privado como `privatelink.azure-automation.net`. Agregue una nueva asignación de registros *A* de DNS a la IP del punto de conexión privado.

- Use el reenviador DNS (opcional). Puede usar su reenviador DNS a fin de invalidar la resolución DNS para un determinado recurso de vínculo privado. Si el servidor DNS se hospeda en una red virtual, puede crear una regla de reenvío de DNS para usar una zona DNS privada con el fin de simplificar la configuración de todos los recursos de vínculo privado.

Para obtener más información, vea Configuración de DNS para puntos de conexión privados de Azure.

Pasos siguientes

Para más información sobre los puntos de conexión privados, vea ¿Qué es un punto de conexión privado de Azure?.

Comentarios

¿Le ha resultado útil esta página? Sí No

Proporcionar comentarios sobre el producto | Obtener ayuda en Microsoft Q&A

Administración de seguridad y permisos de roles en Azure Automation

El control de acceso basado en rol de Azure (RBAC de Azure) permite la administración del acceso a los recursos de Azure. Con RBAC de Azure se pueden separar las tareas dentro del equipo y conceder a los usuarios, los grupos y las aplicaciones únicamente el nivel de acceso que necesitan para realizar su trabajo. El acceso basado en rol se puede conceder a los usuarios mediante Azure Portal, las herramientas de la línea de comandos de Azure o las API de administración de Azure.

Roles en cuentas de Automation

En Azure Automation, el acceso se concede mediante la asignación del rol de Azure adecuado a los usuarios, grupos y aplicaciones en el ámbito de las cuentas de Automation. Estos son los roles integrados compatibles que admiten las cuentas de Automation:

Expandir tabla

Rol	Descripción
Propietario	El rol Propietario permite el acceso a todos los recursos y las acciones de una cuenta de Automation, lo que incluye proporcionar acceso a otros usuarios, grupos y aplicaciones para que administren la cuenta de Automation.
Colaborador	El rol Colaborador permite administrar todo, excepto la modificación de los permisos de acceso de otros usuarios a una cuenta de Automation.
Lector	El rol Lector permite ver todos los recursos de una cuenta de Automation, pero no realizar cambios.
Colaborador de Automation	El rol Colaborador de Automation permite administrar todos los recursos de la cuenta de Automation, excepto modificar los permisos de acceso de otros usuarios a una cuenta de Automation.
Operador de Automation	El rol de función de Operador de Automation permite ver las propiedades y el nombre del runbook, y crear y administrar trabajos para todos los runbooks de una cuenta de Automation. Este rol es útil si se desea proteger los recursos de la cuenta de Automation, como los recursos de credenciales y los runbooks, para que no puedan verse ni modificarse, pero permitir a los miembros de la organización ejecutar los runbooks.
Operador de trabajos de Automation	El rol de Operador de trabajos de Automation permite crear y administrar trabajos para todos los runbooks de una cuenta de Automation.

Rol	Descripción
Operador de runbooks de Automation	El rol de Operador de runbooks de Automation permite leer el nombre y las propiedades de los runbooks.
Colaborador de Log Analytics	El rol de colaborador de Log Analytics permite leer todos los datos de supervisión y editar la configuración de supervisión. La edición de la configuración de supervisión incluye la posibilidad de agregar la extensión de máquina virtual a las máquinas virtuales, leer las claves de las cuentas de almacenamiento para poder configurar la recopilación de registros de Azure Storage, crear y configurar cuentas de Automation, agregar características de Azure Automation y configurar Azure Diagnostics en todos los recursos de Azure.
Lector de Log Analytics	El rol de lector de Log Analytics permite ver y buscar todos los datos de supervisión, así como ver la configuración de supervisión. Esto incluye la visualización de la configuración de Azure Diagnostics en todos los recursos de Azure.
Colaborador de supervisión	El rol de colaborador de supervisión permite leer todos los datos de supervisión y actualizar la configuración de supervisión.
Lector de supervisión	El rol de lector de supervisión permite leer todos los datos de supervisión.
Administrador de acceso de usuario	El rol Administrador de acceso de usuario permite administrar el acceso de los usuarios a las cuentas de Azure Automation.

Permisos de los roles

Las tablas siguientes describen los permisos específicos concedidos a cada rol. Esto puede incluir Acciones, que conceden permisos, y No acciones, que los restringen.

Propietario

Un propietario puede administrar todo, incluido el acceso. La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
Microsoft.Automation/automationAccounts/*	Crear y administrar recursos de todos los tipos.

Colaborador

Un colaborador puede administrar todo, excepto el acceso. La siguiente tabla muestra los permisos concedidos y denegados para el rol:

Expandir tabla

Acciones	Descripción
Microsoft.Automation/automationAccounts/*	Crear y administrar recursos de todos los tipos
No acciones	
Microsoft.Authorization/*/Delete	Eliminar roles y asignaciones de roles.
Microsoft.Authorization/*/Write	Crear roles y asignaciones de roles.
Microsoft.Authorization/elevateAccess/Action	Deniega la capacidad de crear un administrador de acceso de usuario.

Lector

> **Nota:**
>
> Recientemente se ha realizado un cambio en el permiso de rol lector integrado para la cuenta de Automation. Más información

Un lector puede ver todos los recursos de una cuenta de Automation, pero no realizar cambios.

Expandir tabla

Acciones	Descripción
Microsoft.Automation/automationAccounts/read	Ver todos los recursos en una cuenta de Automation.

Colaborador de Automation

Un Colaborador de Automation puede administrar todos los recursos de la cuenta de Automation, excepto el acceso. La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
Microsoft.Automation/automationAccounts/*	Crear y administrar recursos de todos los tipos.

Acciones	Descripción
Microsoft.Authorization/*/read	Roles de lectura y asignaciones de roles.
Microsoft.Resources/deployments/*	Crear y administrar implementaciones de grupos de recursos.
Microsoft.Resources/subscriptions/resourceGroups/read	Leer implementaciones de grupos de recursos.
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico.
Microsoft.Insights/ActionGroups/*	Leer, escribir o eliminar grupos de acción.
Microsoft.Insights/ActivityLogAlerts/*	Leer, escribir o eliminar alertas de registro de actividad.
Microsoft.Insights/diagnosticSettings/*	Configuración de diagnóstico de lectura, escritura y eliminación.
Microsoft.Insights/MetricAlerts/*	Leer, escribir o eliminar alertas de métricas casi en tiempo real.
Microsoft.Insights/ScheduledQueryRules/*	Lea, escriba o elimine alertas de registro en Azure Monitor.
Microsoft.OperationalInsights/workspaces/sharedKeys/action	Enumerar las claves de un área de trabajo de Log Analytics.

Nota:

El rol Colaborador de Automation se puede usar para acceder a cualquier recurso mediante la identidad administrada, si se establecen los permisos adecuados en el recurso de destino, o mediante una cuenta de ejecución. Las cuentas de ejecución de Automation están configuradas de forma predeterminada con derechos de colaborador en la suscripción. Siga el principio de privilegios mínimos y asigne cuidadosamente solo los permisos necesarios para ejecutar el runbook. Por ejemplo, si la cuenta de Automation solo es necesaria para iniciar o detener una VM de Azure, los permisos asignados a la cuenta de ejecución o a la identidad administrada solo deben ser para iniciar o detener la VM. De manera similar, si un runbook lee del almacenamiento de blobs, asigne permisos de solo lectura.

Al asignar permisos, se recomienda usar el control de acceso basado en rol (RBAC) de Azure asignado a una identidad administrada. Revise nuestras recomendaciones sobre el

mejor enfoque para usar una identidad administrada asignada por el sistema o por el usuario, incluida la administración y la gobernanza durante su vigencia.

Operador de Automation

El rol de Operador de Automation permite crear y administrar trabajos, y leer las propiedades y el nombre del runbook de todos los runbooks de una cuenta de Automation.

> **Nota:**
>
> Si desea controlar el acceso de operador a runbooks individuales, no establezca este rol. En su lugar, use los roles **operador de trabajo de Automation** y el **operador de runbook de Automation** en combinación.

La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
Microsoft.Authorization/*/read	Autorización de lectura.
Microsoft.Automation/automationAccounts/hybridRunbookWorkerGroups/read	Lee los recursos de Hybrid Runbook Worker.
Microsoft.Automation/automationAccounts/jobs/read	Enumerar trabajos del runbook.
Microsoft.Automation/automationAccounts/jobs/resume/action	Reanudar un trabajo que está en pausa.
Microsoft.Automation/automationAccounts/jobs/stop/action	Cancelar un trabajo en curso.
Microsoft.Automation/automationAccounts/jobs/streams/read	Leer los flujos de trabajo y los resultados.
Microsoft.Automation/automationAccounts/jobs/output/read	Obtiene la salida de un trabajo.
Microsoft.Automation/automationAccounts/jobs/suspend/action	Pausar un trabajo en curso.
Microsoft.Automation/automationAccounts/jobs/write	Crear trabajos.

Acciones	Descripción
Microsoft.Automation/automationAccounts/jobSchedules/read	Obtiene una programación de trabajos de Azure Automation.
Microsoft.Automation/automationAccounts/jobSchedules/write	Crea una programación de trabajos de Azure Automation.
Microsoft.Automation/automationAccounts/linkedWorkspace/read	Obtiene el área de trabajo vinculada a la cuenta de Automation.
Microsoft.Automation/automationAccounts/read	Obtiene una cuenta de Azure Automation.
Microsoft.Automation/automationAccounts/runbooks/read	Obtiene un runbook de Azure Automation.
Microsoft.Automation/automationAccounts/schedules/read	Obtiene un recurso de programación de Azure Automation.
Microsoft.Automation/automationAccounts/schedules/write	Crea o actualiza un recurso de programación de Azure Automation.
Microsoft.Resources/subscriptions/resourceGroups/read	Roles de lectura y asignaciones de roles.
Microsoft.Resources/deployments/*	Crear y administrar implementaciones de grupos de recursos.
Microsoft.Insights/alertRules/*	Crear y administrar reglas de alertas.
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico.
Microsoft.ResourceHealth/availabilityStatuses/read	Obtiene los estados de disponibilidad de todos los recursos del ámbito especificado.

Operador de trabajos de Automation

El rol de Operador de trabajos de Automation se asigna en el ámbito de la cuenta de Automation. Esto permite a los roles con permiso de Operador crear y administrar trabajos para todos los runbooks de la cuenta. Si a la función de operador de trabajo se le conceden permisos de lectura en el grupo de recursos que contiene la cuenta de Automation, los miembros del rol tendrán la capacidad de iniciar runbooks. Pero no tienen la capacidad de crearlos, modificarlos ni eliminarlos.

La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
Microsoft.Authorization/*/read	Autorización de lectura.
Microsoft.Automation/automationAccounts/jobs/read	Enumerar trabajos del runbook.
Microsoft.Automation/automationAccounts/jobs/resume/action	Reanudar un trabajo que está en pausa.
Microsoft.Automation/automationAccounts/jobs/stop/action	Cancelar un trabajo en curso.
Microsoft.Automation/automationAccounts/jobs/streams/read	Leer los flujos de trabajo y los resultados.
Microsoft.Automation/automationAccounts/jobs/suspend/action	Pausar un trabajo en curso.
Microsoft.Automation/automationAccounts/jobs/write	Crear trabajos.
Microsoft.Resources/subscriptions/resourceGroups/read	Roles de lectura y asignaciones de roles.
Microsoft.Resources/deployments/*	Crear y administrar implementaciones de grupos de recursos.
Microsoft.Insights/alertRules/*	Crear y administrar reglas de alertas.
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico.
Microsoft.Automation/automationAccounts/hybridRunbookWorkerGroups/read	Lee un grupo de Hybrid Runbook Worker.

Acciones	Descripción
Microsoft.Automation/automationAccounts/jobs/output/read	Obtiene la salida de un trabajo.

Operador de runbooks de Automation

El rol de operador de runbooks de Automation se concede en el ámbito del runbook. Un Operador de runbooks de Automation puede ver las propiedades y el nombre del runbook. Este rol, combinado con el de **operador de trabajos de Automation**, permite también al operador crear y administrar trabajos para el runbook. La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
Microsoft.Automation/automationAccounts/runbooks/read	Enumerar los runbooks.
Microsoft.Authorization/*/read	Autorización de lectura.
Microsoft.Resources/subscriptions/resourceGroups/read	Roles de lectura y asignaciones de roles.
Microsoft.Resources/deployments/*	Crear y administrar implementaciones de grupos de recursos.
Microsoft.Insights/alertRules/*	Crear y administrar reglas de alertas.
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico.

Colaborador de Log Analytics

Un colaborador de Log Analytics puede leer todos los datos de supervisión y editar la configuración de supervisión. La edición de la configuración de supervisión incluye la posibilidad de agregar la extensión de máquina virtual a las máquinas virtuales, leer las claves de las cuentas de almacenamiento para poder configurar la recopilación de registros de Azure Storage, crear y configurar cuentas de Automation, agregar características y configurar Azure Diagnostics en todos los recursos de Azure. La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
*/read	Leer recursos de todos los tipos, excepto secretos.
Microsoft.ClassicCompute/virtualMachines/extensions/*	Crear y administrar extensiones de máquinas virtuales.
Microsoft.ClassicStorage/storageAccounts/listKeys/action	Enumerar claves de cuentas de almacenamiento clásicas.
Microsoft.Compute/virtualMachines/extensions/*	Crear y administrar extensiones de máquinas virtuales clásicas.
Microsoft.Insights/alertRules/*	Reglas de alerta de lectura, escritura y eliminación.
Microsoft.Insights/diagnosticSettings/*	Configuración de diagnóstico de lectura, escritura y eliminación.
Microsoft.OperationalInsights/*	Administrar registros de Azure Monitor.
Microsoft.OperationsManagement/*	Administración de características de Azure Automation en áreas de trabajo.
Microsoft.Resources/deployments/*	Crear y administrar implementaciones de grupos de recursos.
Microsoft.Resources/subscriptions/resourcegroups/deployments/*	Crear y administrar implementaciones de grupos de recursos.
Microsoft.Storage/storageAccounts/listKeys/action	Enumerar claves de cuentas de almacenamiento.
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico.
Microsoft.HybridCompute/machines/extensions/write	Instala o actualiza las extensiones de Azure Arc.

Lector de Log Analytics

Un lector de Log Analytics puede ver y buscar los datos de supervisión, así como consultar la configuración, incluida la de Azure Diagnostics en todos los recursos de Azure. La siguiente tabla muestra los permisos concedidos o denegados para el rol:

Expandir tabla

Acciones	Descripción
*/read	Leer recursos de todos los tipos, excepto secretos.
Microsoft.OperationalInsights/workspaces/analytics/query/action	Administrar consultas de búsqueda en registros de Azure Monitor.
Microsoft.OperationalInsights/workspaces/search/action	Buscar datos de registros de Azure Monitor.
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico.
No acciones	
Microsoft.OperationalInsights/workspaces/sharedKeys/read	No se pueden leer las claves de acceso compartido.

Colaborador de supervisión

Un colaborador de supervisión puede leer todos los datos de supervisión y actualizar la configuración de supervisión. La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
*/read	Leer recursos de todos los tipos, excepto secretos.
Microsoft.AlertsManagement/alerts/*	Administrar alertas.
Microsoft.AlertsManagement/alertsSummary/*	Administrar el panel de alertas.
Microsoft.Insights/AlertRules/*	Administrar reglas de alertas.
Microsoft.Insights/components/*	Administrar componentes de Application Insights.
Microsoft.Insights/DiagnosticSettings/*	Administrar la configuración de diagnóstico.
Microsoft.Insights/eventtypes/*	Enumerar eventos del registro de actividades (eventos de administración) de una suscripción. Este permiso es aplicable para el acceso mediante programación y

Acciones	Descripción
	mediante el portal al registro de actividades.
Microsoft.Insights/LogDefinitions/*	Este permiso es necesario para los usuarios que necesitan acceder a registros de actividades a través del portal. Enumere las categorías de registro del registro de actividad.
Microsoft.Insights/MetricDefinitions/*	Leer definiciones de métrica (lista de tipos de métricas disponibles para un recurso).
Microsoft.Insights/Metrics/*	Leer las métricas de un recurso.
Microsoft.Insights/Register/Action	Registrar el proveedor de Microsoft.Insights.
Microsoft.Insights/webtests/*	Administrar pruebas web de Application Insights.
Microsoft.OperationalInsights/workspaces/intelligencepacks/*	Administrar paquetes de soluciones de registros de Azure Monitor.
Microsoft.OperationalInsights/workspaces/savedSearches/*	Administrar búsquedas guardadas de registros de Azure Monitor.
Microsoft.OperationalInsights/workspaces/search/action	Buscar áreas de trabajo de Log Analytics.
Microsoft.OperationalInsights/workspaces/sharedKeys/action	Enumerar las claves de un área de trabajo de Log Analytics.
Microsoft.OperationalInsights/workspaces/storageinsightconfigs/*	Administrar configuraciones de visión de almacenamiento de registros de Azure Monitor.
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico.
Microsoft.WorkloadMonitor/workloads/*	Administrar cargas de trabajo.

Lector de supervisión

Un lector de supervisión puede leer todos los datos de supervisión. La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
*/read	Leer recursos de todos los tipos, excepto secretos.
Microsoft.OperationalInsights/workspaces/search/action	Buscar áreas de trabajo de Log Analytics.
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico

Administrador de acceso de usuario

Un administrador de acceso de usuario puede administrar el acceso de los usuarios a los recursos de Azure. La siguiente tabla muestra los permisos concedidos para el rol:

Expandir tabla

Acciones	Descripción
*/read	Leer todos los recursos.
Microsoft.Authorization/*	Administrar la autorización
Microsoft.Support/*	Crear y administrar incidencias de soporte técnico

Permisos de acceso de rol de lector

> **Importante**
>
> Para reforzar la posición de seguridad general de Azure Automation, el lector de RBAC integrado no tendría acceso a las claves de cuenta de Automation mediante la llamada API: `GET /AUTOMATIONACCOUNTS/AGENTREGISTRATIONINFORMATION`.

El rol de lector integrado de la cuenta de Automation no puede usar `API – GET /AUTOMATIONACCOUNTS/AGENTREGISTRATIONINFORMATION` para capturar las claves de la cuenta de Automation. Se trata de una operación con privilegios elevados que proporciona información confidencial que podría suponer un riesgo para la seguridad si un actor malintencionado no deseado con pocos privilegios obtuviera acceso a las claves de cuenta de Automation y realizara acciones con un nivel de privilegios elevados.

Para acceder a `API – GET /AUTOMATIONACCOUNTS/AGENTREGISTRATIONINFORMATION`, se recomienda cambiar a los roles integrados, como Propietario, Colaborador o Colaborador de Automation, para acceder a las claves de cuenta de Automation. Estos roles, de forma predeterminada, tendrán el permiso *listKeys*. Como procedimiento recomendado, es mejor crear un rol personalizado con permisos limitados para acceder a las claves de cuenta de Automation. En este caso, debe agregar un permiso `Microsoft.Automation/automationAccounts/listKeys/action` a la definición de roles. Más información sobre cómo crear un rol personalizado en Azure Portal.

Permisos de configuración de características

En las secciones siguientes se describen los permisos mínimos necesarios para habilitar las características Update Management y Change Tracking e Inventario.

Permisos para habilitar Update Management y Change Tracking e Inventario en una máquina virtual

Expandir tabla

Acción	Permiso	Ámbito mínimo
Escribir nueva implementación	Microsoft.Resources/deployments/*	Suscripción
Escribir nuevo grupo de recursos	Microsoft.Resources/subscriptions/resourceGroups/write	Suscripción
Crear nueva área de trabajo predeterminada	Microsoft.OperationalInsights/workspaces/write	Grupo de recursos
Crear nueva cuenta	Microsoft.Automation/automationAccounts/write	Grupo de recursos
Vincular área de trabajo y cuenta	Microsoft.OperationalInsights/workspaces/write Microsoft.Automation/automationAccounts/read	Área de trabajo Cuenta de Automation
Crear extensión MMA	Microsoft.Compute/virtualMachines/write	Máquina virtual
Crear búsqueda guardada	Microsoft.OperationalInsights/workspaces/write	Área de trabajo

Acción	Permiso	Ámbito mínimo
Crear configuración de ámbito	Microsoft.OperationalInsights/workspaces/write	Área de trabajo
Comprobación de estado de incorporación: leer área trabajo	Microsoft.OperationalInsights/workspaces/read	Área de trabajo
Comprobación de estado de incorporación: leer propiedad de área de trabajo vinculada de la cuenta	Microsoft.Automation/automationAccounts/read	Cuenta de Automation
Comprobación de estado de incorporación: leer solución	Microsoft.OperationalInsights/workspaces/intelligencepacks/read	Solución
Comprobación de estado de incorporación: leer máquina virtual	Microsoft.Compute/virtualMachines/read	Máquina virtual
Comprobación de estado de incorporación: leer cuenta	Microsoft.Automation/automationAccounts/read	Cuenta de Automation
Incorporación de comprobación de área de trabajo para VM[1]	Microsoft.OperationalInsights/workspaces/read	Suscripción
Registro del proveedor de Log Analytics	Microsoft.Insights/register/action	Suscripción

[1] Este permiso es necesario para habilitar características a través de la experiencia de máquina virtual del portal.

Permisos para habilitar Update Management y Change Tracking e inventario en una cuenta de Automation

Expandir tabla

Acción	Permiso	Ámbito mínimo
Crear una nueva implementación	Microsoft.Resources/deployments/*	Suscripción
Crear un grupo de recursos	Microsoft.Resources/subscriptions/resourceGroups/write	Suscripción
Hoja AutomationOnboarding: crear nueva área de trabajo	Microsoft.OperationalInsights/workspaces/write	Grupo de recursos
Hoja AutomationOnboarding: leer área de trabajo vinculada	Microsoft.Automation/automationAccounts/read	Cuenta de Automation
Hoja AutomationOnboarding: leer solución	Microsoft.OperationalInsights/workspaces/intelligencepacks/read	Solución
Hoja AutomationOnboarding: leer área de trabajo	Microsoft.OperationalInsights/workspaces/intelligencepacks/read	Área de trabajo
Crear vínculo para área de trabajo y cuenta	Microsoft.OperationalInsights/workspaces/write	Área de trabajo
Escribir cuenta para Shoebox	Microsoft.Automation/automationAccounts/write	Cuenta
Crear o editar búsqueda guardada	Microsoft.OperationalInsights/workspaces/write	Área de trabajo
Crear o editar la configuración de ámbito	Microsoft.OperationalInsights/workspaces/write	Área de trabajo
Registro del proveedor de Log Analytics	Microsoft.Insights/register/action	Suscripción
Paso 2: Habilitación de varias máquinas virtuales		
Hoja VMOnboarding: crear extensión MMA	Microsoft.Compute/virtualMachines/write	Máquina virtual
Crear o editar búsqueda	Microsoft.OperationalInsights/workspaces/write	Área de

Acción	Permiso	Ámbito mínimo
guardada		trabajo
Crear o editar la configuración de ámbito	Microsoft.OperationalInsights/workspaces/write	Área de trabajo

Administración de permisos de rol para grupos e instancias de Hybrid Worker

Puede crear roles personalizados de Azure en Automation y conceder los siguientes permisos a grupos de Hybrid Worker e instancias de Hybrid Worker:

- Extensiones basadas en Hybrid Runbook Worker

Configuración de RBAC de Azure para la cuenta de Automation

En la sección siguiente se muestra cómo configurar RBAC de Azure en la cuenta de Automation mediante Azure Portal y PowerShell.

Configuración de RBAC de Azure mediante Azure Portal

1. Inicie sesión en Azure Portal y abra su cuenta de Automation en la página **Cuentas de automatización**.

2. Seleccione **Control de acceso (IAM)** y seleccione un rol en la lista de roles disponibles. Puede elegir cualquiera de los roles integrados disponibles que la cuenta de Automation admita o cualquier rol personalizado que haya definido. Asigne el rol a un usuario al que desea conceder permisos.

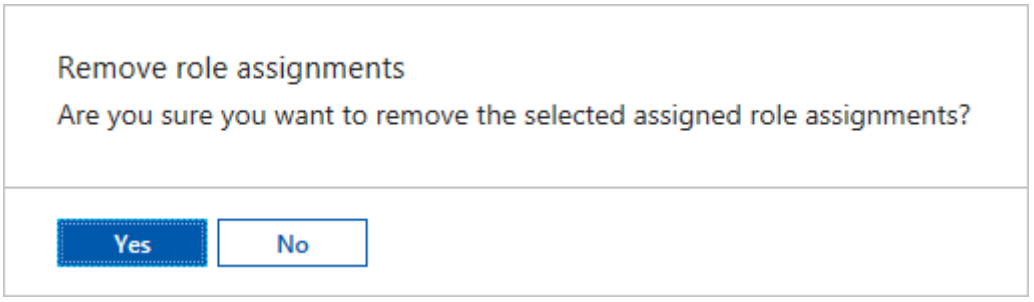
   Para acceder a los pasos detallados, consulte Asignación de roles de Azure mediante Azure Portal.

   > **Nota:**
   >
   > El control de acceso basado en roles solo se puede establecer en el ámbito de la cuenta de Automation, no en los recursos por debajo de la cuenta de Automation.

Eliminación de asignaciones de roles de un usuario

Puede quitar el permiso de usuario de cualquier usuario que no administre la cuenta de Automation o que haya dejado de trabajar en la organización. En los pasos siguientes se muestra cómo quitar las asignaciones de roles de un usuario. Para los pasos detallados, consulte Eliminación de asignaciones de roles de Azure:

1. Abra **Control de acceso (IAM)** en un ámbito como, por ejemplo, grupo de administración, suscripción, grupo de recursos o recurso, en donde quiere quitar el acceso.
2. Seleccione la pestaña **Asignaciones de roles** para ver todas las asignaciones de roles en este ámbito.
3. En la lista de asignaciones de roles, agregue una marca de verificación junto al usuario con la asignación de roles que quiera eliminar.
4. Seleccione **Quitar**.



Configuración de RBAC de Azure mediante PowerShell

El acceso basado en rol también se puede configurar en una cuenta de Automation mediante los siguientes cmdlets de Azure PowerShell:

Get-AzRoleDefinition enumera todos los roles de Azure disponibles en Microsoft Entra ID. Puede usar este cmdlet con el parámetro `Name` para enumerar todas las acciones que puede realizar un rol específico.

Azure PowerShell

```
Get-AzRoleDefinition -Name 'Automation Operator'
```

A continuación se incluye el resultado de ejemplo:

Azure PowerShell

```
Name             : Automation Operator
Id               : d3881f73-407a-4167-8283-e981cbba0404
IsCustom         : False
```

```
Description      : Automation Operators are able to start, stop, suspend, and resume
jobs
Actions          : {Microsoft.Authorization/*/read,
Microsoft.Automation/automationAccounts/jobs/read,
Microsoft.Automation/automationAccounts/jobs/resume/action,
                   Microsoft.Automation/automationAccounts/jobs/stop/action...}
NotActions       : {}
AssignableScopes : {/}
```

Get-AzRoleAssignment enumera las asignaciones de roles de Azure en el ámbito especificado. Sin parámetros, este cmdlet devuelve todas las asignaciones de roles realizadas en la suscripción. Use el parámetro `ExpandPrincipalGroups` para enumerar las asignaciones de acceso del usuario especificado, así como los grupos a los que pertenezca.

Ejemplo: use el siguiente cmdlet para enumerar todos los usuarios de una cuenta de Automation y sus roles.

Azure PowerShell

```
Get-AzRoleAssignment -Scope
'/subscriptions/<SubscriptionID>/resourcegroups/<Resource Group
Name>/Providers/Microsoft.Automation/automationAccounts/<Automation account name>'
```

A continuación se incluye el resultado de ejemplo:

PowerShell

```
RoleAssignmentId   : /subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/myResourceGroup/providers/Microsoft.Automation/automatio
nAccounts/myAutomationAccount/provid
                     ers/Microsoft.Authorization/roleAssignments/00000000-0000-0000-
0000-000000000000
Scope              : /subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/myResourceGroup/providers/Microsoft.Automation/automatio
nAccounts/myAutomationAccount
DisplayName        : admin@contoso.com
SignInName         : admin@contoso.com
RoleDefinitionName : Automation Operator
RoleDefinitionId   : d3881f73-407a-4167-8283-e981cbba0404
ObjectId           : aaaaaaaa-0000-1111-2222-bbbbbbbbbbbb
ObjectType         : User
```

Use New-AzRoleAssignment para asignar acceso a usuarios, grupos y aplicaciones en un ámbito determinado.

Ejemplo: use el siguiente comando para asignar el rol "Operador de Automation" para un usuario en el ámbito de la cuenta de Automation.

Azure PowerShell

```
New-AzRoleAssignment -SignInName <sign-in Id of a user you wish to grant access> -
RoleDefinitionName 'Automation operator' -Scope
'/subscriptions/<SubscriptionID>/resourcegroups/<Resource Group
Name>/Providers/Microsoft.Automation/automationAccounts/<Automation account name>'
```

A continuación se incluye el resultado de ejemplo:

Azure PowerShell

```
RoleAssignmentId   : /subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourcegroups/myResourceGroup/Providers/Microsoft.Automation/automatio
nAccounts/myAutomationAccount/provid
                     ers/Microsoft.Authorization/roleAssignments/00000000-0000-0000-
0000-000000000000
Scope              : /subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourcegroups/myResourceGroup/Providers/Microsoft.Automation/automatio
nAccounts/myAutomationAccount
DisplayName        : admin@contoso.com
SignInName         : admin@contoso.com
RoleDefinitionName : Automation Operator
RoleDefinitionId   : d3881f73-407a-4167-8283-e981cbba0404
ObjectId           : bbbbbbbb-1111-2222-3333-cccccccccccc
ObjectType         : User
```

Use Remove-AzRoleAssignment para quitar el acceso a un usuario, grupo o aplicación concretos de un ámbito determinado.

Ejemplo: use el siguiente comando para quitar el usuario del rol Operador de Automation en el ámbito de la cuenta de Automation.

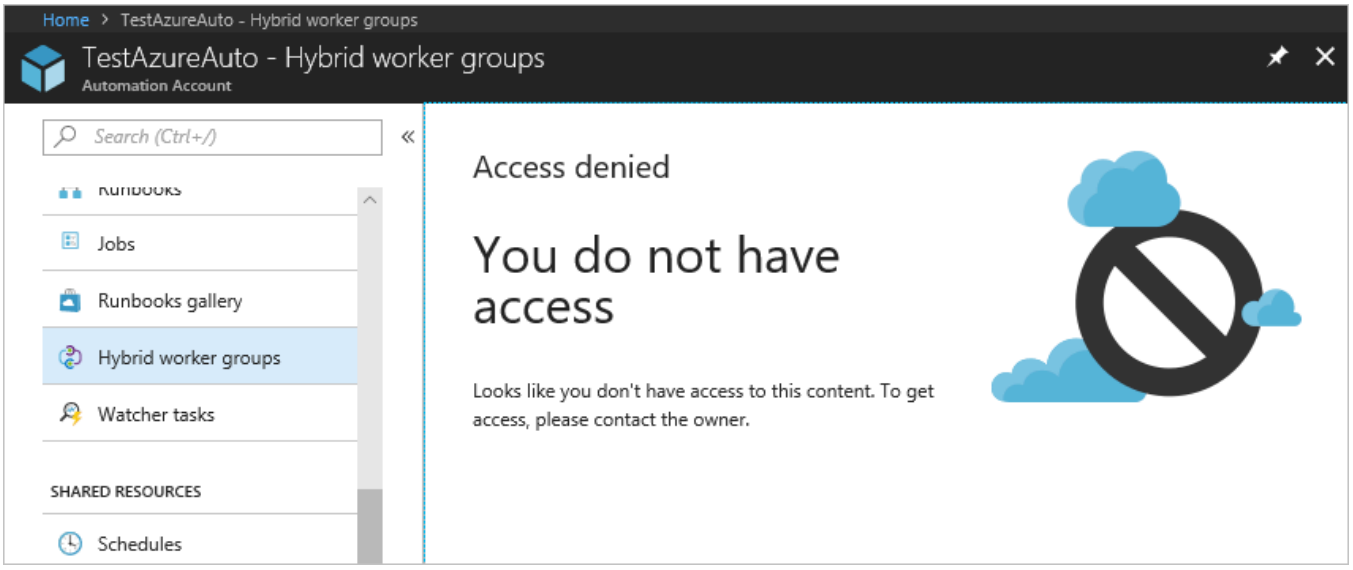
Azure PowerShell

```
Remove-AzRoleAssignment -SignInName <sign-in Id of a user you wish to remove> -
RoleDefinitionName 'Automation Operator' -Scope
'/subscriptions/<SubscriptionID>/resourcegroups/<Resource Group
Name>/Providers/Microsoft.Automation/automationAccounts/<Automation account name>'
```

En el ejemplo anterior, reemplace `sign-in ID of a user you wish to remove`, `SubscriptionID`, `Resource Group Name` y `Automation account name` por los detalles de su cuenta. Elija **Sí** cuando se le pida que confirme antes de proceder a la eliminación de las asignaciones de roles de usuario.

Experiencia del usuario en el rol Operador de Automation: cuenta de Automation

Cuando un usuario asignado al rol Operador de Automation en el ámbito de la cuenta de Automation vea la cuenta de Automation a la que está asignado, solo verá la lista de runbooks, los trabajos de runbook y las programaciones creados en la cuenta de Automation. Este usuario no puede ver las definiciones de estos elementos. El usuario puede iniciar, detener, suspender, reanudar o programar el trabajo de runbook. Pero el usuario no tiene acceso a otros recursos de Automation como configuraciones, grupos de Hybrid Runbook Worker o nodos de DSC.



Configuración de RBAC de Azure para runbooks

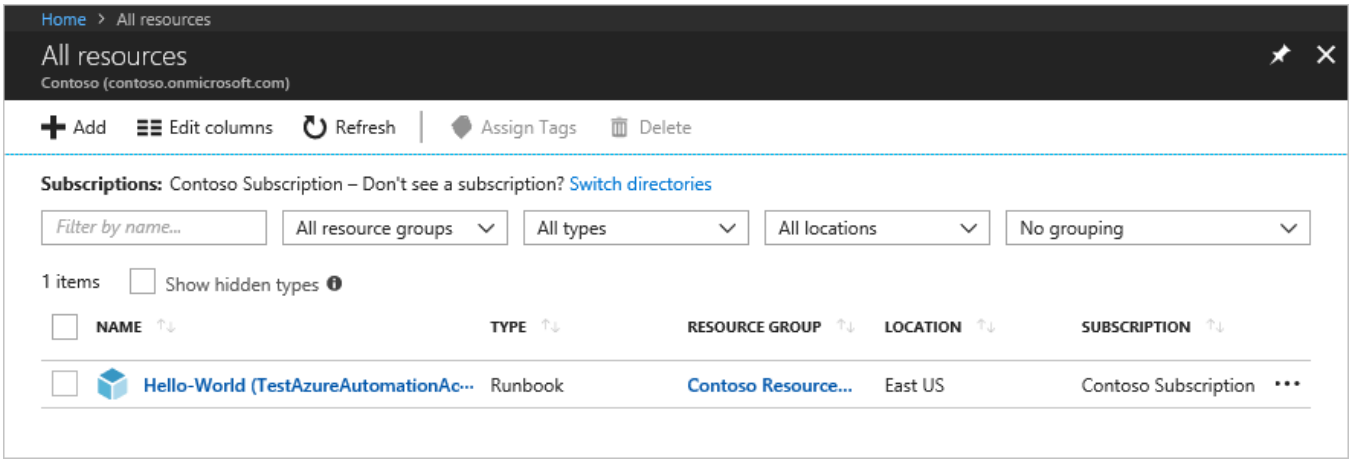
Azure Automation permite asignar roles de Azure a runbooks específicos. Para ello, ejecute el siguiente script para agregar un usuario a un runbook específico. Un administrador de la cuenta de Automation o un administrador de inquilinos pueden ejecutar este script.

Azure PowerShell

```
$rgName = "<Resource Group Name>" # Resource Group name for the Automation account
$automationAccountName ="<Automation account name>" # Name of the Automation account
$rbName = "<Name of Runbook>" # Name of the runbook
$userId = "<User ObjectId>" # Azure Active Directory (AAD) user's ObjectId from the
directory

# Gets the Automation account resource
$aa = Get-AzResource -ResourceGroupName $rgName -ResourceType
"Microsoft.Automation/automationAccounts" -ResourceName $automationAccountName

# Get the Runbook resource
$rb = Get-AzResource -ResourceGroupName $rgName -ResourceType
"Microsoft.Automation/automationAccounts/runbooks" -ResourceName "$rbName"

# The Automation Job Operator role only needs to be run once per user.
New-AzRoleAssignment -ObjectId $userId -RoleDefinitionName "Automation Job Operator"
-Scope $aa.ResourceId
```

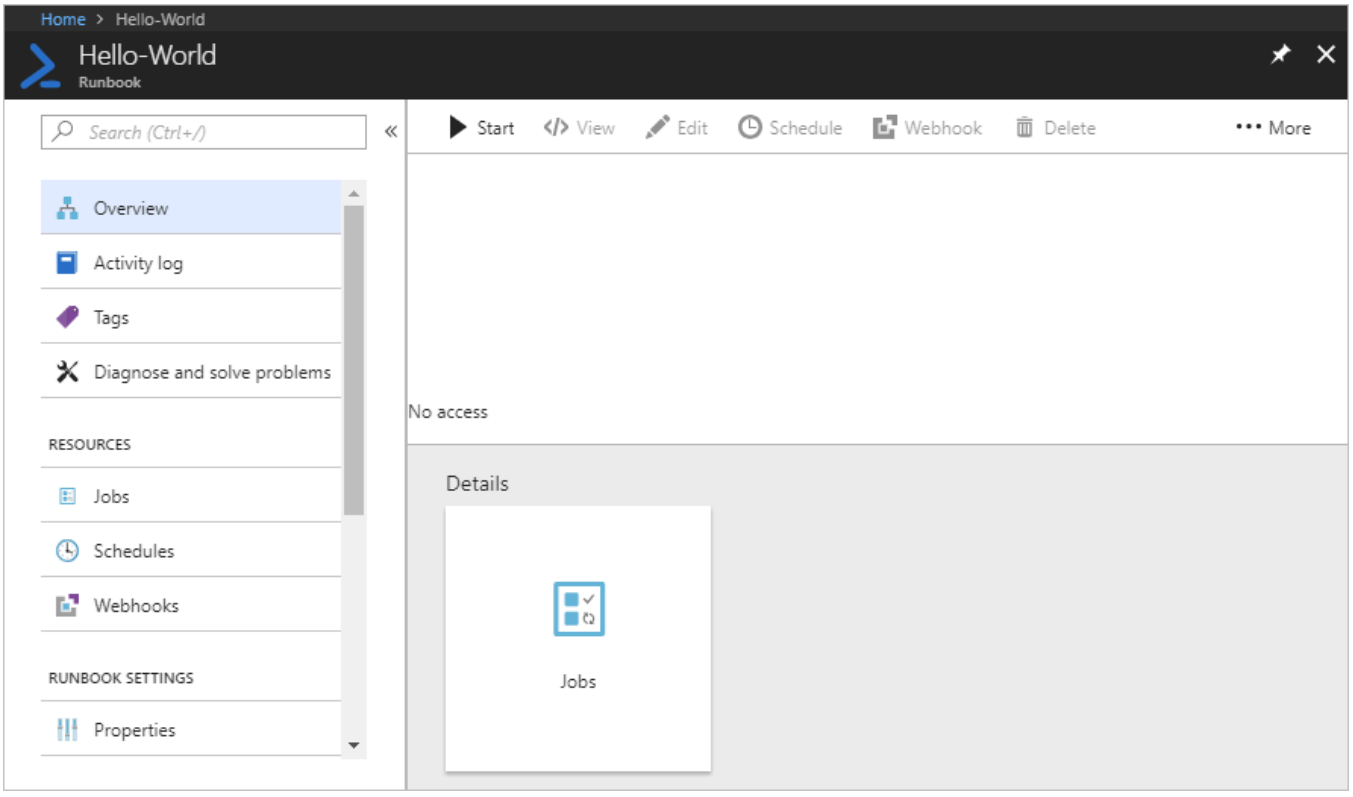
```
# Adds the user to the Automation Runbook Operator role to the Runbook scope
New-AzRoleAssignment -ObjectId $userId -RoleDefinitionName "Automation Runbook Operator" -Scope $rb.ResourceId
```

Una vez ejecutado el script, haga que el usuario inicie sesión en Azure Portal y seleccione **Todos los recursos**. En la lista, el usuario puede ver el runbook para el que se ha agregado como Operador de runbook de Automation.



Experiencia del usuario en el rol Operador de Automation: runbook

Cuando un usuario asignado al rol Operador de Automation en el ámbito del runbook vea un runbook al que esté asignado, solo podrá iniciar el runbook y ver los trabajos de runbook.



Pasos siguientes

- Para obtener información sobre las directrices de seguridad, consulte Procedimientos recomendados de seguridad en Azure Automation.
- Para obtener más información sobre RBAC de Azure con PowerShell, vea Incorporación o eliminación de asignaciones de roles de Azure con Azure PowerShell.
- Para más información sobre los tipos de runbooks, consulte Tipos de runbooks de Azure Automation.
- Para iniciar un runbook, consulte Inicio de un runbook en Azure Automation.

Last updated on 17/11/2025

Traslado de la cuenta de Azure Automation a otra suscripción

Azure Automation permite mover algunos recursos a un nuevo grupo de recursos o suscripción. Puede mover recursos a través del portal de Azure, PowerShell, el CLI de Azure o la API REST. Para más información sobre el proceso, consulte Movimiento de recursos a un grupo de recursos o suscripción nuevos.

La cuenta de Automation es uno de los recursos que se pueden mover. En este artículo, conocerá el procedimiento para mover cuentas de Automation a otro recurso o suscripción. Los pasos generales para mover la cuenta de Automation son:

1. Deshabilite sus funcionalidades.
2. Desvincule el área de trabajo.
3. Mueva la cuenta de Automatización.
4. Reactiva tus funciones.

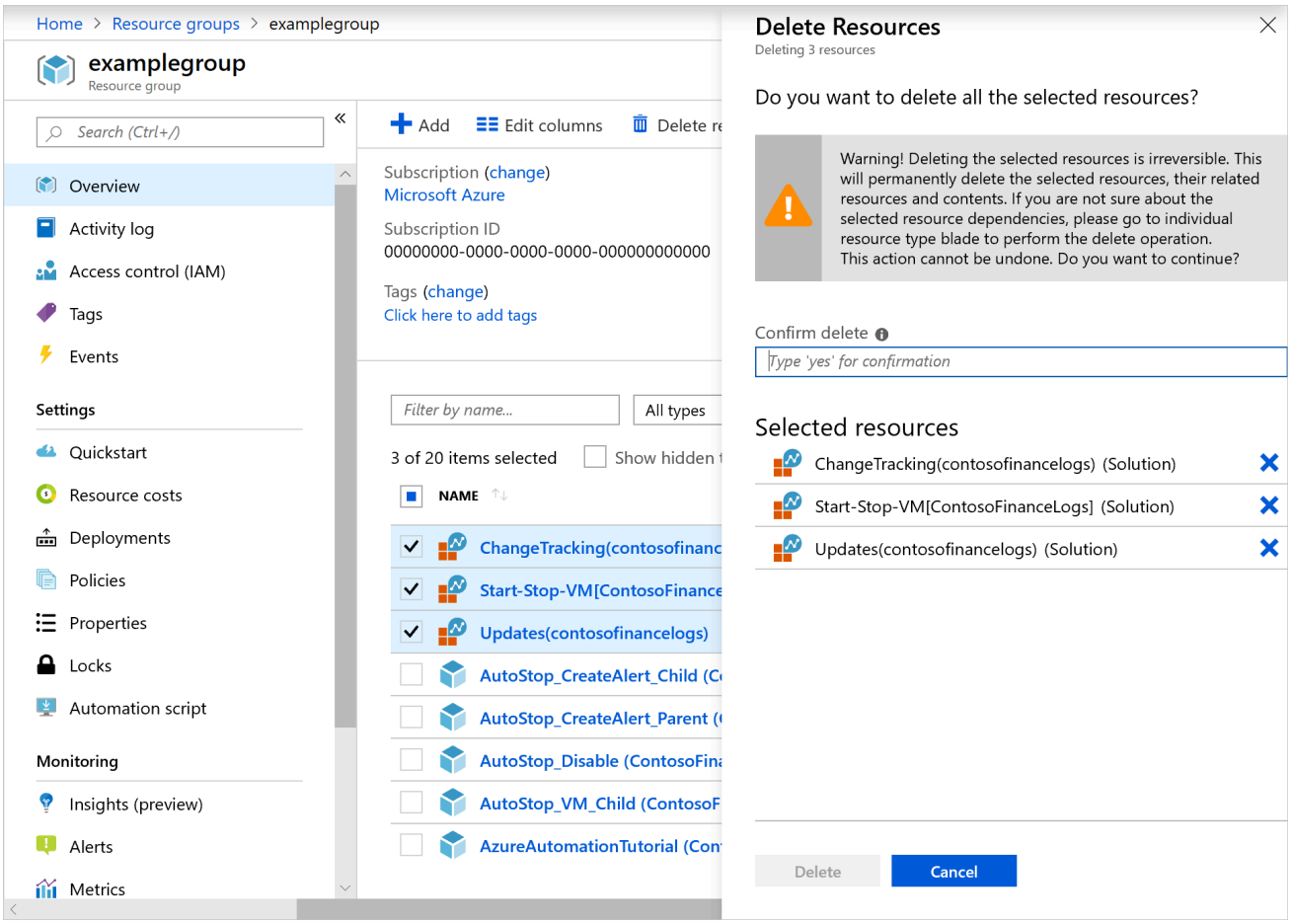
Eliminar características

Para desvincular el área de trabajo de la cuenta de Automation, se deben quitar los recursos de características del área de trabajo:

- Seguimiento de Cambios e Inventario
- Administración de actualizaciones
- Inicio y detención de máquinas virtuales durante las horas no laborables

1. En el portal de Azure, busque el grupo de recursos.

2. Busque cada característica y seleccione **Eliminar** en la página **Eliminar recursos**.



Si lo prefiere, puede eliminar los recursos mediante el cmdlet Remove-AzResource:

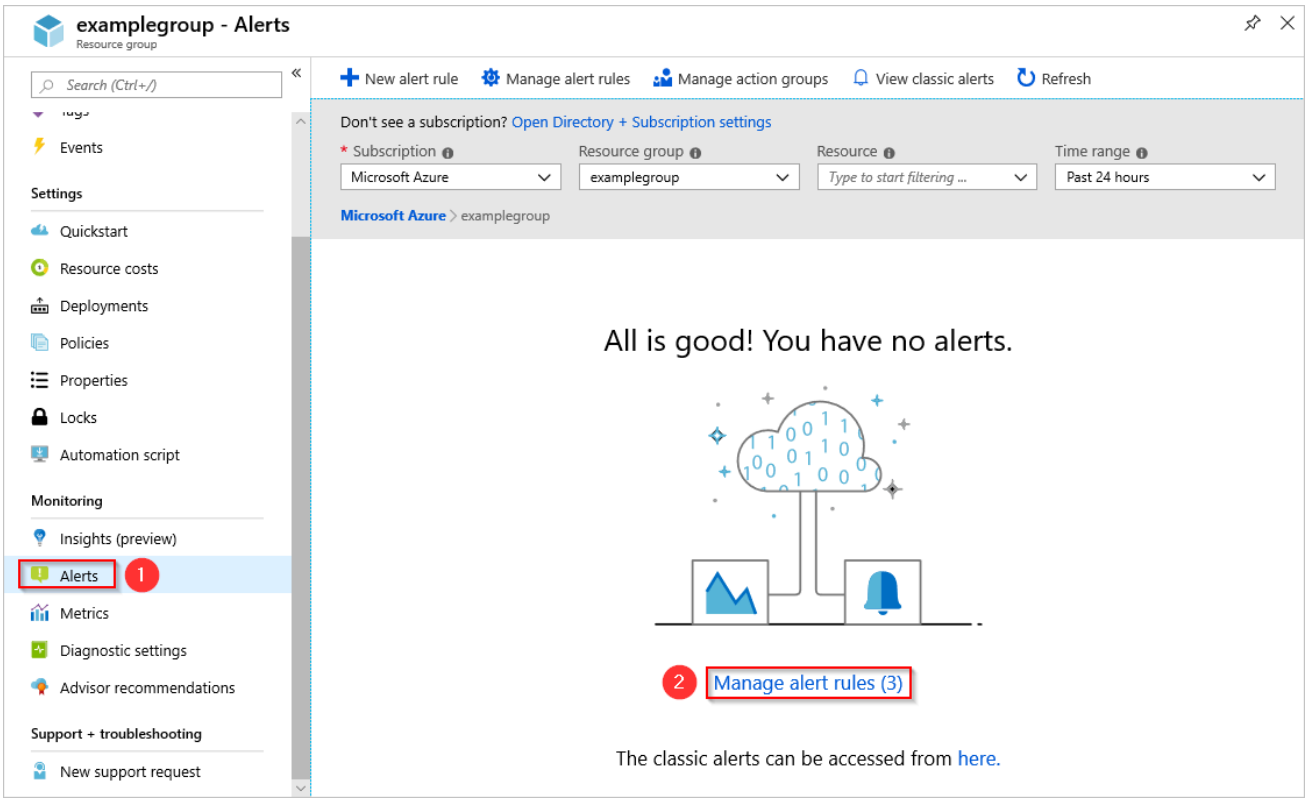
Azure PowerShell

```
$workspaceName = <myWorkspaceName>
$resourceGroupName = <myResourceGroup>
Remove-AzResource -ResourceType 'Microsoft.OperationsManagement/solutions' -
ResourceName "ChangeTracking($workspaceName)" -ResourceGroupName $resourceGroupName
Remove-AzResource -ResourceType 'Microsoft.OperationsManagement/solutions' -
ResourceName "Updates($workspaceName)" -ResourceGroupName $resourceGroupName
Remove-AzResource -ResourceType 'Microsoft.OperationsManagement/solutions' -
ResourceName "Start-Stop-VM($workspaceName)" -ResourceGroupName $resourceGroupName
```

Eliminar las reglas de alerta para el inicio/parada de las máquinas virtuales fuera del horario laboral.

En el caso de la característica Start/Stop VMs during off hours, también debe quitar las reglas de alerta creadas por ella.

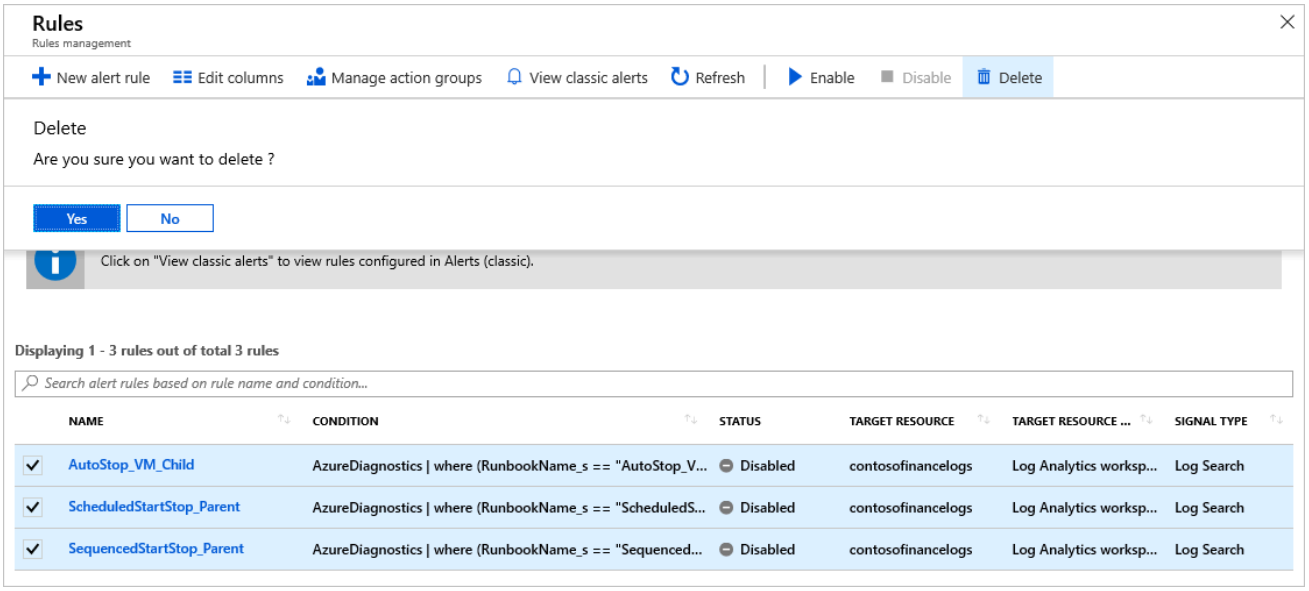
1. En el portal de Azure, vaya al grupo de recursos y seleccione **Monitoring**>**Alerts**>**Administrar reglas de alerta**.



2. En la página Reglas, debería ver una lista de las alertas configuradas en ese grupo de recursos. La característica crea estas reglas:

    - AutoStop_VM_Child
    - ScheduledStartStop_Parent
    - SequencedStartStop_Parent

3. Seleccione las reglas de una en una y haga clic en **Eliminar** para quitarlas.



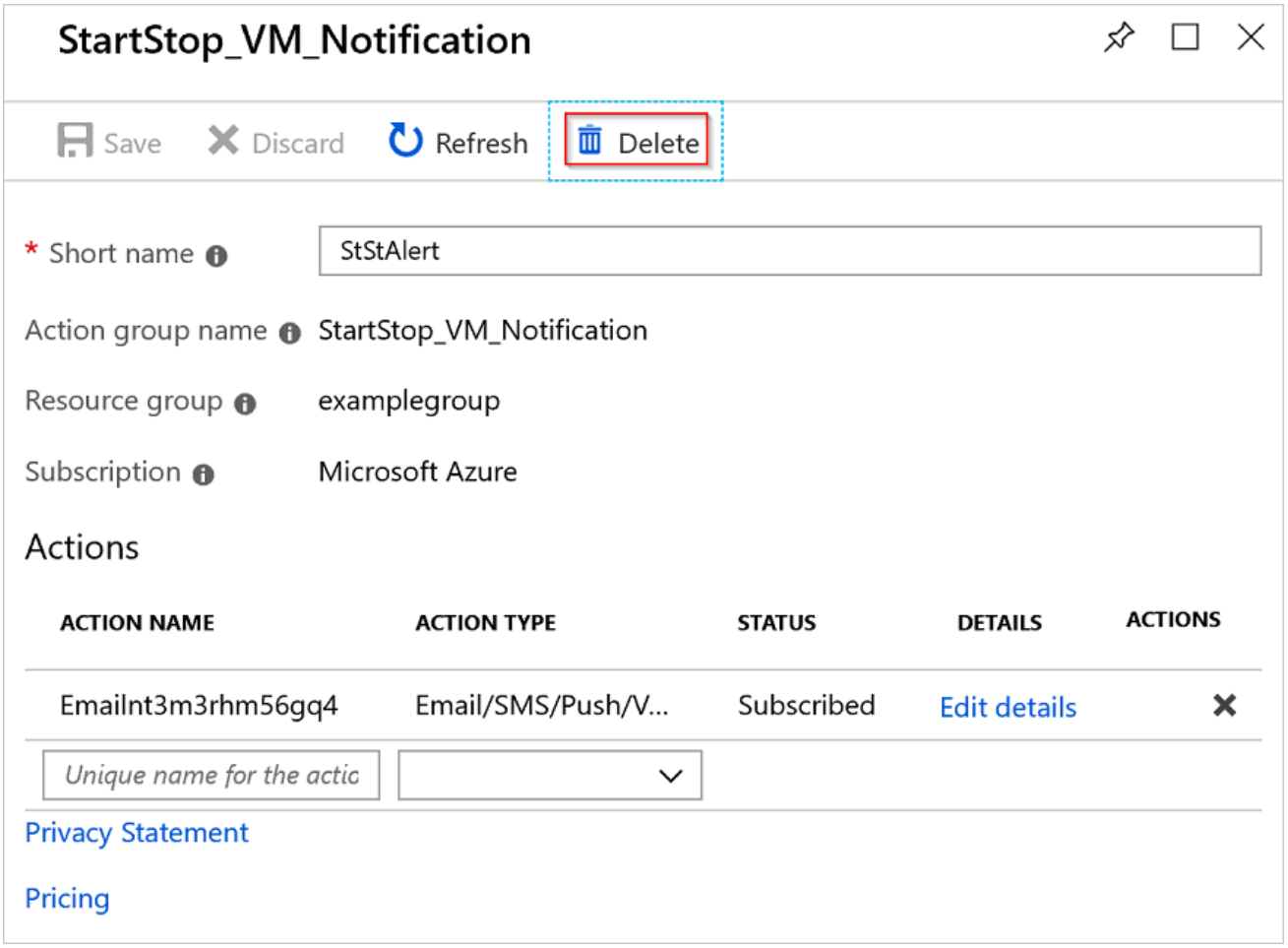
Nota:

> Si no ve ninguna regla de alerta en la página Reglas, cambie el campo **Estado** a **Deshabilitado** para mostrar las alertas deshabilitadas.

4. Cuando quite las reglas de alerta, debe quitar el grupo de acciones creado para las notificaciones de Start/Stop VMs during off hours. En el portal de Azure, seleccione **Monitor**>**Alerts**>**Administrar grupos de acciones**.

5. Seleccione **StartStop_VM_Notification**.

6. En la página del grupo de acciones, seleccione **Eliminar**.



Si lo prefiere, puede eliminar el grupo de acciones mediante el cmdlet Remove-AzActionGroup:

Azure PowerShell

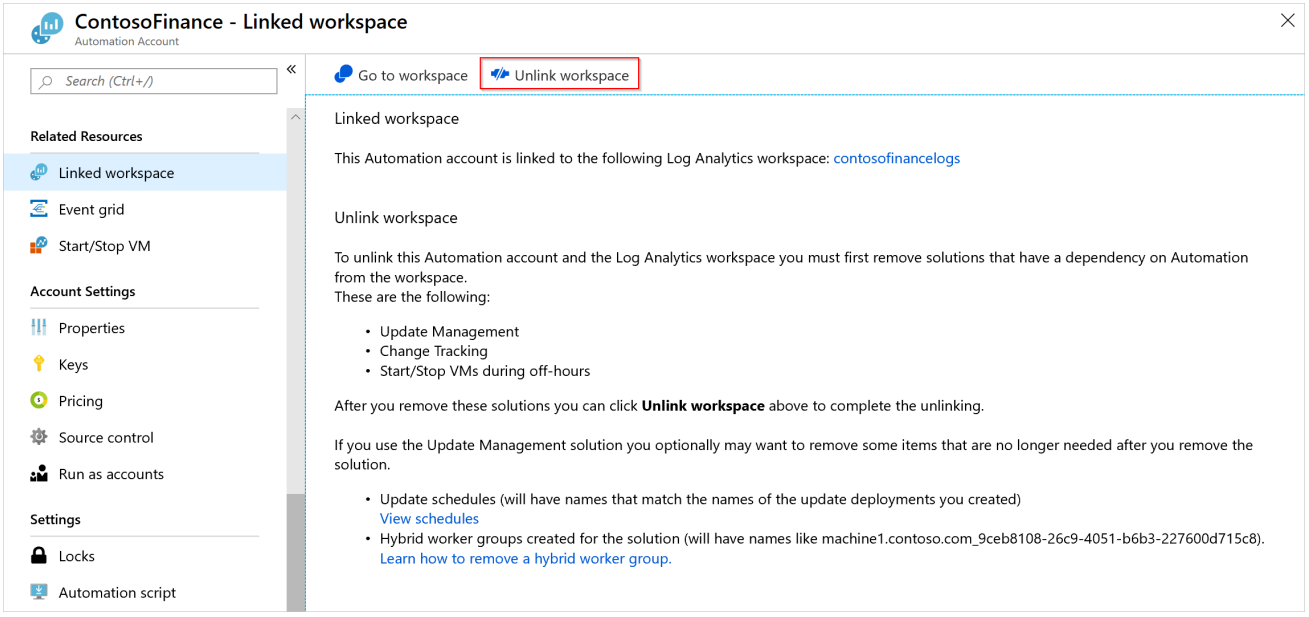
```
Remove-AzActionGroup -ResourceGroupName <myResourceGroup> -Name
StartStop_VM_Notification
```

Desvincula tu área de trabajo

Ahora puede desvincular su área de trabajo:

1. En el portal de Azure, seleccione **Cuenta de automatización** > **Recursos relacionados** > **Espacio de trabajo vinculado**.

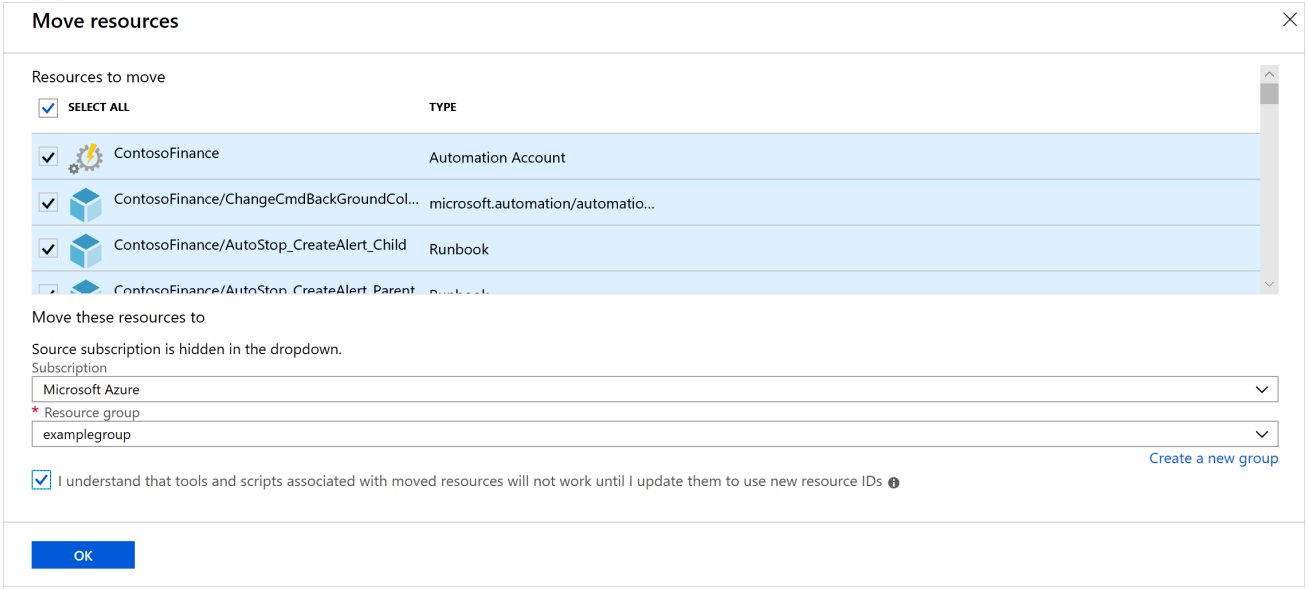
2. Seleccione **Desvincular área de trabajo** para desvincular el área de trabajo de la cuenta de Automation.



Traslado de la cuenta de Automation

Ahora puede mover la cuenta de Automation y sus runbooks.

1. En el portal de Azure, vaya al grupo de recursos de la cuenta de Automation. Seleccione **Mover** > **Mover a otra suscripción**.



2. Seleccione los recursos del grupo de recursos que quiere mover. Asegúrese de incluir su cuenta de Automation, los runbooks y los recursos del área de trabajo de Log Analytics.

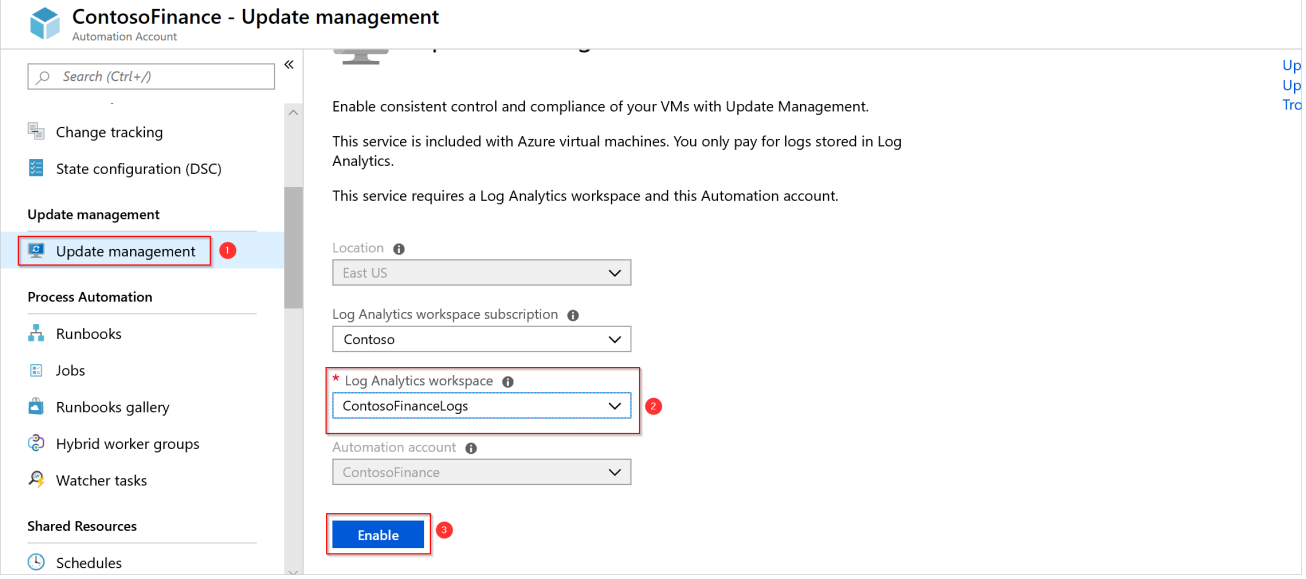
> **Nota:**
>
> El movimiento de la identidad administrada asignada por el sistema y de la identidad administrada asignada por el usuario se realiza automáticamente con la cuenta de Automation.

Habilitación de las características

Debe volver a activar las funciones que quitó antes del traslado:

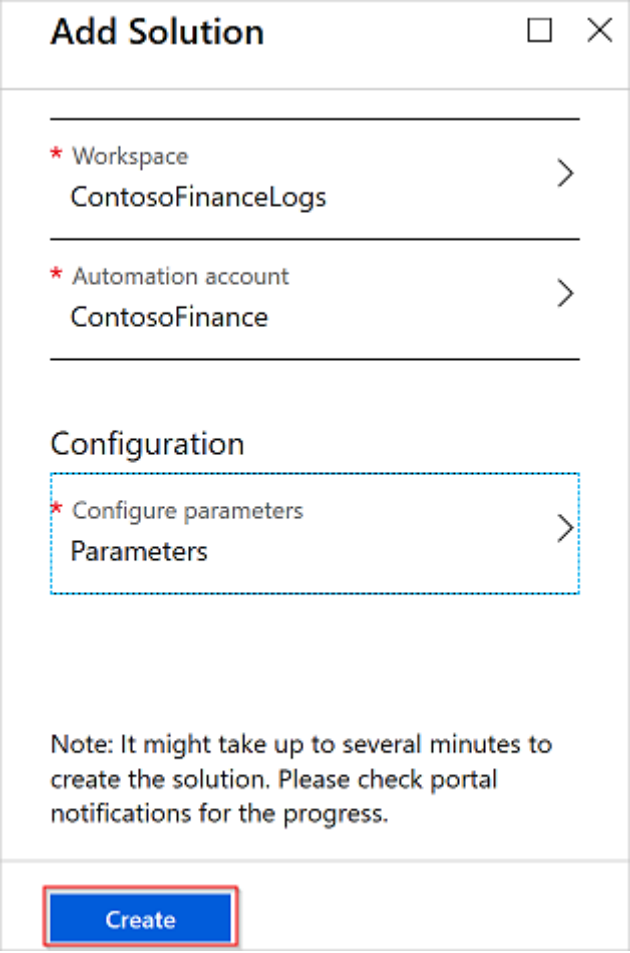
1. Para activar Seguimiento de cambios e inventario, seleccione **Seguimiento de cambios e inventario** en la cuenta de Automation. Elija el área de trabajo de Log Analytics que ha movido y seleccione **Enable**.

2. Repita el paso 1 con Update Management.



3. Las máquinas que tienen tus características habilitadas son visibles cuando se ha conectado el área de trabajo de Log Analytics existente.

4. En la página Agregar solución, elija el área de trabajo de Log Analytics y la cuenta de Automation.



Comprobación del traslado

Una vez completado el traslado, compruebe las que funcionalidades de la lista siguiente estén habilitadas.

Expandir tabla

Capacidad	Pruebas	Solución de problemas
Runbooks	Un libro de ejecución puede ejecutarse y conectarse correctamente a recursos de Azure.	Resolución de problemas de runbooks
Control de código fuente	Puede ejecutar una sincronización manual en el repositorio de control de código fuente.	Integración de control de código fuente
Change Tracking e Inventario	Compruebe que puede ver los datos del inventario actual en las máquinas.	Solución de problemas de seguimiento de cambios
Administración de actualizaciones	Compruebe que ve las máquinas y que están en buen estado. Ejecute una implementación de actualización de software de prueba.	Solución de problemas de gestión de actualizaciones
Recursos compartidos	Compruebe que ve todos los recursos	

Capacidad	Pruebas	Solución de problemas
	compartidos, como credenciales y variables.	

Pasos siguientes

- Para obtener información sobre cómo mover Automation a una nueva región, consulte Traslado de la cuenta de Automation a otra región.
- Para obtener información sobre cómo mover recursos en Azure, consulte Mover recursos en Azure.
- Para solucionar problemas de runbook al mover la cuenta de Azure Automation a otra suscripción, consulte Solucionar problemas de runbook.

Last updated on 20/04/2026

Reubicación de Azure Automation en otra región

15/09/2025

En este artículo se tratan las instrucciones de reubicación para reubicar Azure Automation entre regiones.

Hay varias razones por las que sería conveniente mover los recursos de Azure existentes de una región a otra. Puede que desee:

- Aproveche las ventajas de una nueva región de Azure.
- Implemente características o servicios que solo están disponibles en regiones determinadas.
- Cumpla con los requisitos internos de política y gobernanza.
- Alineación con fusiones y adquisiciones de la empresa
- Cumpla los requisitos de planeamiento de capacidad.

Si la instancia de Azure Automation no tiene ninguna configuración y la propia instancia debe moverse sola, puede optar por volver a implementar la instancia de NetApp File mediante Bicep, plantilla de ARM o Terraform.

Prerrequisitos

- Identifique todos los recursos dependientes de Automation.
- Si la identidad administrada asignada por el sistema no se usa en el origen, debe asignar la identidad administrada asignada por el usuario en el destino.
- Si el destino de Azure Automation debe habilitarse para el acceso privado, asócielo a la red virtual para el punto de conexión privado.
- Si la instancia de Azure Automation de origen está habilitada con una conexión privada, cree un vínculo privado y configure el vínculo privado con el sistema de nombres de dominio (DNS) en el destino.
- Para que Azure Automation se comunique con Hybrid RunBook Worker, Administrador de actualizaciones de Azure, Change Tracking, Configuración de inventario y Automation State Configuration, debe habilitar el puerto 443 para el acceso entrante y saliente a Internet.

Tiempo de inactividad

Para comprender los posibles tiempos de inactividad que podrían producirse, consulte Cloud Adoption Framework para Azure: Seleccionar un método de reubicación.

Preparación

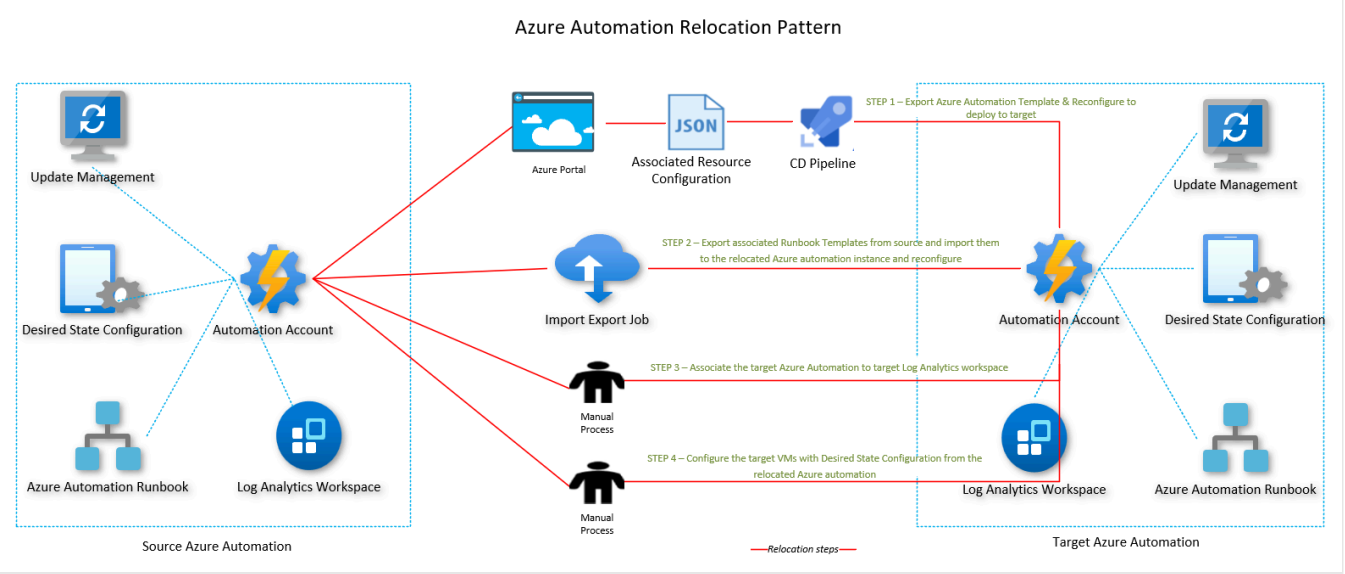
Para empezar, exporte una plantilla de Resource Manager. Esta plantilla contiene la configuración que describe el espacio de nombres de Automation.

1. Inicie sesión en Azure Portal.
2. Seleccione **Todos los recursos** y, después, seleccione el recurso de Automation.
3. Seleccione **Exportar plantilla**.
4. Elija **Descargar** en la página **Exportar plantilla**.
5. Busque el archivo ZIP que descargó desde el portal y descomprímalo en la carpeta que prefiera.

   Este archivo ZIP contiene los archivos .json que incluyen la plantilla y los scripts para implementar la plantilla.

Volver a implementar

En el diagrama siguiente, las líneas de flujo rojas muestran la reimplementación de la instancia de destino junto con el movimiento de configuración.



Para implementar la plantilla para crear una instancia de Automation en la región de destino:

1. Reconfigure los parámetros de la plantilla para el objetivo.
2. Implemente la plantilla mediante Azure Resource Manager, Azure Portal o Azure PowerShell.

3. Utilice PowerShell para exportar todos los runbooks asociados desde la instancia de Azure Automation de origen e importarlos a la instancia de destino. Vuelva a configurar las propiedades según el objetivo. Para más información, consulte Export-AzAutomationRunbook.
4. Asocie la instancia de Azure Automation reubicada al área de trabajo de Log Analytics de destino.
5. Configure las máquinas virtuales de destino con la configuración de estado deseada de la instancia de Azure Automation reubicada según el origen.

Pasos siguientes

Para obtener más información sobre cómo trasladar recursos entre regiones y la recuperación ante desastres en Azure, consulte:

- Traslado de los recursos a un nuevo grupo de recursos o a una nueva suscripción
- Traslado de máquinas virtuales de Azure a otra región

Administración de una cuenta de Azure Automation

12/05/2025

Después de habilitar una cuenta de Azure Automation para ayudarle a automatizar el proceso de TI o de negocio, o habilitar sus otras características para que admitan la administración de operaciones de las máquinas de Azure y que no son de Azure, puede decidir dejar de usar la cuenta de Automation. Si ha habilitado características que dependen de la integración con un área de trabajo de Log Analytics de Azure Monitor, es necesario realizar más pasos para completar esta acción.

En este artículo se explica no solo cómo quitar completamente una cuenta de Automation mediante Azure Portal, Azure PowerShell, la CLI de Azure o la API REST, sino también cómo se restaura una cuenta de Azure Automation eliminada.

Eliminación de la cuenta de Azure Automation

La eliminación de la cuenta de Automation puede realizarse mediante uno de los métodos siguientes, en función de los modelos de implementación admitidos:

- Elimine el grupo de recursos que contiene la cuenta de Automation.
- Elimine el grupo de recursos que contiene la cuenta de Automation y el área de trabajo de Log Analytics de Azure Monitor vinculada, en los siguientes casos:
  - La cuenta y el área de trabajo están dedicadas a las características Update Management, Seguimiento de cambios e inventario o Start/Stop VMs during off-hours.
  - La cuenta está dedicada a la automatización de los procesos y se integra con un área de trabajo para enviar el estado y las transmisiones de trabajos del runbook.
- Desvincule el área de trabajo de Log Analytics de la cuenta de Automation y elimine la cuenta de Automation.
- Elimine la característica del área de trabajo vinculada, desvincule la cuenta del área de trabajo y, luego, elimine la cuenta de Automation.

Requisito previo

Compruebe que no haya ningún bloqueo de Resource Manager aplicado a la suscripción, el grupo de recursos o el recurso que impida la eliminación o la modificación accidentales de recursos críticos. Si ha implementado la solución Start/Stop VMs during off-hours, se establece el nivel de bloqueo en **CanNotDelete** en varios recursos dependientes de la cuenta de

Automation (en concreto, sus runbooks y variables). Quite los bloqueos antes de eliminar la cuenta de Automation.

> **Nota**
>
> Si recibe un mensaje de error similar a "`The link cannot be updated or deleted because it is linked to Update Management and/or ChangeTracking Solutions`", la cuenta de Automation se vincula a un área de trabajo de Log Analytics con las características Update Management o Seguimiento de cambios e inventario habilitadas. Para obtener más información, consulte Eliminación de una cuenta de Automation de capacidad compartida a continuación.

Eliminación del grupo de recursos dedicado

Para eliminar la cuenta de Automation (y también el área de trabajo de Log Analytics si está vinculada a ella) creada en el mismo grupo de recursos dedicado a la cuenta, siga los pasos descritos en el artículo Eliminación de grupos de recursos y recursos en Azure Resource Manager.

Eliminación de una cuenta de Automation independiente

Si la cuenta de Automation no está vinculada a un área de trabajo de Log Analytics, realice los pasos siguientes para eliminarla.

Azure Portal

1. Inicie sesión en Azure en https://portal.azure.com.
2. En Azure Portal, vaya a las **cuentas de Automation**.
3. Abra su cuenta de Automation y seleccione **Eliminar** en el menú.

Mientras se comprueba la información y se elimina la cuenta, puede hacer un seguimiento del progreso en **Notificaciones**, una opción del menú.

Eliminación de una cuenta de Automation independiente vinculada al área de trabajo

Si la cuenta de Automation está vinculada a un área de trabajo de Log Analytics para recopilar transmisiones y registros de trabajos, realice los pasos siguientes para eliminarla.

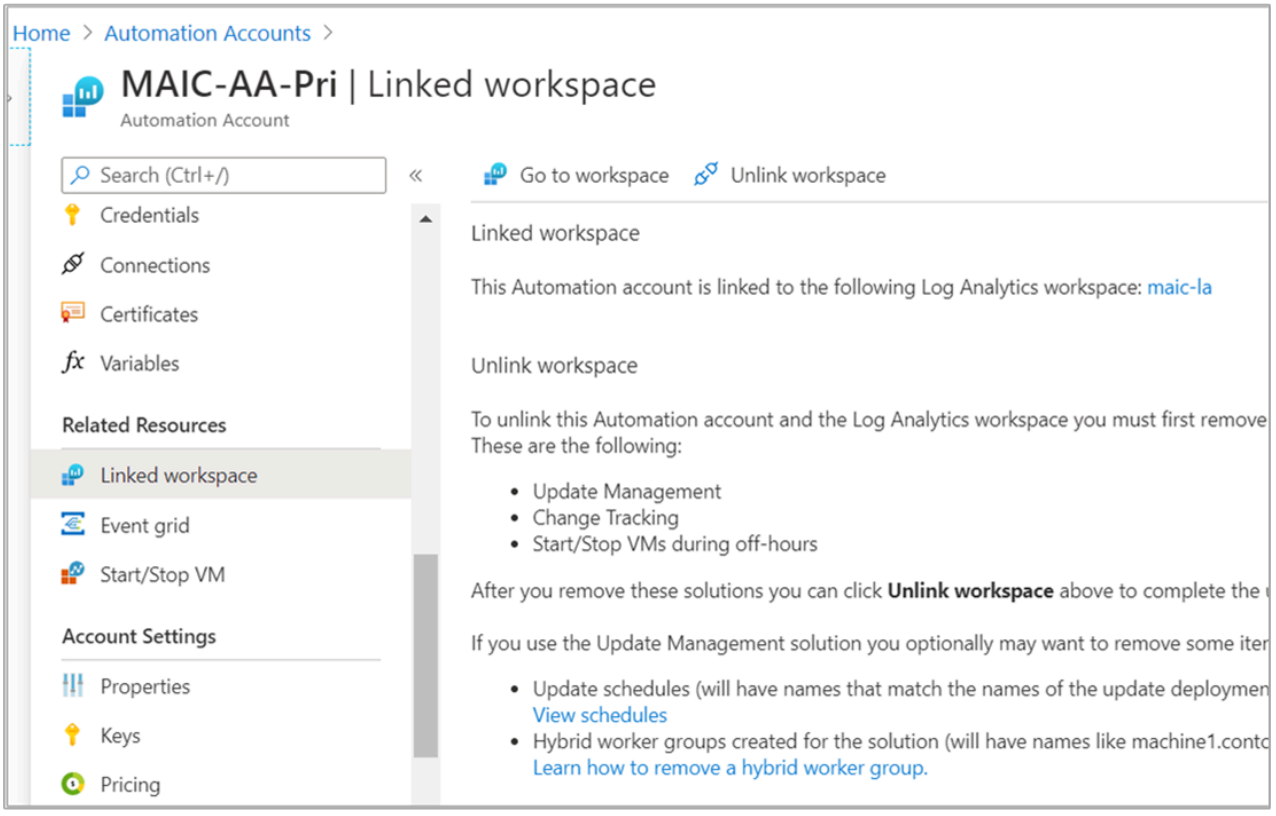
Hay dos opciones para desvincular el área de trabajo de Log Analytics de la cuenta de Automation. Puede realizar este proceso desde la cuenta de Automation o desde el área de trabajo vinculada.

Para desvincularla desde la cuenta de Automation, realice los pasos siguientes.

1. En Azure Portal, vaya a las **cuentas de Automation**.

2. Abra la cuenta de Automation y seleccione **Área de trabajo vinculada** en **Recursos relacionados** a la izquierda.

3. En la página **Desvincular área de trabajo**, seleccione **Desvincular área de trabajo** y responda a los avisos.



   Mientras intenta desvincular el área de trabajo de Log Analytics, puede realizar el seguimiento del progreso en la opción **Notificaciones** del menú.

Para desvincularla desde el área de trabajo, realice los pasos siguientes.

1. En Azure Portal, vaya a **Areas de trabajo de Log Analytics**.

2. En el área de trabajo, seleccione **Cuenta de Automation** en **Recursos relacionados**.
3. En la página Cuenta de Automation, seleccione **Desvincular cuenta** y responda a los avisos.

Mientras intenta desvincular la cuenta de Automation, puede seguir el progreso en **Notificaciones** desde el menú.

Después de que la cuenta de Automation se haya desvinculado correctamente del área de trabajo, siga los pasos de la sección Cuenta de Automation independiente para eliminar la cuenta.

Es posible que se produzca un error en la eliminación de la cuenta de Automation después de desvincular correctamente el área de trabajo de Log Analytics. Vuelva a intentar la eliminación después de 24 horas.

> **Nota**
>
> Un área de trabajo de Log Analytics eliminada previamente no se puede desvincular correctamente de la cuenta de Automation. En este escenario, primero recupere y elimínela de forma permanente para eliminar el vínculo definitivamente. Aprenda **cómo recuperar un área de trabajo en un estado de eliminación temporal** y **eliminarla de forma permanente**.

Eliminación de una cuenta de Automation de capacidad compartida

Para eliminar la cuenta de Automation vinculada a un área de trabajo de Log Analytics para Update Management, Seguimiento de cambios e inventario o Start/Stop VMs during off-hours, realice los pasos siguientes.

Paso 1: Eliminación de la solución del área de trabajo vinculada

Azure Portal

1. Inicie sesión en Azure en https://portal.azure.com .
2. Vaya a la cuenta de Automation y seleccione **Área de trabajo vinculada**.

3. En **Recursos relacionados**, seleccione **Área de trabajo vinculada** y, a continuación, seleccione **Ir al área de trabajo**.

4. En **Clásico**, seleccione **Soluciones heredadas**.

5. En la página Soluciones, seleccione una de las siguientes opciones en función de las características implementadas en la cuenta:

    - En Start/Stop VMs during off-hours, seleccione **Start-Stop-VM [nombre del área de trabajo]**.
    - En Update Management, seleccione **Actualizaciones (nombre del área de trabajo)**.
    - En Seguimiento de cambios e inventario, seleccione **ChangeTracking (nombre del área de trabajo)**.

6. En la página **Solución**, seleccione **Eliminar** en el menú. Si hay implementada más de una de las características enumeradas anteriormente en la cuenta de Automation y el área de trabajo vinculada, deberá seleccionar y eliminar cada una de ellas antes de continuar.

7. Mientras se comprueba la información y se elimina la característica, puede hacer un seguimiento del progreso en **Notificaciones**, una opción del menú. Después del proceso de eliminación, se le devuelve a la página Soluciones.

Paso 2: Desvinculación de un área de trabajo de una cuenta de Automation

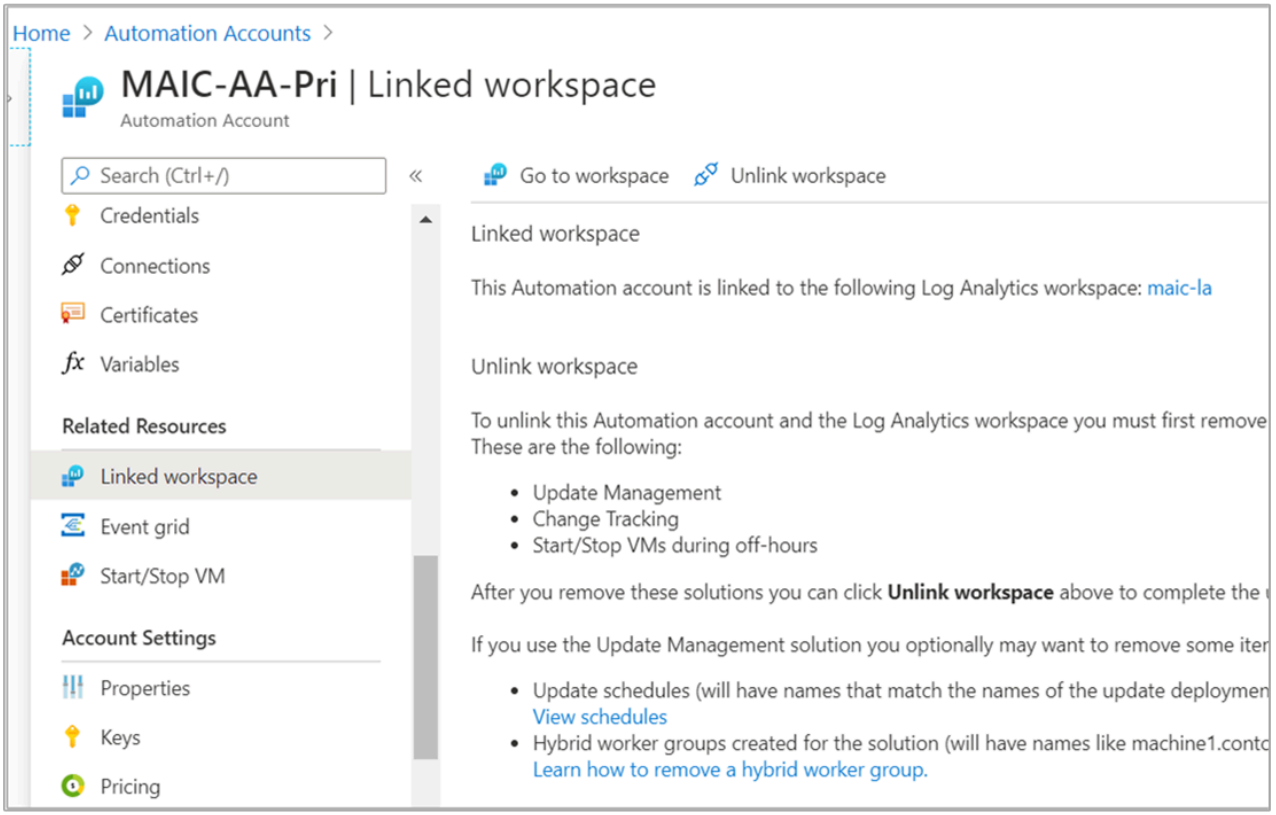
Hay dos opciones para desvincular el área de trabajo de Log Analytics de la cuenta de Automation. Puede realizar este proceso desde la cuenta de Automation o desde el área de trabajo vinculada.

Para desvincularla desde la cuenta de Automation, realice los pasos siguientes.

1. En Azure Portal, vaya a las **cuentas de Automation**.

2. Abra la cuenta de Automation y seleccione **Área de trabajo vinculada** en **Recursos relacionados** a la izquierda.

3. En la página **Desvincular área de trabajo**, seleccione **Desvincular área de trabajo** y responda a los avisos.



Mientras intenta desvincular el área de trabajo de Log Analytics, puede realizar el seguimiento del progreso en la opción **Notificaciones** del menú.

Para desvincularla desde el área de trabajo, realice los pasos siguientes.

1. En Azure Portal, vaya a **Areas de trabajo de Log Analytics**.
2. En el área de trabajo, seleccione **Cuenta de Automation** en **Recursos relacionados**.
3. En la página Cuenta de Automation, seleccione **Desvincular cuenta** y responda a los avisos.

Mientras intenta desvincular la cuenta de Automation, puede seguir el progreso en **Notificaciones** desde el menú.

Paso 3: Eliminación de una cuenta de Automation

Después de que la cuenta de Automation se haya desvinculado correctamente del área de trabajo, siga los pasos de la sección Cuenta de Automation independiente para eliminar la cuenta.

Restauración de las cuentas de Automation eliminadas

Las cuentas de Automation eliminadas se pueden recuperar desde Azure Portal.

Para recuperar una cuenta de Automation, asegúrese de que se cumplen las condiciones siguientes:

- Ha creado la cuenta de Automation con el modelo de implementación de Azure Resource Manager y la ha eliminado en los últimos 30 días.
- Antes de intentar recuperar una cuenta de Azure Automation, asegúrese de que el grupo de recursos de esa cuenta existe.

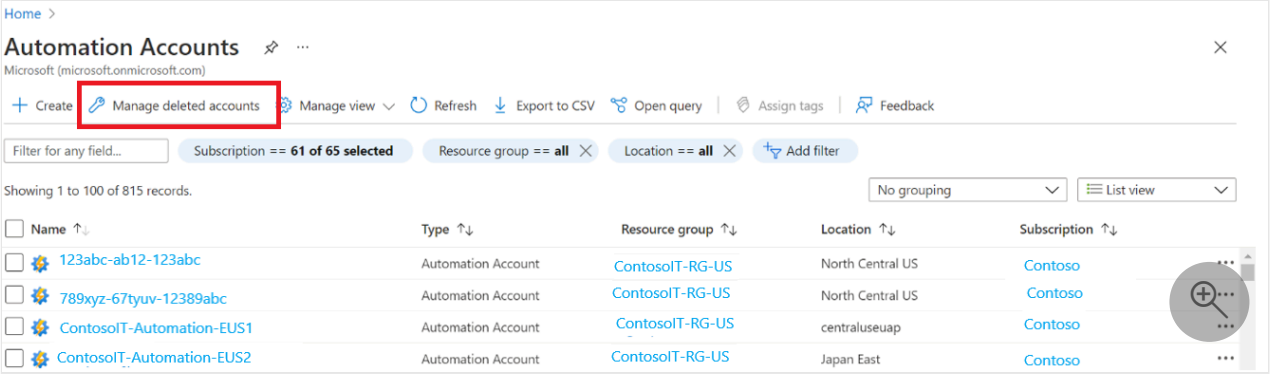
> **Nota**
>
> - Si se elimina el grupo de recursos de la cuenta de Automation, para recuperarlo, se debe volver a crear el grupo de recursos con el mismo nombre.
> - Aunque el grupo de recursos no está presente, puede ver la cuenta de Automation en la lista de eliminados. Si el grupo de recursos no está presente, se produce un error en la operación de restauración de la cuenta con el error *Restauración de la cuenta fallida*.

Recuperación de cuentas de Automation eliminadas

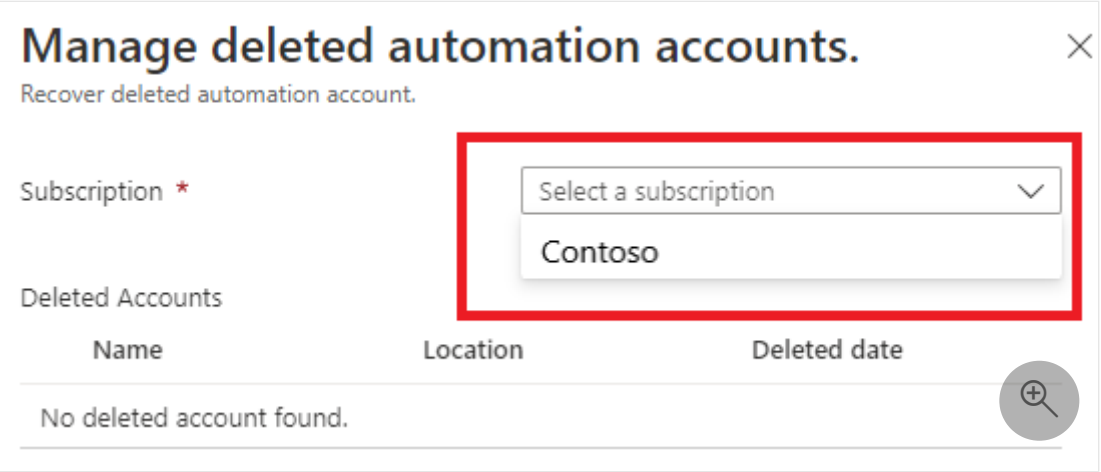
1. Inicie sesión en Azure Portal.

2. Vaya a su cuenta de Automation.

3. En la página **Cuentas de Automation**, seleccione **Administrar cuentas eliminadas**.

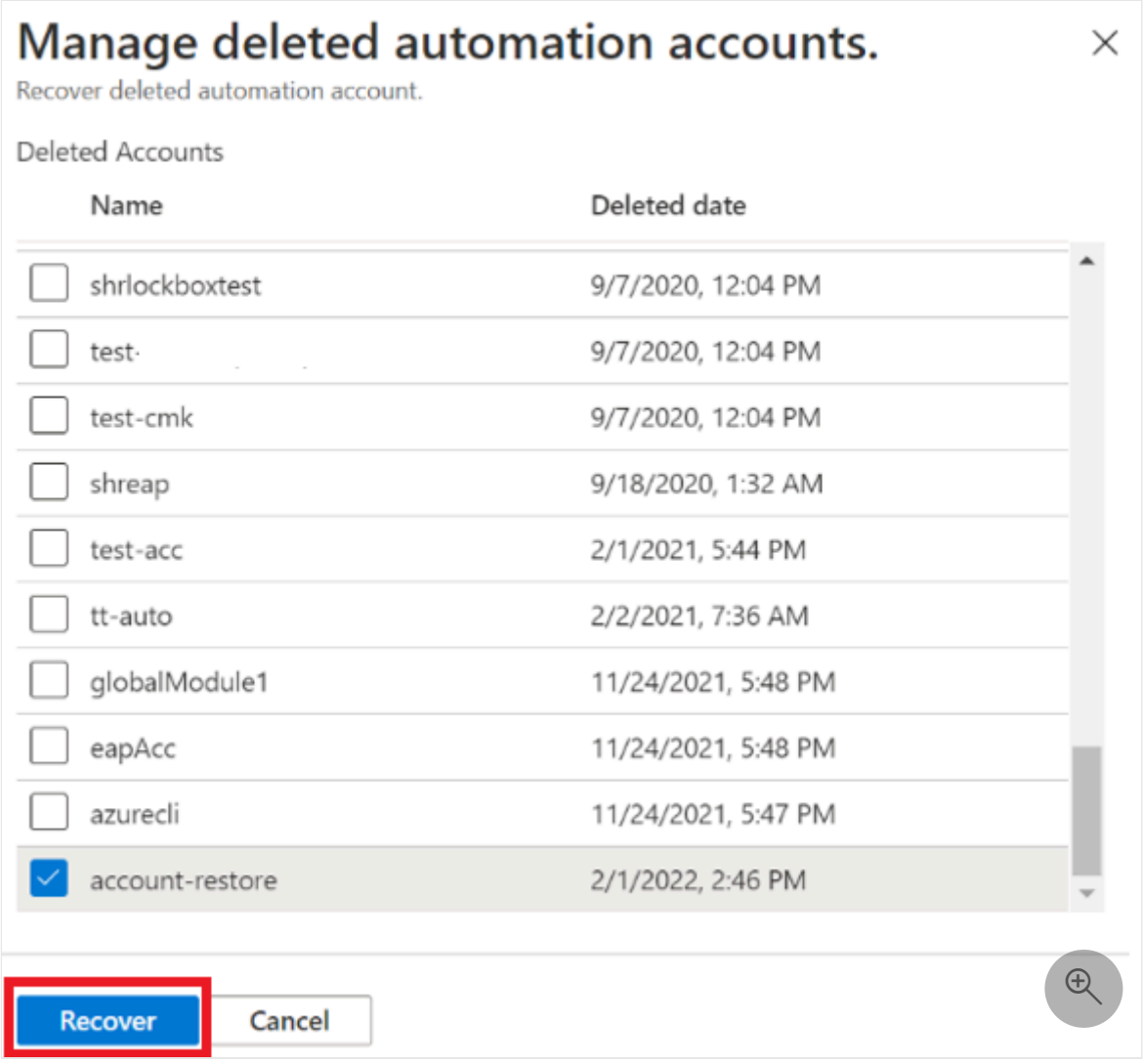


4. En el panel **Manage deleted automation accounts** (Administrar cuentas de Automatización eliminadas), seleccione **Subscription** (Suscripción) en la lista desplegable.

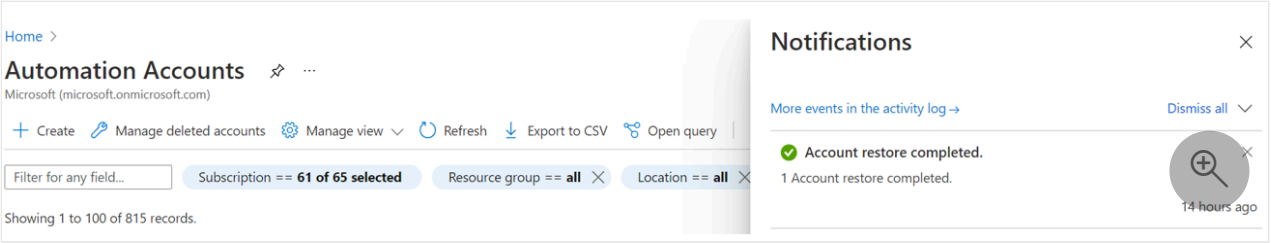


Se muestra la lista de cuentas eliminadas de esa suscripción.

5. Active la casilla de las cuentas que desea restaurar y haga clic en **Recover** (Recuperar).



Aparece una notificación que confirma que se ha restaurado la cuenta.



Pasos siguientes

Para crear una cuenta de Automation desde Azure Portal, consulte Creación de una cuenta independiente de Azure Automation. Si prefiere crear su cuenta mediante una plantilla, consulte Creación de una cuenta de Automation con una plantilla de Azure Resource Manager.

Migración de Orchestrator a Azure Automation (versión beta)

Los runbooks de System Center 2012: Orchestrator se basan en actividades provenientes de paquetes de integración escritos específicamente para Orchestrator, mientras que los runbooks de Azure Automation se basan en Windows PowerShell. Runbooks gráficos de Azure Automation tienen una apariencia similar a los runbooks de Orchestrator, donde sus actividades representan cmdlets de PowerShell, runbooks secundarios y recursos. Además de convertir los runbooks, debe convertir los paquetes de integración con las actividades que los runbooks usan en módulos de integración con cmdlets de Windows PowerShell.

Service Management Automation (SMA) almacena y ejecuta runbooks en el centro de datos local como Orchestrator y usa los mismos módulos de integración que Azure Automation. El Convertidor de runbooks convierte los runbooks de Orchestrator en runbooks gráficos, que no son compatibles con SMA. Aún así puede instalar el módulo Actividades estándar y los módulos de integración de System Center Orchestrator en SMA, pero tendrá que reescribir los runbooks manualmente.

Descarga del kit de herramientas de migración de Orchestrator

El primer paso de la migración consiste en descargar el kit de herramientas para migración de System Center Orchestrator. Este kit incluye herramientas que le ayudan a convertir runbooks de Orchestrator a Azure Automation.

Importación del módulo Actividades estándar

Importe el módulo Actividades estándar a Azure Automation. Esto incluye versiones convertidas de las actividades estándar de Orchestrator que los runbooks gráficos convertidos pueden usar.

Importación de los módulos de integración de System Center Orchestrator

Microsoft proporciona paquetes de integración para crear runbooks a fin de automatizar componentes de System Center y otros productos. Algunos de estos paquetes de integración se basan en OIT pero actualmente no se pueden convertir en módulos de integración debido a problemas conocidos. Importe los Módulos de integración de System Center Orchestrator a

Azure Automation correspondientes a los paquetes de integración que usan los runbooks que acceden a System Center. Este paquete incluye las versiones convertidas de los paquetes de integración que se pueden importar a Azure Automation y a Service Management Automation.

Conversión de los paquetes de integración

Use el Convertidor de paquetes de integración para convertir los paquetes de integración creados con Orchestrator Integration Toolkit (OIT) en módulos de integración basados en PowerShell que se pueden importar a Azure Automation o a Service Management Automation. Al ejecutar el Convertidor de paquetes de integración, aparece un asistente que le permite seleccionar un archivo de paquete de integración (.oip). Después, el asistente muestra las actividades incluidas en ese paquete de integración y le permite seleccionar las que quiere migrar. Al completar el asistente, se crea un módulo de integración que incluye un cmdlet correspondiente para cada una de las actividades del paquete de integración original.

> **Nota:**
>
> No puede usar el Convertidor de paquetes de integración para convertir paquetes de integración que no se han creado con OIT. También hay algunos módulos de integración ofrecidos por Microsoft que actualmente no se pueden convertir con esta herramienta. Las versiones convertidas de estos paquetes de integración se ofrecen para descarga a fin de que se puedan instalar en Azure Automation o Service Management Automation.

Parámetros

Las propiedades de una actividad en el paquete de integración se convierten en parámetros del cmdlet correspondiente en el módulo de integración. Los cmdlets de Windows PowerShell tienen un conjunto de parámetros comunes que se pueden usar con todos los cmdlets. Por ejemplo, el parámetro -Verbose hace que un cmdlet genere información detallada acerca de su operación. Ningún cmdlet puede tener un parámetro con el mismo nombre que un parámetro común. Si una actividad sí tiene una propiedad con el mismo nombre que un parámetro común, el asistente le solicita que cambie el nombre del parámetro.

Actividad de supervisión

Los runbooks de supervisión en Orchestrator comienzan con una actividad de supervisión y se ejecutan de manera constante, mientras esperan que un evento determinado los invoque. Azure Automation no admite los runbooks de supervisión, por lo que no se convierte ninguna actividad de supervisión del paquete de integración. En su lugar, se crea un cmdlet de

marcador de posición en el módulo de integración para la actividad de supervisión. Este cmdlet no tiene ninguna funcionalidad, sino que permite la instalación de cualquier runbook convertir que lo use. Este runbook no se puede ejecutar en Azure Automation, pero sí se puede instalar para que el usuario pueda modificarlo.

Orchestrator incluye un conjunto de actividades estándar que no están incluidas en un paquete de integración, pero que son usadas por muchos runbooks. El módulo Actividades estándar es un módulo de integración que incluye un cmdlet equivalente para cada una de estas actividades. Debe instalar este módulo de integración en Azure Automation antes de importar cualquier runbook convertido que usa una actividad estándar.

Además de admitir los runbooks convertidos, es posible que un usuario familiarizado con Orchestrator pueda usar los cmdlets del módulo de actividades estándar para crear nuevos runbooks en Azure Automation. A pesar de que la funcionalidad de todas las actividades estándar se puede realizar con cmdlets, es posible que funcionen de manera diferente. Los cmdlets del módulo Actividades estándar convertidas funcionan de la misma manera que sus actividades correspondientes y usan los mismos parámetros. Esto puede ayudarle a realizar la transición a runbooks de Azure Automation.

Conversión de runbooks de Orchestrator

El Convertidor de runbooks de Orchestrator convierte los runbooks de Orchestrator en runbooks gráficos que se pueden importar a Azure Automation. El Convertidor de runbooks se implementa como módulo de PowerShell con el cmdlet `ConvertFrom-SCORunbook` que realiza la conversión. Al instalarlo, el convertidor crea un acceso directo a una sesión de PowerShell que carga el cmdlet.

Estos son los pasos básicos para convertir un runbook e importarlo a Azure Automation. Más adelante en esta sección se ofrecen detalles sobre el uso del cmdlet.

1. Exporte uno o varios runbooks desde Orchestrator.
2. Obtenga los módulos de integración para todas las actividades del runbook.
3. Convierta los runbooks de Orchestrator en el archivo exportado.
4. Revise la información de los registros para validar la conversión y determinar las tareas manuales necesarias.
5. Importe los runbooks convertidos a Azure Automation.
6. Cree todos los recursos necesarios en Azure Automation.
7. Edite el runbook en Azure Automation para modificar las actividades necesarias.

La sintaxis de `ConvertFrom-SCORunbook` es:

PowerShell

```
ConvertFrom-SCORunbook -RunbookPath <string> -Module <string[]> -OutputFolder
<string>
```

- RunbookPath: ruta de acceso al archivo de exportación que contiene los runbooks que se convertirá.
- Module: lista delimitada por comas de los módulos de integración que contienen las actividades de los runbooks.
- OutputFolder: ruta de acceso a la carpeta para crear runbooks gráficos convertidos.

El siguiente comando de ejemplo convierte los runbooks en un archivo de exportación llamado **MyRunbooks.ois_export**. Estos runbooks usan los paquetes de integración de Active Directory y Data Protection Manager.

PowerShell

```
ConvertFrom-SCORunbook -RunbookPath "c:\runbooks\MyRunbooks.ois_export" -Module
c:\ip\SystemCenter_IntegrationModule_ActiveDirectory.zip,c:\ip\SystemCenter_Integrat
ionModule_DPM.zip -OutputFolder "c:\runbooks"
```

Uso de archivos de registro del Convertidor de runbooks

El Convertidor de runbooks crea los siguientes archivos de registro en la misma ubicación que el runbook convertido. Si los archivos ya existen, se sobrescriben con la información de la última conversión.

Expandir tabla

Archivo	Contenido
Runbook Converter - Progress.log	Pasos detallados de la conversión, incluidas información de cada actividad convertida correctamente y advertencias para cada actividad no convertida.
Runbook Converter - Summary.log	Resumen de la última conversión, incluidas todas las advertencias y realizar tareas de seguimiento que debe realizar, como la creación de una variable necesaria para el runbook convertido.

Exportación de runbooks desde Orchestrator

El Convertidor de runbooks funciona con un archivo de exportación de Orchestrator que contiene uno o más runbooks. Crea un runbook de Azure Automation correspondiente para cada runbook de Orchestrator en el archivo de exportación.

Para exportar un runbook de Orchestrator, haga clic con el botón derecho en el nombre del runbook en Runbook Designer y seleccione **Exportar**. Para exportar todos los runbooks a una

carpeta, haga clic con el botón derecho en el nombre de la carpeta y seleccione **Exportar**.

Conversión de actividades de runbook

El Convertidor de runbooks convierte cada actividad del runbook de Orchestrator a una actividad correspondiente en Azure Automation. Para las actividades que no se puede convertir, se crea una actividad de marcador de posición en el runbook con el texto de la advertencia. Después de importar el runbook convertido a Azure Automation, debe reemplazar cualquiera de estas actividades por actividades válidas que realizan la funcionalidad requerida.

Las actividades de Orchestrator en el módulo Actividades estándar se convierten. Sin embargo, hay algunas actividades de Orchestrator estándar que no están en este módulo y no se convierten. Por ejemplo, `Send Platform Event` no tiene ningún equivalente de Azure Automation, ya que es un evento específico de Orchestrator.

Actividad de supervisión no se convierten porque no hay ningún equivalente para ellas en Azure Automation. La excepción son las actividades de supervisión de paquetes de integración convertidos, que se convierten en la actividad de marcador de posición.

Todas las actividades de un paquete de integración convertido se convertirán si proporciona la ruta de acceso al módulo de integración con el parámetro `modules`. En los paquetes de integración de System Center, puede usar los Módulos de integración de System Center Orchestrator.

Administración de recursos de Orchestrator

El Convertidor de runbooks solo convierte runbooks, no otros recursos de Orchestrator como contadores, variables o conexiones. Los contadores no se admiten en Azure Automation. Las variables y conexiones se admiten, pero es necesario crearlas manualmente. Los archivos de registro le indican si el runbook requiere estos recursos y especifican los recursos correspondientes que se deben crear en Azure Automation para que el runbook convertido funcione correctamente.

Por ejemplo, un runbook puede usar una variable para rellenar un determinado valor en una actividad. El runbook convertido convierte la actividad y especifica un recurso de variable en Azure Automation con el mismo nombre que la variable de Orchestrator. Esta acción se anota en el archivo **Runbook Converter - Summary.log** que se crea después de la conversión. Tiene que crear manualmente este recurso de variable en Azure Automation para poder usar el runbook.

Trabajo con parámetros de entrada de Orchestrator

Los runbooks de Orchestrator aceptan parámetros de entrada con la actividad `Initialize Data`. Si el runbook que se convierte incluye esta actividad, se crea un parámetro de entrada en el runbook de Azure Automation para cada parámetro de la actividad. Se crea una actividad Control de Script de flujo de trabajo en el runbook convertido que recupera y devuelve cada parámetro. Las actividades del runbook que usan un parámetro de entrada hacen referencia a la salida de esta actividad.

La razón por la que se usa esta estrategia es para reflejar mejor la funcionalidad del runbook de Orchestrator. Las actividades de los nuevos runbooks gráficos deben hacer referencia directamente a parámetros de entrada que usan un origen de datos de entrada de Runbook.

Actividad Invocar Runbook

Los runbooks de Orchestrator inician otros runbooks con la actividad `Invoke Runbook`. Si el runbook que se convierte incluye esta actividad y la opción `Wait for completion` está establecida, se crea una actividad de runbook para ella en el runbook convertido. Si la opción `Wait for completion` no está establecida, se crea una actividad de script de flujo de trabajo que usa Start-AzAutomationRunbook para iniciar el runbook. Después de importar el runbook convertido a Azure Automation, debe modificar esta actividad con la información especificada en la actividad.

Creación de recursos de Orchestrator

El Convertidor de runbooks no convierte recursos de Orchestrator. Debe crear manualmente los recursos de Orchestrator necesarios en Azure Automation.

Configuración de Hybrid Runbook Worker

Orchestrator almacena los runbooks en un servidor de base de datos y los ejecuta en servidores de runbooks, ambas cosas en el centro de datos local. Los runbooks de Azure Automation se almacenan en la nube de Azure y se pueden ejecutar en el centro de datos local con una instancia de Hybrid Runbook Worker. Configure un rol de trabajo que ejecute los runbooks convertidos desde Orchestrator, dado que están diseñados para ejecutarse en servidores locales y acceder a recursos locales.

Artículos relacionados

- Para más información sobre Orchestrator, consulte System Center 2012: Orchestrator.

- Obtenga más información sobre la automatización de la administración de servicios en Service Management Automation.
- Puede encontrar detalles de las actividades de Orchestrator en Actividades estándar de Orchestrator.
- Para obtener el kit de herramientas de migración de Orchestrator, consulte Descarga del kit de herramientas para migración de System Center Orchestrator.
- Para obtener información general sobre Hybrid Runbook Worker de Azure Automation, consulte Introducción a Hybrid Runbook Worker.

Last updated on 17/11/2025

Cambio de contexto en Azure Automation

Un cambio de contexto se produce cuando el contexto de un proceso cambia al contexto de un proceso diferente. Un contexto de Azure es un conjunto de información que define el destino de Azure PowerShell cmdlets. El contexto consta de las propiedades siguientes:

Expandir tabla

Propiedad	Descripción
Nombre	Nombre del contexto.
Cuenta	Nombre de usuario o entidad de servicio que se usa para autenticar las comunicaciones con Azure.
Entorno	Representa el Azure global o una de las nubes Azure nacionales, como Azure Government. También puede especificar una plataforma de nube híbrida, como Azure Stack.
Suscripción	Representa la suscripción Azure que contiene los recursos que desea administrar.
Inquilino	Una instancia dedicada y de confianza de Microsoft Entra ID que representa una sola organización.
Credenciales	La información utilizada por Azure para comprobar su identidad y confirmar la autorización para acceder a los recursos en Azure.

Cuando una cuenta inicia una sesión que puede acceder a varias suscripciones, cualquiera de esas suscripciones puede agregarse al contexto del usuario. Para garantizar la suscripción correcta, debe declararla al realizar la conexión. Por ejemplo, use `Add-AzAccount -Credential $Cred -subscription 'aaaa0a0a-bb1b-cc2c-dd3d-eeeeee4e4e4e'`. Sin embargo, pueden surgir problemas cuando los runbooks que administran una suscripción se ejecutan en el mismo proceso de espacio aislado que los demás runbooks que administran los recursos de otra suscripción desde la misma cuenta de Automation. Los cambios del contexto que realice un runbook pueden afectar al resto de runbooks que usan el contexto predeterminado. Dado que el contexto incluye información, como las credenciales que se usarán y la suscripción de destino, los cmdlets podrían dirigirse a la suscripción incorrecta, lo que provocaría `not found` o errores de permisos. Este problema se conoce como **cambio de contexto.**

Administrar contextos de Azure

Para evitar que los runbooks se ejecuten en recursos de una suscripción incorrecta, revise las recomendaciones siguientes:

1. Deshabilite el almacenamiento del contexto de espacio aislado en el runbook de Automation mediante el comando siguiente al principio de cada runbook: `Disable-AzContextAutosave -Scope Process`.
2. Los cmdlets Azure PowerShell admiten el parámetro `-DefaultProfile`. Este parámetro se agregó a todos los cmdlets de Az y Azure Resource Manager (AzureRM), para admitir la ejecución de varios scripts en el mismo proceso, permitiéndole especificar para cada cmdlet qué contexto utilizar. Guarde el objeto de contexto en el runbook cuando se cree y cada vez que se cambie. Después, haga referencia a este en cada llamada que se realice con el cmdlet de Az o AzureRM. Por ejemplo, `$AzureContext = Set-AzContext -SubscriptionId $subID`.
3. Pase el objeto de contexto al cmdlet de PowerShell, por ejemplo, `Get-AzVM -ResourceGroupName "myGroup" -DefaultProfile $AzureContext`.

Este es un fragmento de código de runbook de PowerShell que usa una identidad administrada asignada por el sistema de acuerdo con las recomendaciones para evitar el cambio de contexto.

PowerShell

```
# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process

# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext

# Pass context object - even though the context had just been set
# This is the step that guarantees the context will not be switched.
Get-AzVM -ResourceGroupName "resourceGroupName" -DefaultProfile $AzureContext | Select
Name
```

Posibles síntomas

Puede que no surja ningún problema aunque no siga estas recomendaciones, pero también es posible que lo haya. El problema subyacente de esta situación es la sincronización; es decir, depende de la acción que esté realizando cada runbook en el momento en que el otro runbook cambia su contexto. A continuación, se muestran algunos mensajes de error posibles. Sin

embargo, estos mensajes de error pueden deberse a condiciones no relacionadas con el cambio de contexto.

`The subscription named <subscription name> cannot be found.`

error

```
Get-AzVM : The client '<clientid>' with object id '<objectid>' does not have
authorization to perform action 'Microsoft.Compute/virtualMachines/read' over scope
'/subscriptions/<subscriptionIdOfSubscriptionWhichDoesntContainTheVM>/resourceGroups/R
EsourceGroupName/providers/Microsoft.Compute/virtualMachines/VMName '.
   ErrorCode: AuthorizationFailed
   StatusCode: 403
   ReasonPhrase: Forbidden Operation
   ID : <AGuidRepresentingTheOperation> At line:51 char:7 + $vm = Get-AzVM -
ResourceGroupName $ResourceGroupName -Name $UNBV... +
```

error

```
Get-AzureRmResource : Resource group "SomeResourceGroupName" could not be found.
... resources = Get-AzResource -ResourceGroupName $group.ResourceGro ...
                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : CloseError: (:) [Get-AzResource], CloudException
    + FullyQualifiedErrorId :
Microsoft.Azure.Commands.ResourceManager.Cmdlets.Implementation.GetAzureResourceCmdlet
```

Pasos siguientes

- Introducción a la autenticación de cuentas de Azure Automation
- ejecución de Runbook en Azure Automation
- Administrar módulos en Azure Automation
- Para solucionar problemas de runbook relacionados con el cambio de contexto en Azure Automation, consulte Solucionar problemas de runbook.

Last updated on 07/05/2026

Administración de certificados en Azure Automation

Azure Automation almacena los certificados de forma segura para que los runbooks o las configuraciones DSC puedan acceder a ellos mediante el cmdlet Get-AzAutomationCertificate para recursos de Azure Resource Manager. El almacenamiento seguro de certificados permite crear runbooks y configuraciones DSC que usan certificados para autenticación, o agregarlos a Azure o a recursos de terceros.

> **Nota:**
>
> Los recursos protegidos en Azure Automation incluyen credenciales, certificados, conexiones y variables cifradas. Estos recursos se cifran y se almacenan en Automation con una clave única que se genera para cada cuenta de Automation. Automation almacena la clave en el servicio de Key Vault administrado por el sistema. Antes de almacenar un recurso seguro, Automation carga la clave desde Key Vault y después la usa para cifrar el recurso.

Cmdlets de PowerShell para acceder a los certificados

Los cmdlets de la tabla siguiente permiten crear y administrar certificados de Automation con PowerShell. Se suministran como un componente de los módulos Az.

Expandir tabla

Cmdlet	Descripción
Get-AzAutomationCertificate	Recupera información sobre un certificado para utilizarlo en un runbook o en la configuración de DSC. Solo puede recuperar el certificado mediante el cmdlet `Get-AutomationCertificate` interno.
New-AzAutomationCertificate	Crea un certificado nuevo en Automation.
Remove-AzAutomationCertificate	Quita un certificado de Automation.
Set-AzAutomationCertificate	Establece las propiedades de un certificado existente, incluidos la carga del archivo de certificado y el establecimiento de la contraseña para un archivo **.pfx**.

Cmdlets internos para acceder a los certificados

El cmdlet interno de la tabla siguiente se usa para acceder a los certificados de los runbooks. Este cmdlet se incluye en el módulo global `Orchestrator.AssetManagement.Cmdlets`. Para más información, consulte Cmdlets internos.

Expandir tabla

Cmdlet interno	Descripción
`Get-AutomationCertificate`	Obtiene un certificado para usarlo en un runbook o una configuración de DSC. Devuelve un objeto System.Security.Cryptography.X509Certificates.X509Certificate2.

> **Nota:**
>
> Debe evitar el uso de variables en el parámetro `Name` de `Get-AutomationCertificate` en un runbook o configuración de DSC. Estas variables pueden complicar la detección de dependencias entre runbooks o configuraciones DSC y variables de Automation durante el tiempo de diseño.

Funciones de Python para acceder a los certificados

La función de la tabla siguiente se usa para acceder a los certificados de un runbook de Python 2 y 3. Actualmente, los runbooks de Python 3 están en versión preliminar.

Expandir tabla

Función	Descripción
`automationassets.get_automation_certificate`	Recupera información de un recurso de certificado.

> **Nota:**
>
> Para acceder a las funciones del recurso, debe importar el módulo `automationassets` al principio del runbook de Python.

Crear un nuevo certificado

Cuando crea un certificado nuevo, debe cargar un archivo .cer o .pfx a Automation. Si marca el certificado como exportable, podrá transferirlo fuera del almacén de certificados de Automation. Si no es exportable, solo se puede usar para firmar dentro del runbook o la configuración de DSC.

> ⓘ **Nota:**
>
> Azure Automation requiere que el certificado tenga el proveedor **Microsoft Enhanced RSA and AES Cryptographic Provider**.

Creación de un certificado nuevo con Azure Portal

1. En la cuenta de Automation, en el panel izquierdo, seleccione **Certificados** en **Recurso compartido**.
2. En la página **Certificados**, seleccione **agregar un certificado**.
3. Escriba un nombre para el certificado en el campo **Nombre**.
4. Para buscar un archivo **.cer** o **.pfx**, elija **Seleccionar un archivo** en **Cargar un archivo de certificado**. Si selecciona un archivo **.pfx**, especifique una contraseña e indique si se puede exportar. Si usa el portal de Azure Automation para cargar certificados, es posible que se produzca un error en las cuentas de asociados (CSP). Se recomienda usar cmdlets de PowerShell como solución alternativa para este problema.
5. Seleccione **Crear** para guardar el recurso de certificado nuevo.

Creación de un certificado nuevo con PowerShell

En el ejemplo siguiente se muestra cómo crear un nuevo certificado de Automation y marcarlo como exportable. En este ejemplo se importa un archivo **.pfx** existente.

PowerShell

```
$certificateName = 'MyCertificate'
$PfxCertPath = '.\MyCert.pfx'
$CertificatePassword = ConvertTo-SecureString -String 'P@$$w0rd' -AsPlainText -Force
$ResourceGroup = "ResourceGroup01"

New-AzAutomationCertificate -AutomationAccountName "MyAutomationAccount" -Name
$certificateName -Path $PfxCertPath -Password $CertificatePassword -Exportable -
ResourceGroupName $ResourceGroup
```

Creación de un certificado nuevo con una plantilla de Resource Manager

En el ejemplo siguiente se muestra cómo implementar un certificado en su cuenta de Automation mediante una plantilla de Resource Manager a través de PowerShell:

PowerShell

```
$AutomationAccountName = "<automation account name>"
$PfxCertPath = '<PFX cert path and filename>'
$CertificatePassword = '<password>'
$certificateName = '<certificate name>' #A name of your choosing
$ResourceGroupName = '<resource group name>' #The one that holds your automation
account
$flags =
[System.Security.Cryptography.X509Certificates.X509KeyStorageFlags]::Exportable `
    -bor
[System.Security.Cryptography.X509Certificates.X509KeyStorageFlags]::PersistKeySet `
    -bor
[System.Security.Cryptography.X509Certificates.X509KeyStorageFlags]::MachineKeySet
# Load the certificate into memory
$PfxCert = New-Object -TypeName
System.Security.Cryptography.X509Certificates.X509Certificate2 -ArgumentList
@($PfxCertPath, $CertificatePassword, $flags)
# Export the certificate and convert into base 64 string
$Base64Value =
[System.Convert]::ToBase64String($PfxCert.Export([System.Security.Cryptography.X509C
ertificates.X509ContentType]::Pkcs12))
$Thumbprint = $PfxCert.Thumbprint


$json = @"
{
    ``$schema': 'https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#',
    'contentVersion': '1.0.0.0',
    'resources': [
        {
            'name': '$AutomationAccountName/$certificateName',
            'type': 'Microsoft.Automation/automationAccounts/certificates',
            'apiVersion': '2015-10-31',
            'properties': {
                'base64Value': '$Base64Value',
                'thumbprint': '$Thumbprint',
                'isExportable': true
            }
        }
    ]
}
"@

$json | out-file .\template.json
New-AzResourceGroupDeployment -Name NewCert -ResourceGroupName $ResourceGroupName -
TemplateFile .\template.json
```

Obtener un certificado

Para recuperar un certificado, use el cmdlet `Get-AutomationCertificate` interno. No puede usar el cmdlet Get-AzAutomationCertificate porque devuelve información sobre el recurso de certificado, pero no el certificado en sí.

Ejemplo de un runbook textual

PowerShell

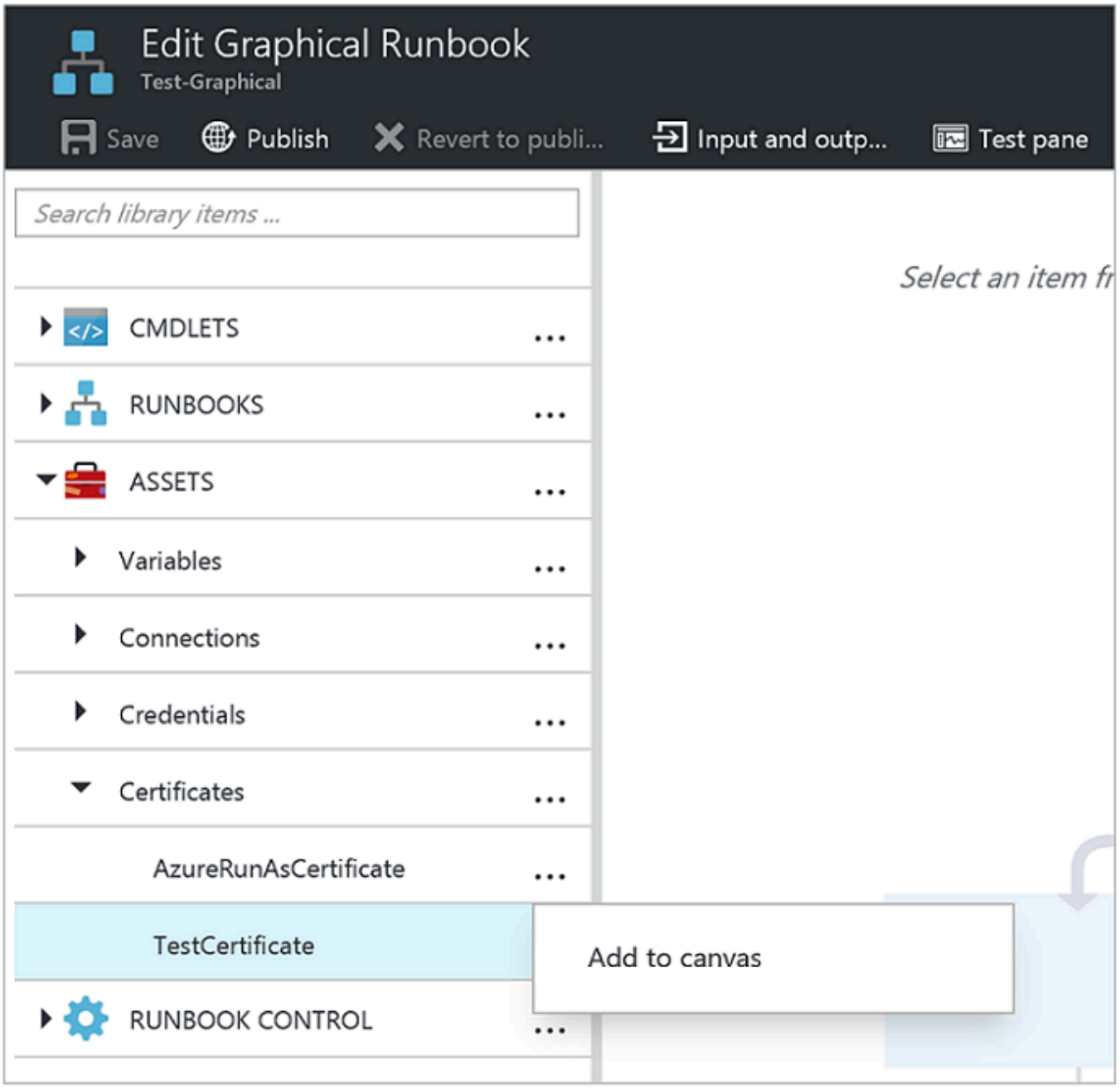
En el ejemplo siguiente se muestra cómo agregar un certificado a un servicio en la nube en un runbook. En este ejemplo, la contraseña se recupera a partir de una variable de automatización cifrada.

PowerShell

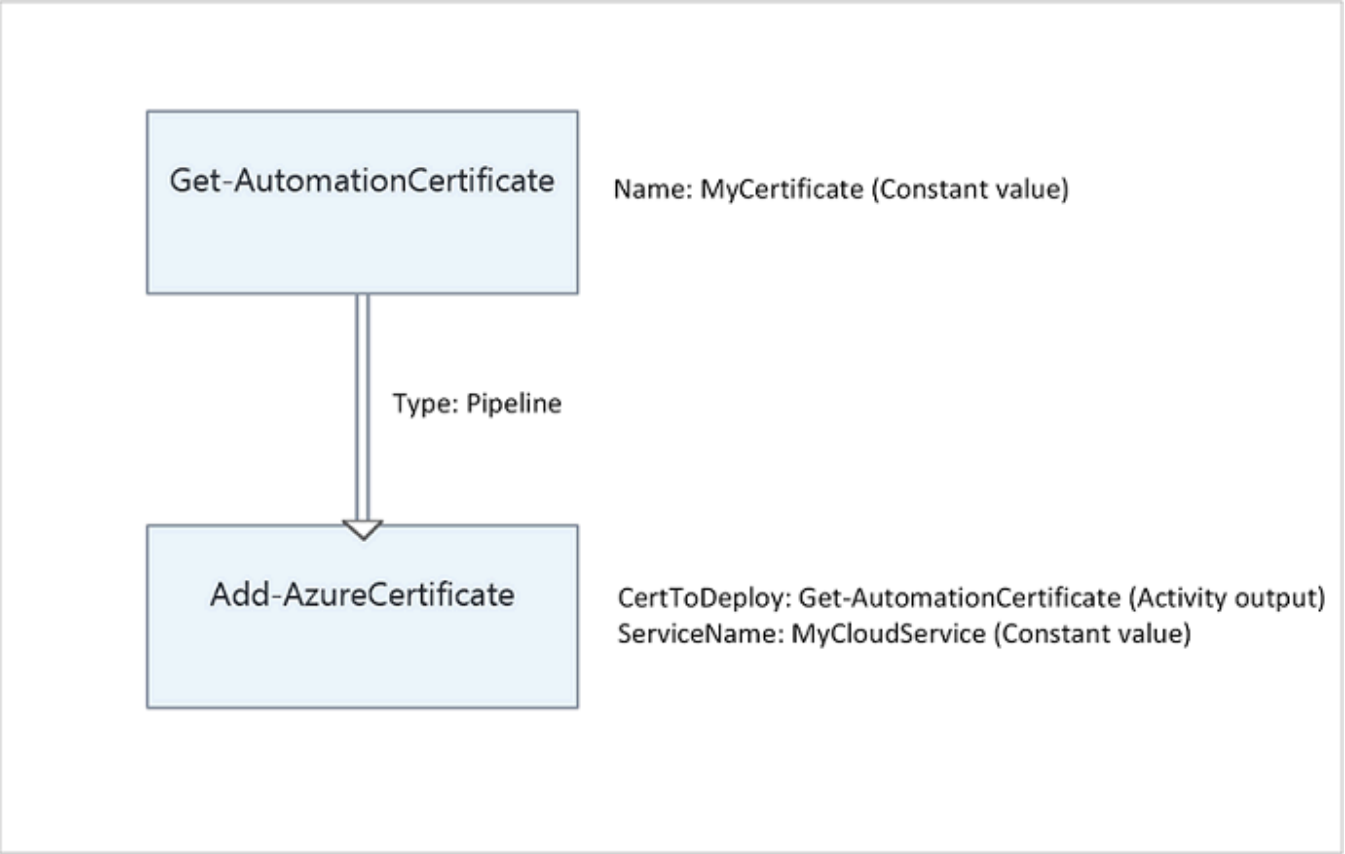
```
$serviceName = 'MyCloudService'
$cert = Get-AutomationCertificate -Name 'MyCertificate'
$certPwd = Get-AzAutomationVariable -ResourceGroupName "ResourceGroup01" `
-AutomationAccountName "MyAutomationAccount" -Name 'MyCertPassword'
Add-AzureCertificate -ServiceName $serviceName -CertToDeploy $cert
```

Ejemplos de un runbook gráfico

Para agregar una actividad para el cmdlet `Get-AutomationCertificate` interno a un runbook gráfico, haga clic con el botón derecho en el certificado en el panel Biblioteca y seleccione **Agregar al lienzo**.



La imagen siguiente muestra un ejemplo de cómo usar un certificado en un runbook gráfico.



Pasos siguientes

- Para más información sobre los cmdlets que se usan para acceder a los certificados, consulte Administración de módulos en Azure Automation.
- Para obtener información general sobre los runbooks, consulte Ejecución de un runbook en Azure Automation.
- Para más información sobre las configuraciones DSC, vea Introducción a State Configuration de Azure Automation.

Last updated on 17/11/2025

Administración de conexiones en Azure Automation

Un recurso de conexión de Azure Automation contiene la información que se muestra a continuación. Esta información es necesaria para conectarse a aplicaciones o servicios externos desde un runbook o la configuración de DSC.

- Información necesaria para la autenticación, como el nombre de usuario y la contraseña
- Información de conexión, como la dirección URL o el puerto

El recurso de conexión mantiene juntas todas las propiedades para conectarse a una aplicación determinada, por lo que no es necesario crear varias variables. Puede editar los valores de una conexión en un lugar y pasar el nombre de una conexión a un runbook o una configuración de DSC en un solo parámetro. El runbook o la configuración acceden a las propiedades de una conexión mediante el cmdlet `Get-AutomationConnection` interno.

Cuando crea una conexión, debe especificar un tipo de conexión. El tipo de conexión es una plantilla que define un conjunto de propiedades. Puede agregar un tipo de conexión a Azure Automation mediante un módulo de integración con un archivo de metadatos. Se puede crear un tipo de conexión mediante la API de Azure Automation si el módulo de integración incluye un tipo de conexión y se importa en su cuenta de Automation.

> **Nota:**
>
> Los recursos protegidos en Azure Automation incluyen credenciales, certificados, conexiones y variables cifradas. Estos recursos se cifran y se almacenan en Azure Automation con una clave única que se genera para cada cuenta de Automation. Azure Automation almacena la clave en la instancia de Key Vault administrada por el sistema. Antes de almacenar un recurso seguro, Azure Automation carga la clave desde Key Vault y después la usa para cifrar el recurso.

Tipos de conexión

Azure Automation pone a disposición de los usuarios los siguientes tipos de conexión integrados:

- `Azure`: representa una conexión que se usa para administrar recursos clásicos.
- `AzureServicePrincipal`: representa una conexión que se usa para administrar recursos en Azure mediante una entidad de servicio.

- `AzureClassicCertificate`: este tipo de conexión se usa para administrar los recursos de Azure que se crearon mediante el modelo de implementación clásica que no admite la autenticación de la entidad de servicio.

Cmdlets de PowerShell para acceder a las conexiones

Los cmdlets de la tabla siguiente permiten crear y administrar conexiones de Automation con PowerShell. Se suministran como un componente de los módulos Az.

Expandir tabla

Cmdlet	Descripción
Get-AzAutomationConnection	Recupera información acerca de una conexión.
New-AzAutomationConnection	Crea una conexión nueva.
Remove-AzAutomationConnection	Quita una conexión existente.
Set-AzAutomationConnectionFieldValue	Establece el valor de un campo determinado para una conexión existente.

Cmdlets internos para acceder a las conexiones

El cmdlet interno de la tabla siguiente se usa para acceder a las conexiones de los runbooks y las configuraciones de DSC. Este cmdlet se incluye en el módulo global `Orchestrator.AssetManagement.Cmdlets`. Para más información, consulte Cmdlets internos.

Expandir tabla

Cmdlet interno	Descripción
`Get-AutomationConnection`	Recupera los valores de los distintos campos de la conexión y los devuelve como una tabla hash. Esta tabla hash se puede usar después con los comandos adecuados en el runbook o la configuración de DSC.

> **Nota:**
>
> Evite el uso de variables con el parámetro `Name` de `Get-AutomationConnection`. El uso de variables en este caso puede complicar la detección de dependencias entre runbooks o configuraciones de DSC y los recursos de conexión durante el tiempo de diseño.

Funciones de Python para acceder a las conexiones

La función de la tabla siguiente se usa para acceder a las conexiones de un runbook de Python 2 y 3. Actualmente, los runbooks de Python 3 están en versión preliminar.

Expandir tabla

Función	Descripción
`automationassets.get_automation_connection`	Recupera una conexión. Devuelve un diccionario con las propiedades de la conexión.

> **Nota:**
>
> Para acceder a las funciones del activo, debe importar el módulo `automationassets` en la parte superior del runbook de Python.

Crear una nueva conexión

Creación de una conexión con Azure Portal

Para crear una conexión nueva en Azure Portal:

1. En la cuenta de Automation, haga clic en **Conexiones** en **Recursos compartidos**.
2. En la página Conexiones, haga clic en **+ Agregar una conexión**.
3. En el campo **Tipo** del panel Nueva conexión, seleccione el tipo de conexión que desea crear. Las opciones son `Azure`, `AzureServicePrincipal` y `AzureClassicCertificate`.
4. El formulario presenta las propiedades del tipo de conexión que ha elegido. Complete el formulario y haga clic en **Crear** para guardar la conexión nueva.

Creación de una conexión con Windows PowerShell

Cree una conexión con Windows PowerShell mediante el cmdlet `New-AzAutomationConnection`. Este cmdlet tiene un parámetro `ConnectionFieldValues` que espera una tabla hash que defina los valores de cada una de las propiedades definidas por el tipo de conexión.

Puede usar los siguientes comandos de ejemplo para crear una conexión que se pueda usar para la autenticación mediante la entidad de servicio de Azure.

PowerShell

```
$ConnectionAssetName = "AzureConnection"
$ConnectionFieldValues = @{"ApplicationId" = $Application.ApplicationId; "TenantId"
= $TenantID.TenantId; "CertificateThumbprint" = $Cert.Thumbprint; "SubscriptionId" =
$SubscriptionId}
New-AzAutomationConnection -ResourceGroupName $ResourceGroup -AutomationAccountName
$AutomationAccountName -Name $ConnectionAssetName -ConnectionTypeName
AzureServicePrincipal -ConnectionFieldValues $ConnectionFieldValues
```

Si intenta crear un recurso de conexión para conectarse a un servicio o a una aplicación con un método de autenticación diferente, la operación generará un error porque el tipo de conexión no se ha definido todavía en la cuenta de Automation. Para más información sobre cómo crear su propio tipo de conexión para un módulo personalizado, consulte Agregar un tipo de conexión.

Agregar un tipo de conexión

Si el runbook o la configuración de DSC se conectan a un servicio externo, debe definir un tipo de conexión en un módulo personalizado denominado "módulo de integración". Este módulo incluye un archivo de metadatos que especifica las propiedades del tipo de conexión, se denomina **<ModuleName>-Automation.json** y se encuentra en la carpeta del módulo del archivo **.zip** comprimido. Este archivo contiene los campos de una conexión que se necesitan para conectarse al sistema o servicio que representa el módulo. Con este archivo, puede establecer los nombres de campo, los tipos de datos, el estado de cifrado y el estado opcional del tipo de conexión. No se admiten varios tipos de conexión en este archivo.

El ejemplo siguiente es una plantilla en el formato de archivo **.json** que define propiedades de nombre de usuario y contraseña de un tipo de conexión predeterminado denominado `MyModuleConnection`:

JSON

```
{
   "ConnectionFields": [
   {
      "IsEncrypted":  false,
      "IsOptional":  true,
      "Name":  "Username",
      "TypeName":  "System.String"
   },
   {
      "IsEncrypted":  true,
      "IsOptional":  false,
      "Name":  "Password",
      "TypeName":  "System.String"
   }
   ],
```

```
    "ConnectionTypeName":  "MyModuleConnection",
    "IntegrationModuleName":  "MyModule"
}
```

Obtención de una conexión en un runbook o una configuración de DSC

Recupere una conexión de un runbook o una configuración de DSC con el cmdlet `Get-AutomationConnection` interno. Este cmdlet es preferible al cmdlet `Get-AzAutomationConnection`, ya que recupera los valores de conexión en lugar de la información sobre la conexión.

PowerShell

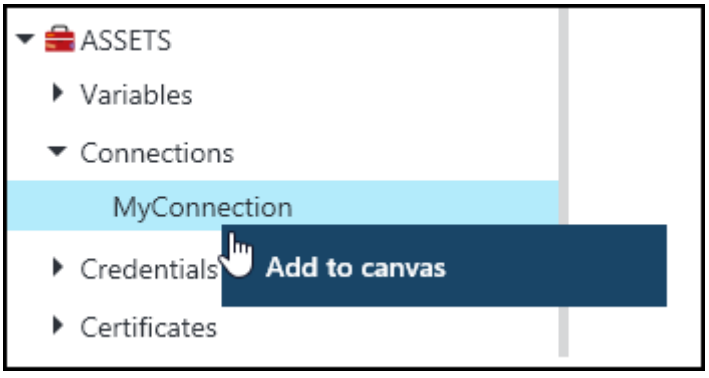
El ejemplo siguiente muestra cómo utilizar una conexión para autenticar con recursos de Azure Resource Manager en su runbook. Utiliza el recurso de conexión, que hace referencia a la entidad de servicio basada en certificado.

PowerShell

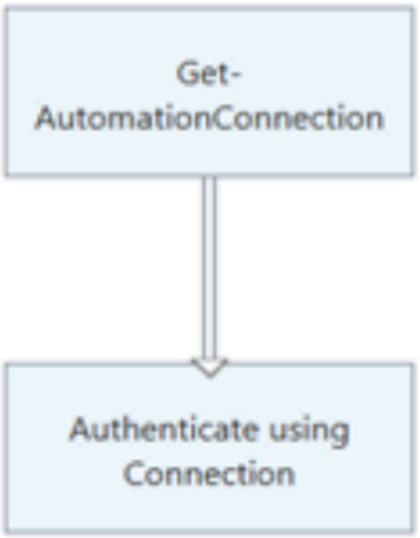
```
$Conn = Get-AutomationConnection -Name AzureConnection
Connect-AzAccount -ServicePrincipal -Tenant $Conn.TenantID -ApplicationId
$Conn.ApplicationID -CertificateThumbprint $Conn.CertificateThumbprint
```

Ejemplos de runbook gráfico

Puede agregar una actividad para el cmdlet `Get-AutomationConnection` interno a un runbook gráfico. Haga clic con el botón derecho en la conexión en el panel Biblioteca del editor gráfico y seleccione **Agregar al lienzo**.



La imagen siguiente muestra un ejemplo de cómo usar un objeto de conexión en un runbook gráfico.



Name: AzureConnection (Constant Value)

Parameter Set: Service PrincipalCertificate

ApplicationId: Get AutomationConnection (Activity output) Field path: ApplicationId) CertificateThumbprint: Get Automation Connection (Activity output) Field path: Certificate ThumbprintServicePrincipal: True (Constant value) TenantId: Get AutomationConnection (Activity output) Field Path: TenantId

Pasos siguientes

- Para obtener más información sobre los cmdlets que se usan para acceder a las conexiones, consulte Administración de módulos en Azure Automation.
- Para obtener información general sobre los runbooks, consulte Ejecución de un runbook en Azure Automation.
- Para más información sobre las configuraciones de DSC, consulte Introducción a State Configuration.

Last updated on 17/11/2025

Administración de credenciales en Azure Automation

Un recurso de credencial de Automation incluye un objeto que contiene credenciales de seguridad, como un nombre de usuario y una contraseña. Los runbooks y las configuraciones de DSC usan cmdlets que aceptan un objeto PSCredential para la autenticación. Como alternativa, pueden extraer el nombre de usuario y la contraseña del objeto `PSCredential` para proporcionarlos a alguna aplicación o servicio que requiera autenticación.

> **Nota:**
>
> Los recursos protegidos en Azure Automation incluyen credenciales, certificados, conexiones y variables cifradas. Estos recursos se cifran y se almacenan en Azure Automation con una clave única que se genera para cada cuenta de Automation. Azure Automation almacena la clave en la instancia de Key Vault administrada por el sistema. Antes de almacenar un recurso seguro, Azure Automation carga la clave desde Key Vault y después la usa para cifrar el recurso.

> **Nota:**
>
> Para obtener información acerca de cómo ver o eliminar datos personales, consulte Solicitudes generales de interesados para el RGPD, Solicitudes de los interesados de Azure para el RGPD, o Solicitudes de los interesados de Windows para el RGPD, en función de sus necesidades y áreas específicas. Para más información sobre RGPD, consulte Información sobre los procedimientos recomendados para el cumplimiento del RGPD y la sección RGPD del portal de confianza de servicios.

Cmdlets de PowerShell usados para acceder a credenciales

Los cmdlets de la tabla siguiente permiten crear y administrar credenciales de Automation con PowerShell. Se suministran como un componente de los módulos Az.

Expandir tabla

Cmdlet	Descripción
Get-AzAutomationCredential	Recupera un objeto CredentialInfo que contiene metadatos sobre la

Cmdlet	Descripción
	credencial. El cmdlet no recupera el objeto `PSCredential` mismo.
New-AzAutomationCredential	Crea una nueva credencial de Automation.
Remove-AzAutomationCredential	Quita una credencial de Automation.
Set-AzAutomationCredential	Establece las propiedades para una credencial de Automatización existente.

Otros cmdlets usados para acceder a credenciales

Los cmdlets de la tabla siguiente se usan para acceder a las credenciales en los runbook y las configuraciones de DSC.

Expandir tabla

Cmdlet	Descripción
`Get-AutomationPSCredential`	Obtiene un objeto `PSCredential` para usarlo en un runbook o una configuración de DSC. La mayoría de las veces debe usar este cmdlet interno en lugar del cmdlet `Get-AzAutomationCredential`, ya que este último solo recupera la información de credenciales. Esta información no suele ser útil para pasar a otro cmdlet.
Get-Credential	Obtiene una credencia con una solicitud de nombre de usuario y contraseña. Este cmdlet forma parte del módulo Microsoft.PowerShell.Security predeterminado. Consulte Módulos predeterminados.
New-AzureAutomationCredential	Crea un recurso de credencial. Este cmdlet forma parte del módulo de Azure predeterminado. Consulte Módulos predeterminados.

Para recuperar objetos `PSCredential` en el código, debe importar el módulo `Orchestrator.AssetManagement.Cmdlets`. Para más información, consulte Administración de módulos en Azure Automation.

PowerShell

```
Import-Module Orchestrator.AssetManagement.Cmdlets -ErrorAction SilentlyContinue
```

Nota:

Debe evitar el uso de variables en el parámetro `Name` de `Get-AutomationPSCredential`. Su uso puede complicar la detección de dependencias entre runbooks o configuraciones de DSC y recursos de credenciales durante el tiempo de diseño.

Funciones de Python que acceden a las credenciales

La función de la tabla siguiente se usa para acceder a las credenciales de un runbook de Python 2 y 3. Actualmente, los runbooks de Python 3 están en versión preliminar.

Expandir tabla

Función	Descripción
`automationassets.get_automation_credential`	Recupera información acerca de un recurso de credencial.

Nota:

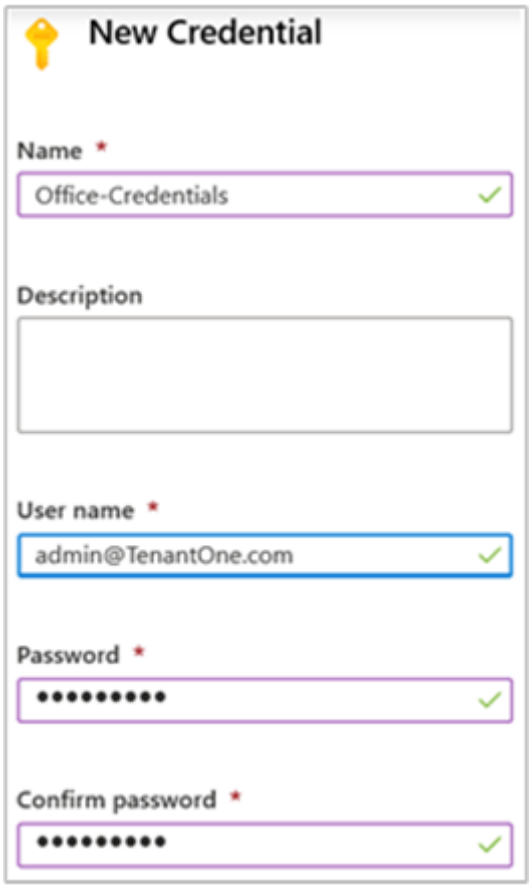
Para acceder a las funciones del recurso, importe el módulo `automationassets` en la parte superior del runbook de Python.

Creación de un nuevo recurso de credencial

Puede crear un recurso de credencial mediante Azure Portal o Windows PowerShell.

Creación de un nuevo recurso de credencial con Azure Portal

1. En la cuenta de Automation, en el panel izquierdo, seleccione **Credenciales** en **Recursos compartidos**.
2. En la página **Credenciales**, seleccione **Agregar una credencial**.
3. En el panel Nueva credencial, escriba un nombre de credencial adecuado que siga los estándares de nomenclatura.
4. Escriba el identificador de acceso en el campo **Nombre de usuario**.
5. En ambos campos de contraseña, escriba la clave de acceso secreta.



6. Si la casilla de autenticación multifactor está activada, desactívela.

7. Haga clic en **Crear** para guardar el nuevo recurso de credencial.

> ⓘ **Nota:**
>
> Azure Automation no admite cuentas de usuario que usen la autenticación multifactor.

Creación de un nuevo recurso de credencial con Windows PowerShell

En el ejemplo siguiente se muestra cómo crear un recurso de credencial de Automation. Primero se crea un objeto `PSCredential` con el nombre y la contraseña y, luego, se usa para crear el recurso de credencial. En su lugar, puede usar el cmdlet `Get-Credential` para solicitar al usuario que escriba un nombre y una contraseña.

PowerShell

```
$user = "MyDomain\MyUser"
$pw = ConvertTo-SecureString "PassWord!" -AsPlainText -Force
$cred = New-Object –TypeName System.Management.Automation.PSCredential –ArgumentList
$user, $pw
```

```
New-AzureAutomationCredential -AutomationAccountName "MyAutomationAccount" -Name
"MyCredential" -Value $cred
```

Obtención de un recurso de credencial

Un runbook o una configuración de DSC recupera un recurso de credencial con el cmdlet `Get-AutomationPSCredential` interno. Este cmdlet recupera un objeto `PSCredential` que puede usar con un cmdlet que requiere una credencial. También puede recuperar las propiedades del objeto de credencial para usarlas individualmente. El objeto tiene propiedades para el nombre de usuario y la contraseña segura.

> **Nota:**
>
> El cmdlet `Get-AzAutomationCredential` no recupera ningún objeto `PSCredential` que se pueda usar para la autenticación. Solo proporciona información acerca de la credencial. Si necesita usar una credencial en un runbook, debe recuperarla como un objeto `PSCredential` mediante `Get-AutomationPSCredential`.

Como alternativa, puede usar el método GetNetworkCredential para recuperar un objeto NetworkCredential que representa una versión no segura de la contraseña.

Ejemplo de un runbook textual

PowerShell

En el ejemplo siguiente se muestra cómo usar una credencial de PowerShell en un runbook. Se recupera la credencial y se asigna su nombre de usuario y contraseña a las variables.

PowerShell

```
$myCredential = Get-AutomationPSCredential -Name 'MyCredential'
$userName = $myCredential.UserName
$securePassword = $myCredential.Password
$password = $myCredential.GetNetworkCredential().Password
```

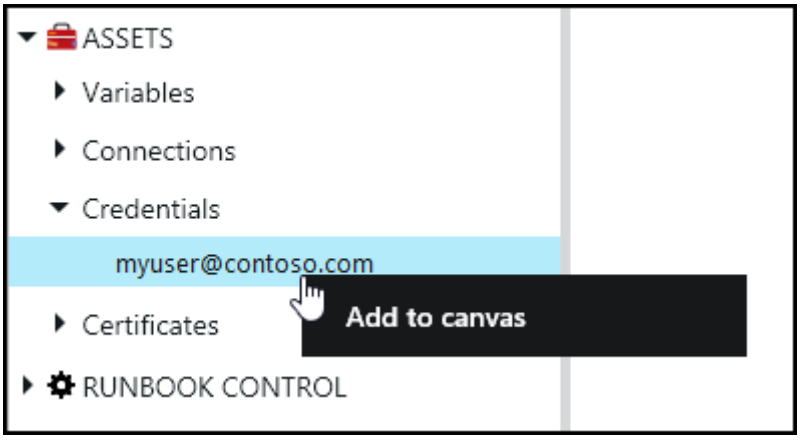
También puede usar una credencial para autenticarse en Azure con Connect-AzAccount después de conectarse primero a una identidad administrada. En ese ejemplo se usa una identidad administrada asignada por el sistema.

PowerShell

```
# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process

# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext

# Get credential
$myCred = Get-AutomationPSCredential -Name "MyCredential"
$userName = $myCred.UserName
$securePassword = $myCred.Password
$password = $myCred.GetNetworkCredential().Password

$myPsCred = New-Object System.Management.Automation.PSCredential
($userName,$securePassword)

# Connect to Azure with credential
$AzureContext = (Connect-AzAccount -Credential $myPsCred -TenantId
$AzureContext.Subscription.TenantId).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription `
    -TenantId $AzureContext.Subscription.TenantId `
    -DefaultProfile $AzureContext
```

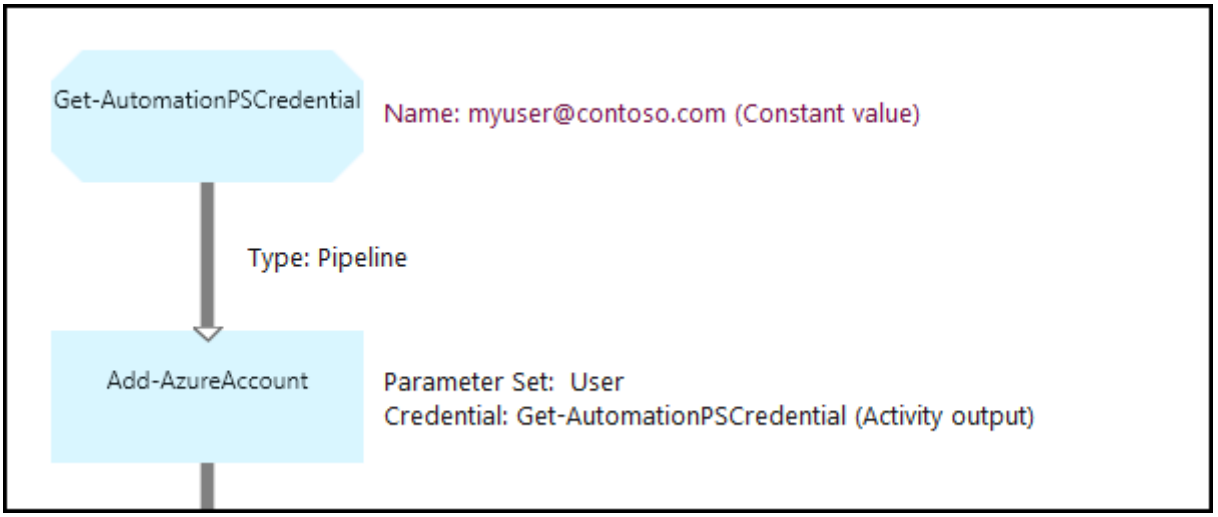
Ejemplos de un runbook gráfico

Para agregar una actividad para el cmdlet `Get-AutomationPSCredential` interno a un runbook gráfico, haga clic con el botón derecho en la credencial en el panel Biblioteca del editor gráfico y, después, seleccione **Agregar al lienzo**.



La imagen siguiente muestra un ejemplo de cómo usar una credencial en un runbook gráfico. En este caso, la credencial proporciona la autenticación de un runbook en los recursos de Azure, como se describe en Uso de Microsoft Entra ID en Azure Automation para autenticarse

en Azure. La primera actividad recupera la credencial que tiene acceso a la suscripción de Azure. A continuación, la actividad de conexión de la cuenta usa esta credencial para proporcionar autenticación para cualquier actividad que venga después. Aquí se usa un vínculo de canalización, debido a que `Get-AutomationPSCredential` espera un solo objeto.



Uso de credenciales en una configuración de DSC

Aunque las configuraciones de DSC en Azure Automation pueden funcionar con recursos de credenciales mediante `Get-AutomationPSCredential`, también los pueden pasar a través de parámetros. Para obtener más información, consulte Compilación de configuraciones en DSC de Azure Automation.

Pasos siguientes

- Para más información sobre los cmdlets que se usan para acceder a los certificados, consulte Administración de módulos en Azure Automation.
- Para obtener información general sobre los runbooks, consulte Ejecución de un runbook en Azure Automation.
- Para más información sobre las configuraciones DSC, vea Introducción a State Configuration de Azure Automation.

Last updated on 17/11/2025

Administración de módulos en Azure Automation

> **Nota:**
>
> A partir del **1 de febrero de 2025**, Azure Automation *descontinuará* la ejecución de todos los runbooks que usan módulos AzureRM. A partir del **1 de noviembre de 2024**, no podrá crear nuevos runbooks utilizando los módulos AzureRM. El módulo AzureRM PowerShell ha quedado oficialmente en desuso a partir del **29 de febrero de 2024**. Le recomendamos que migre del módulo AzureRM al módulo Az PowerShell para garantizar un soporte y actualizaciones continuos. Aunque el módulo AzureRM puede seguir funcionando, ya no se mantiene ni se admite, y el uso continuado del módulo AzureRM está en riesgo del usuario. Para obtener más información, consulte recursos de migración para obtener instrucciones sobre la transición al módulo Az.

Azure Automation usa una serie de módulos de PowerShell para habilitar cmdlets en runbooks y recursos de DSC en configuraciones de DSC. Los módulos compatibles incluyen:

- Módulos Az de Azure PowerShell.
- Otros módulos de PowerShell.
- Módulo interno `Orchestrator.AssetManagement.Cmdlets` (no disponible en Hybrid Runbook Worker de Linux).
- Módulos de Python 2 y Python 3.
- Módulos personalizados creados por el usuario.

Al crear una cuenta de Automation, Azure Automation importa algunos módulos de forma predeterminada. Consulte Módulos predeterminados.

> **Importante**
>
> - La nueva experiencia del entorno de runtime permite administrar módulos y paquetes configurando el entorno de ejecución del trabajo. En la nueva experiencia, las hojas Módulos y Paquetes no están disponibles. Para administrar los módulos y paquetes, consulte Administración del entorno de runtime y los runbooks asociados.

- Los paquetes y módulos cargados en una cuenta de Automation se tratan como **contenido propiedad del cliente (IP del cliente).** Microsoft no inspecciona ni valida estos paquetes. Los paquetes se **cargan en tiempo de ejecución en el entorno de cliente** y los clientes son responsables de garantizar su integridad y seguridad.

Espacios aislados

Cuando Automation ejecuta trabajos de compilación de runbook y DSC, carga los módulos en espacios aislados en los que se pueden ejecutar los runbooks y se pueden compilar las configuraciones de DSC. Automation también coloca automáticamente los recursos de DSC en los módulos del servidor de extracción de DSC. Las máquinas pueden extraer los recursos cuando aplican las configuraciones de DSC.

El espacio aislado de nube admite 48 llamadas del sistema como máximo y, por motivos de seguridad, restringe todas las demás. Otras funcionalidades, como la administración de credenciales y algunas redes, no se admiten en el espacio aislado en la nube.

Debido al número de módulos y cmdlets incluidos, es difícil saber de antemano cuál de los cmdlets realizará llamadas no admitidas. Por lo general, se han detectado problemas con cmdlets que requieren acceso con privilegios elevados, que requieren una credencial como parámetro o con cmdlets relacionados con redes. Los cmdlets que realizan operaciones completas de red no se admiten en el entorno de pruebas, incluidos Connect-AipService del módulo de PowerShell AIPService y Resolve-DnsName del módulo DNSClient.

Estas son limitaciones conocidas del espacio aislado. La solución alternativa recomendada es implementar un Hybrid Runbook Worker o usar Azure Functions.

> **ⓘ Importante**
>
> No incluya la palabra clave "AzureRm" en ningún script diseñado para ejecutarse con el módulo Az. La inclusión de la palabra clave, incluso en un comentario, puede hacer que AzureRm se cargue y, a continuación, entre en conflicto con el módulo Az.

Módulos predeterminados

Todas las nuevas cuentas de Automation tienen la versión más reciente del módulo Az de PowerShell importado de manera predeterminada. El módulo Az reemplaza AzureRM y es el

módulo recomendado para usarlo con Azure. Los **módulos predeterminados** de la nueva cuenta de Automation incluyen los 24 módulos AzureRM existentes y más de 60 módulos Az.

Hay una opción nativa para actualizar los módulos al módulo Az más reciente por parte del usuario para las cuentas de Automation. La operación controlará todas las dependencias del módulo en el fondo, lo que elimina las molestias de actualizar los módulos manualmente o ejecutar el runbook para actualizar los módulos de Azure.

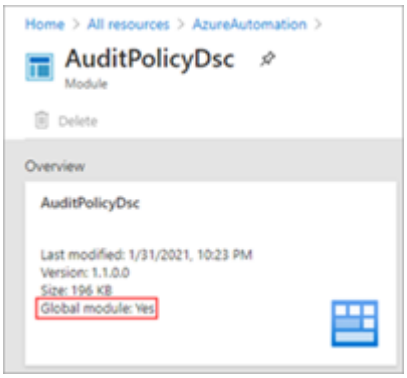
Si la cuenta de Automation existente solo tiene módulos AzureRM, la opción Actualizar módulos Az actualizará la cuenta de Automation con la versión seleccionada por el usuario del módulo Az.

Si la cuenta de Automation existente tiene AzureRM y algunos módulos Az, la opción importará los módulos Az restantes a la cuenta de Automation. Los módulos Az existentes tendrán preferencia y la operación de actualización no actualizará esos módulos. Esto es para asegurarse de que la operación del módulo de actualización no conduce a ningún error de ejecución de runbook mediante la actualización involuntaria de un módulo que usa un runbook. La manera recomendada para este escenario es eliminar primero los módulos Az existentes y, a continuación, realizar las operaciones de actualización para importar el módulo Az más reciente en la cuenta de Automation. Estos tipos de módulos, no importados de manera predeterminada, se conocen como **personalizados**. Los módulos **personalizados** siempre tendrán preferencia sobre los módulos **predeterminados**.

Por ejemplo: si ya tiene el módulo `Az.Aks` importado con la versión 2.3.0 que proporciona el módulo Az 6.3.0 e intenta actualizar el módulo Az a la versión 6.4.0 más reciente. La operación de actualización importará todos los módulos Az desde el paquete 6.4.0, excepto `Az.Aks`. Para tener la versión más reciente de `Az.Aks`, elimine primero el módulo existente y, a continuación, realice la operación de actualización, o bien también puede actualizar este módulo por separado, como se describe en Importación de módulos Az para importar una versión diferente de un módulo específico.

En la tabla siguiente se enumeran los módulos que Azure Automation importan de forma predeterminada al crear la cuenta de Automation. Automation puede importar versiones más recientes de estos módulos. Pero no se puede quitar la versión original de la cuenta de Automation aunque se elimine una versión más reciente.

Los módulos predeterminados también se conocen como módulos globales. En el portal de Azure, la propiedad **Global** será **true** al ver un módulo que se importó cuando se creó la cuenta.



> **Nota:**
>
> No es recomendable alterar módulos y runbooks de cuentas de Automation utilizadas para la implementación de Start/Stop VMs during off-hours

Expandir tabla

Nombre del módulo	Versión
Az.*	Consulte la lista completa en **Detalles del paquete** en Galería de PowerShell
AuditPolicyDsc	1.1.0.0
Azure	1.0.3
Azure.Storage	1.0.3
AzureRM.Automation	1.0.3
AzureRM.Compute	1.2.1
AzureRM.Profile	1.0.3
AzureRM.Resources	1.0.3
AzureRM.Sql	1.0.3
AzureRM.Storage	1.0.3
ComputerManagementDsc	5.0.0.0
GPRegistryPolicyParser	0,2
Microsoft.PowerShell.Core	0
Microsoft.PowerShell.Diagnostics	
Microsoft.PowerShell.Management	

Nombre del módulo	Versión
Microsoft.PowerShell.Security	
Microsoft.PowerShell.Utility	
Microsoft. WSMan.Management	
Orchestrator.AssetManagement.Cmdlets	1
PSDscResources	2.9.0.0
SecurityPolicyDsc	2.1.0.0
StateConfigCompositeResources	1
xDSCDomainjoin	1.1
xPowerShellExecutionPolicy	1.1.0.0
xRemoteDesktopAdmin	1.1.0.0

Cmdlets internos

Azure Automation proporciona cmdlets internos que están disponibles exclusivamente cuando los runbooks se ejecutan en el entorno de espacio aislado de Azure o en una instancia de Hybrid Runbook Worker de Windows. El módulo `Orchestrator.AssetManagement.Cmdlets`, que contiene estos cmdlets internos, se instala de forma predeterminada en la cuenta de Automation y, específicamente, cuando el rol de Hybrid Runbook Worker de Windows está instalado en un equipo Windows.

> **Nota:**
>
> Esta funcionalidad no está disponible en Hybrid Runbook Worker de Linux.

En la tabla siguiente se definen los cmdlets internos. Estos cmdlets están diseñados para usarse en lugar de los cmdlets de Azure PowerShell para interactuar con los recursos de tu cuenta de Automation. Los cmdlets recuperan secretos a partir de variables cifradas, credenciales y conexiones cifradas.

Expandir tabla

Nombre	Descripción
Get-AutomationCertificate	`Get-AutomationCertificate [-Name] <string> [<CommonParameters>]`
Get-AutomationConnection	`Get-AutomationConnection [-Name] <string> [-DoNotDecrypt] [<CommonParameters>]`
Get-AutomationPSCredential	`Get-AutomationPSCredential [-Name] <string> [<CommonParameters>]`
Get-AutomationVariable	`Get-AutomationVariable [-Name] <string> [-DoNotDecrypt] [<CommonParameters>]`
Set-AutomationVariable	`Set-AutomationVariable [-Name] <string> -Value <Object> [<CommonParameters>]`
Start-AutomationRunbook	`Start-AutomationRunbook [-Name] <string> [-Parameters <IDictionary>] [-RunOn <string>] [-JobId <guid>] [<CommonParameters>]`
Wait-AutomationJob	`Wait-AutomationJob -Id <guid[]> [-TimeoutInMinutes <int>] [-DelayInSeconds <int>] [-OutputJobsTransitionedToRunning] [<CommonParameters>]`

Tenga en cuenta que los cmdlets internos difieren en la nomenclatura de los cmdlets de Az y AzureRM. Los nombres de cmdlet internos no contienen palabras como `Azure` o `Az` en el nombre, pero usan la palabra `Automation`. Se recomienda su uso antes que el uso de los cmdlets Az o AzureRM durante la ejecución del runbook en un espacio aislado de Azure o en una instancia de Hybrid Runbook Worker de Windows porque requieren menos parámetros y se ejecutan en el contexto del trabajo durante la ejecución.

Use los cmdlets de Az o AzureRM para manipular recursos de Automation fuera del contexto de un runbook.

módulos de Python

Puede crear runbooks de Python 2 en Azure Automation. Para obtener información sobre el módulo de Python, consulte Administrar paquetes de Python 2 en Azure Automation.

Módulos personalizados

Azure Automation admite módulos personalizados de PowerShell que se crean para usarlos con los runbooks y las configuraciones de DSC. Un tipo de módulo personalizado es un módulo de integración que puede contener un archivo de metadatos para definir la funcionalidad

personalizada para los cmdlets del módulo. Un ejemplo del uso de un módulo de integración se proporciona en Incorporación de un tipo de conexión.

Azure Automation puede importar un módulo personalizado para que sus cmdlets estén disponibles. En segundo plano, almacena el módulo y lo usa en las zonas seguras de Azure, al igual que hace con otros módulos.

Migración a módulos Az

En esta sección se explica cómo migrar a los módulos Az en Automation. Para más información, consulte Migrate Azure PowerShell de AzureRM a Az.

No es recomendable ejecutar módulos Az y AzureRM en la misma cuenta de Automation. Cuando esté seguro de que quiere migrar de AzureRM a Az, es mejor confirmar completamente una migración entera. Automation suele reutilizar espacios aislados dentro de la cuenta de Automation para ahorrar en los tiempos de inicio. Si no realiza una migración completa del módulo, puede iniciar un trabajo que usa solo módulos AzureRM y luego iniciar otro trabajo que usa solo módulos Az. El espacio aislado pronto se bloquea, y recibe un error que indica que los módulos no son compatibles. Esta situación da lugar a bloqueos que se producen aleatoriamente de cualquier runbook o configuración específicos.

> **Nota:**
>
> Cuando se crea una cuenta de Automation, incluso después de la migración a módulos Az, Automation instala los módulos AzureRM de manera predeterminada.

Prueba de los runbooks y las configuraciones de DSC antes de la migración de módulos

Asegúrese de probar todos los runbooks y las configuraciones de DSC cuidadosamente en una cuenta de Automation independiente antes de migrar a los módulos Az.

Detención y anulación de la programación de todos los runbooks que usan módulos de AzureRM

Para asegurarse de no ejecutar ningún runbook ni configuración de DSC existente que use módulos AzureRM, debe detener y anular la programación de todos los runbooks y las configuraciones afectados. En primer lugar, asegúrese de revisar cada runbook o configuración

de DSC y sus programaciones por separado para garantizar que puede volver a programar el elemento en el futuro en caso necesario.

Cuando esté listo para quitar las programaciones, puede usar Azure Portal o el cmdlet Remove-AzAutomationSchedule. Consulte Eliminación de una programación.

Eliminación de los módulos AzureRM

Es posible quitar los módulos AzureRM antes de importar los módulos Az. Sin embargo, si lo hace, puede interrumpir la sincronización del control de código fuente y provocar errores en los scripts que siguen programados. Si decide quitar los módulos, ver Desinstalación de AzureRM.

Importación de módulos Az

Al importar un módulo Az en la cuenta de Automation, no se importa automáticamente ese módulo en la sesión de PowerShell que usan los runbooks. Los módulos se importan en la sesión de PowerShell en las situaciones siguientes:

- Cuando un runbook invoca un cmdlet desde un módulo.
- Cuando un runbook importa el módulo de forma explícita con el cmdlet Import-Module.
- Cuando un runbook importa el módulo de forma explícita con la instrucciónusing module. La instrucción using se admite a partir de Windows PowerShell 5.0 y admite clases y importación de tipos de enumeración.
- Cuando un runbook importa otro módulo dependiente.

Puede importar los módulos Az en la cuenta de Automation desde el portal de Azure. Dado que Az.Accounts es una dependencia para los otros módulos Az, asegúrese de importar este módulo antes que cualquier otro.

> **Nota:**
>
> Con la introducción de la compatibilidad con **PowerShell 7.1 (versión preliminar)**, la opción **Examinar galería** se ha actualizado con los cambios siguientes:

- **Examinar galería** está disponible en la hoja **Automatización de procesos**>**Módulos**.
- La página **Módulos** muestra dos columnas nuevas: la **versión del módulo** y la **versión el entorno de ejecución**.

1. Inicie sesión en el Azure portal.

2. Busque y seleccione **Cuentas de Automation**.

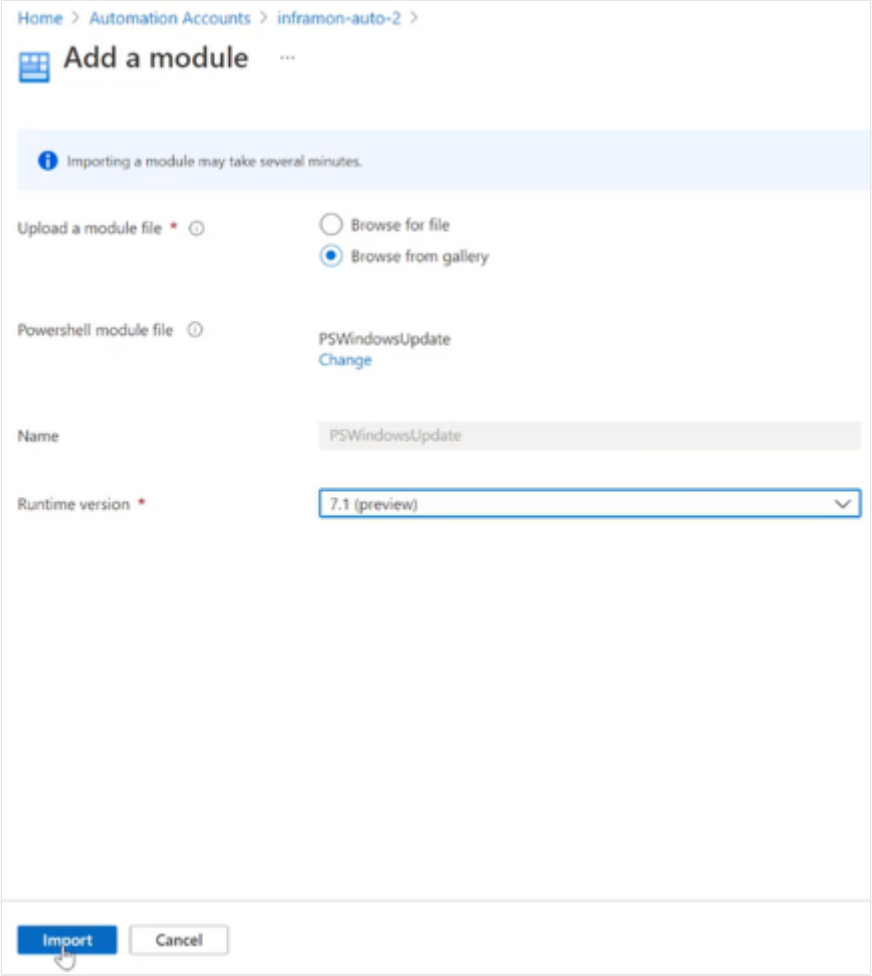
3. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.

4. En la cuenta de Automation, en **Recursos compartidos**, seleccione **Módulos**.

5. Seleccione **Agregar un módulo**. En la página **Agregar un módulo**, puede seleccionar cualquiera de las siguientes opciones:
   a. **Buscar un archivo**: esta opción selecciona un archivo de la máquina local.
   b. **Examinar desde la galería**: puede examinar y seleccionar un módulo existente de la galería.

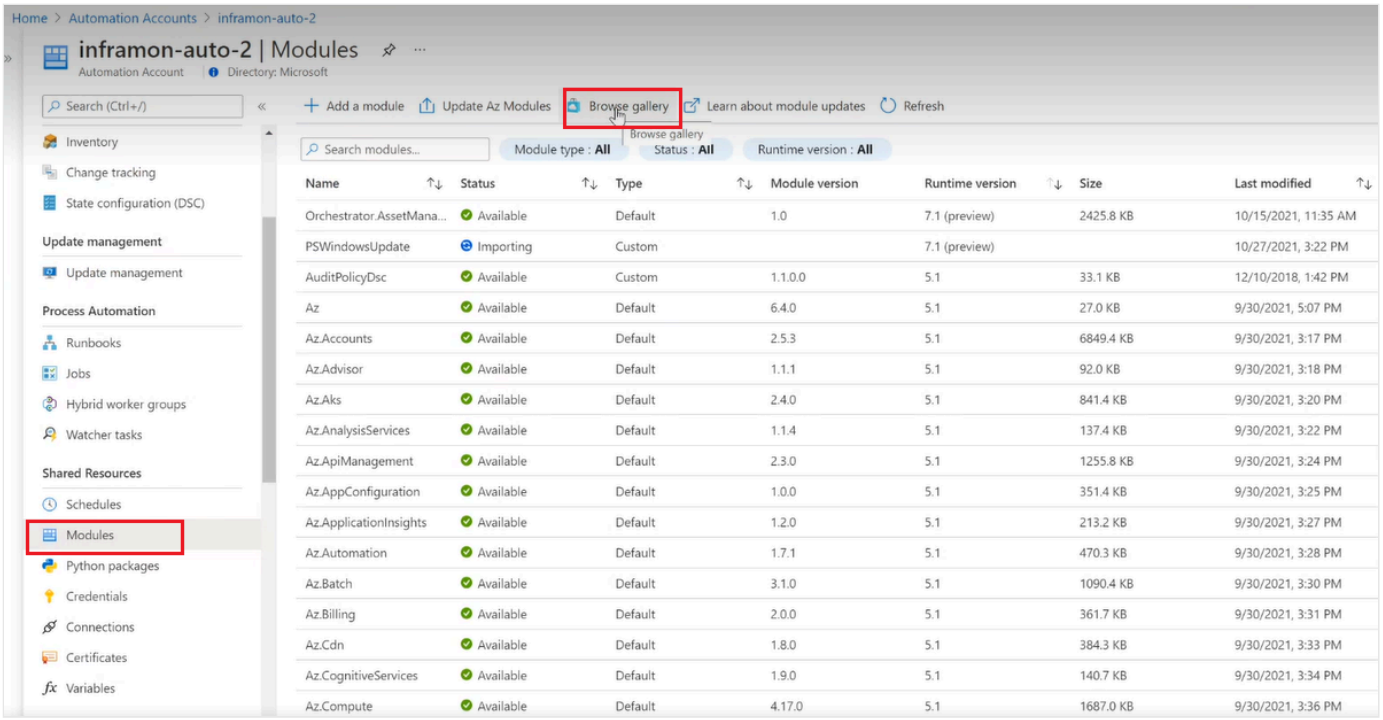
6. Haga clic en **Seleccionar** para seleccionar un módulo.

7. Seleccione **Versión del entorno de ejecución** y haga clic en **Importar**.



8. En la página **Módulos**, puede ver el módulo importado en la cuenta de Automation.

También puede realizar esta importación a través del Galería de PowerShell, buscando el módulo que se va a importar. Cuando encuentre el módulo, selecciónelo y elija la pestaña **Azure Automation**. Seleccione **Deploy to Azure Automation**.



Prueba de los runbooks

Una vez que los módulos Az se han importado en la cuenta de Automation, puede empezar a editar los runbooks y las configuraciones de DSC para usar los nuevos módulos. Una manera de probar la modificación de un runbook para usar los nuevos cmdlets consiste en utilizar el comando `Enable-AzureRmAlias -Scope Process` al principio del runbook. Al incorporar este comando al runbook, el script se puede ejecutar sin cambios.

Creación de módulos

Se recomienda seguir las consideraciones de esta sección al crear un módulo personalizado de PowerShell para su uso en Azure Automation. Para preparar el módulo para la importación, debe crear al menos un archivo **.dll** de módulo psd1, psm1 o PowerShell con el mismo nombre que la carpeta del módulo. A continuación, comprima la carpeta del módulo para que Azure Automation pueda importarla como un solo archivo. El paquete **.zip** debe tener el mismo nombre que la carpeta de módulo que contiene.

Para obtener más información sobre la creación de un módulo de PowerShell, consulte Cómo escribir un módulo de script de PowerShell.

Carpeta de versiones

El control de versiones de módulos en paralelo de PowerShell le permite usar más de una versión de un módulo en PowerShell. Esto puede ser útil si tiene scripts anteriores que se han probado y

solo funcionan con una versión determinada de un módulo de PowerShell, pero para otros necesita una versión más reciente del mismo módulo.

A fin de crear módulos de PowerShell para que contengan varias versiones, cree la carpeta del módulo y, después, cree una carpeta en su interior para cada versión del módulo que quiera usar. En el ejemplo siguiente, un módulo denominado *TestModule* proporciona dos versiones: 1.0.0 y 2.0.0.

dos

```
TestModule
   1.0.0
   2.0.0
```

Dentro de cada una de las carpetas de versión, copie los archivos **.dll** de módulo de PowerShell o .psm1 y .psd1 que componen un módulo en la carpeta de la versión correspondiente. Comprima la carpeta del módulo para que Azure Automation pueda importarla como un solo archivo .zip. Aunque Automation solo muestra una de las versiones del módulo importadas, si el paquete del módulo contiene versiones en paralelo del módulo, están disponibles para su uso en los runbooks o configuraciones de DSC.

Aunque Automation admite módulos que contienen versiones en paralelo dentro del mismo paquete, no admite el uso de varias versiones de un módulo en las importaciones de paquetes de módulo. Por ejemplo, puede importar el **módulo A**, que contiene las versiones 1 y 2 a la cuenta de Automation. Después, actualice el **módulo A** para incluir las versiones 3 y 4; cuando lo importe en la cuenta de Automation, solo las versiones 3 y 4 se pueden usar en cualquier runbook o configuración de DSC. Si necesita que todas las versiones (1, 2, 3 y 4) estén disponibles, se deben incluir en el archivo .zip.

Si va a usar otras versiones del mismo módulo entre runbooks, siempre debe declarar la versión que quiera utilizar en el runbook con el cmdlet `Import-Module` e incluir el parámetro `-RequiredVersion <version>`. Incluso si la versión que quiere usar es la más reciente. Esto se debe a que los trabajos de runbook se pueden ejecutar en el mismo espacio aislado. Si el espacio aislado ya ha cargado de forma explícita un módulo de un número de versión concreto, porque un trabajo anterior de ese espacio aislado lo haya indicado, los trabajos futuros de ese espacio aislado no cargarán automáticamente la versión más reciente de ese módulo. Esto se debe a que alguna versión ya está cargada en el espacio aislado.

Para un recurso de DSC, use el comando siguiente para especificar una versión concreta:

PowerShell

```
Import-DscResource -ModuleName "<ModuleName>" -ModuleVersion "<version>"
```

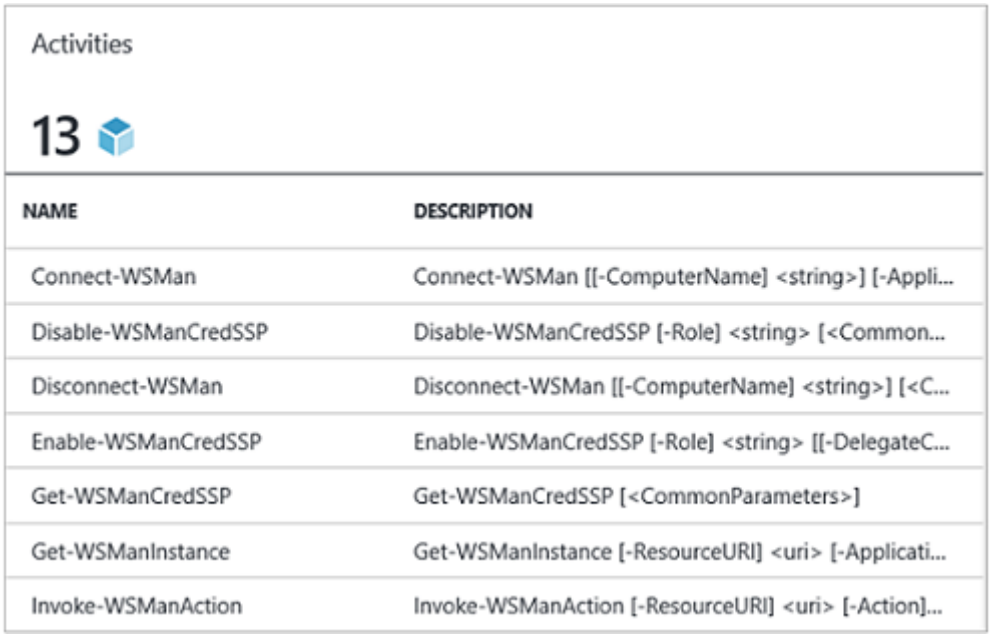
Información de ayuda

Incluya una sinopsis, una descripción y un identificador URI de ayuda para todos los cmdlets del módulo. En PowerShell, puede definir la información de ayuda de los cmdlets mediante el cmdlet `Get-Help`. En el ejemplo siguiente se muestra cómo definir una sinopsis y un identificador URI de ayuda en un archivo de módulo **.psm1**.

PowerShell

```
<#
     .SYNOPSIS
      Gets a Contoso User account
#>
function Get-ContosoUser {
[CmdletBinding](DefaultParameterSetName='UseConnectionObject', `
HelpUri='https://www.contoso.com/docs/information')]
[OutputType([String])]
param(
   [Parameter(ParameterSetName='UserAccount', Mandatory=true)]
   [ValidateNotNullOrEmpty()]
   [string]
   $UserName,

   [Parameter(ParameterSetName='UserAccount', Mandatory=true)]
   [ValidateNotNullOrEmpty()]
   [string]
   $Password,

   [Parameter(ParameterSetName='ConnectionObject', Mandatory=true)]
   [ValidateNotNullOrEmpty()]
   [Hashtable]
   $Connection
)

switch ($PSCmdlet.ParameterSetName) {
   "UserAccount" {
      $cred = New-Object -TypeName System.Management.Automation.PSCredential -
ArgumentList $UserName, $Password
      Connect-Contoso -Credential $cred
   }
   "ConnectionObject" {
      Connect-Contoso -Connection $Connection
  }
}
}
```

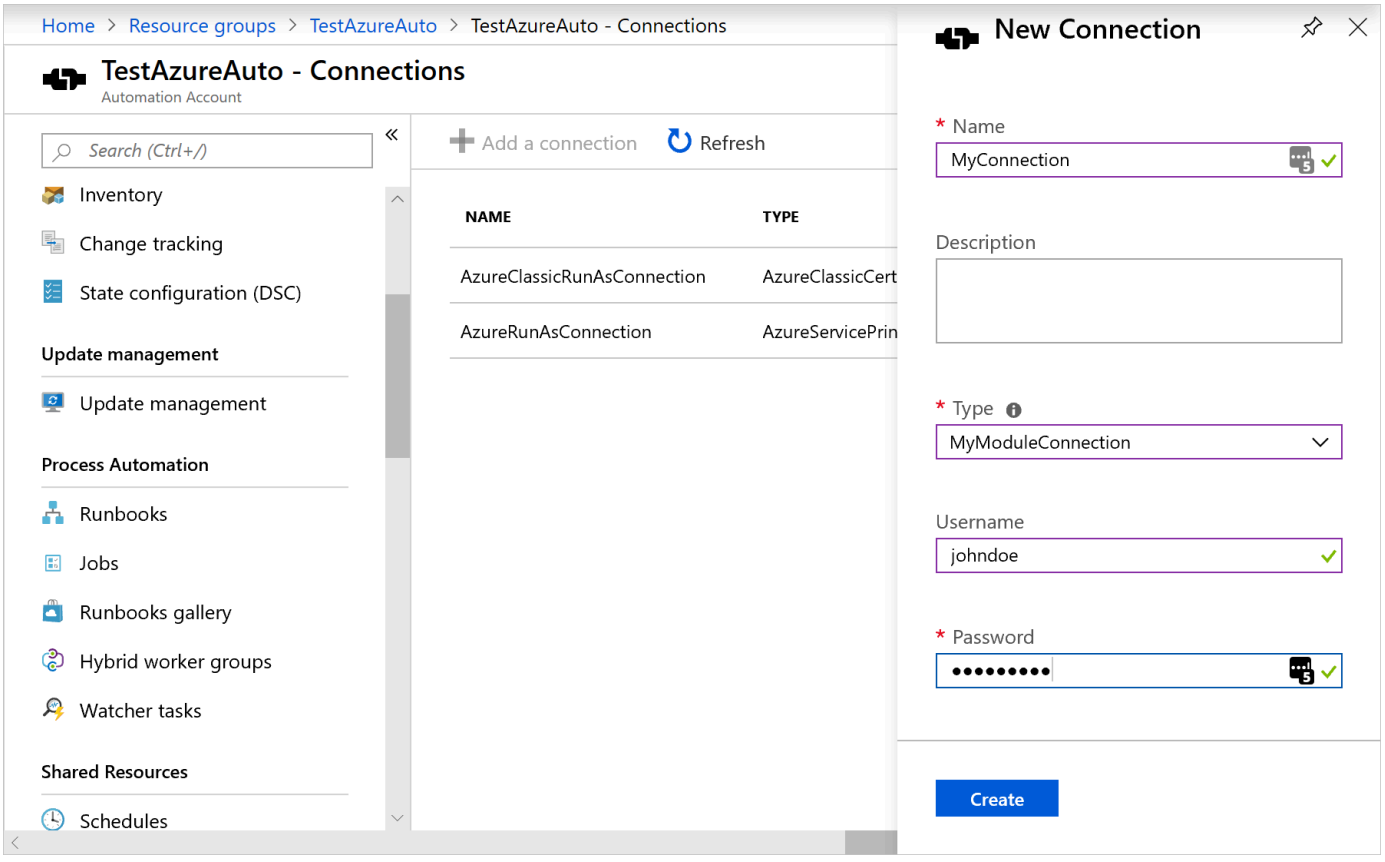
Proporcionar esta información permite mostrar el texto de ayuda mediante el cmdlet `Get-Help` en la consola de PowerShell. Este texto también se muestra en el portal de Azure.



NAME	DESCRIPTION
Connect-WSMan	Connect-WSMan [[-ComputerName] <string>] [-Appli...
Disable-WSManCredSSP	Disable-WSManCredSSP [-Role] <string> [<Common...
Disconnect-WSMan	Disconnect-WSMan [[-ComputerName] <string>] [<C...
Enable-WSManCredSSP	Enable-WSManCredSSP [-Role] <string> [[-DelegateC...
Get-WSManCredSSP	Get-WSManCredSSP [<CommonParameters>]
Get-WSManInstance	Get-WSManInstance [-ResourceURI] <uri> [-Applicati...
Invoke-WSManAction	Invoke-WSManAction [-ResourceURI] <uri> [-Action]...

Tipo de conexión

Si el módulo se conecta a un servicio externo, defina un tipo de conexión mediante un módulo de integración personalizado. Cada cmdlet del módulo debería aceptar una instancia de ese tipo de conexión (objeto de conexión) como parámetro. Los usuarios asignan parámetros del recurso de conexión a los parámetros correspondientes del cmdlet cada vez que llaman a un cmdlet.



El ejemplo de runbook siguiente usa un recurso de conexión de Contoso denominado `ContosoConnection` para acceder a los recursos de Contoso y devolver datos desde el servicio externo. En este ejemplo, los campos se asignan a las propiedades `UserName` y `Password` de un objeto `PSCredential` y, a continuación, se pasan al cmdlet.

PowerShell

```
$contosoConnection = Get-AutomationConnection -Name 'ContosoConnection'

$cred = New-Object -TypeName System.Management.Automation.PSCredential -ArgumentList
$contosoConnection.UserName, $contosoConnection.Password
Connect-Contoso -Credential $cred
}
```

Existe otra forma más fácil y eficaz de enfocar este comportamiento y consiste en pasar directamente el objeto de conexión al cmdlet:

PowerShell

```
$contosoConnection = Get-AutomationConnection -Name 'ContosoConnection'

Connect-Contoso -Connection $contosoConnection
}
```

Para habilitar un comportamiento como este en sus cmdlets, permítales que acepten directamente un objeto de conexión como parámetro en lugar de usar solamente campos de conexión en los parámetros. Normalmente, necesitará un conjunto de parámetros para cada uno de ellos, de forma que los usuarios que no utilizan Automation puedan llamar a los cmdlets sin necesidad de crear una tabla hash que actúe como objeto de conexión. El conjunto de parámetros `UserAccount` se usa para pasar las propiedades de los campos de conexión. `ConnectionObject` permite pasar directamente la conexión.

Tipo de salida

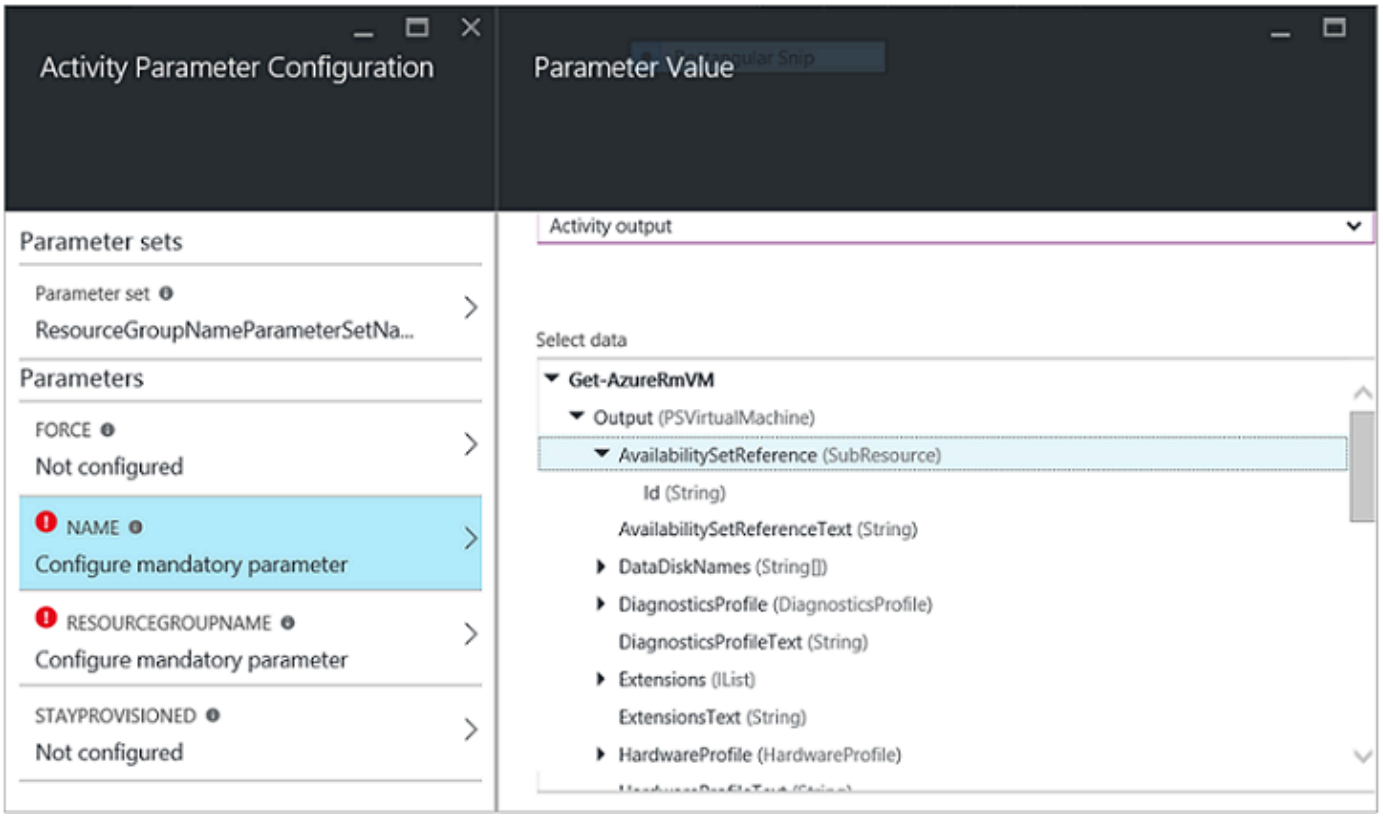
Defina el tipo de salida de todos los cmdlets del módulo. Definir un tipo de salida en un cmdlet permite que IntelliSense le ayude a determinar, en tiempo de diseño, las propiedades de salida del cmdlet durante la creación. Este procedimiento recomendado resulta especialmente útil durante la creación gráfica de un runbook, para el que la información en tiempo de diseño resulta esencial para facilitar la experiencia del usuario con el módulo.

Agregue `[OutputType([<MyOutputType>])]`, donde `MyOutputType` es un tipo válido. Para obtener más información sobre `OutputType`, consulte About Functions OutputTypeAttribute (Acerca de las

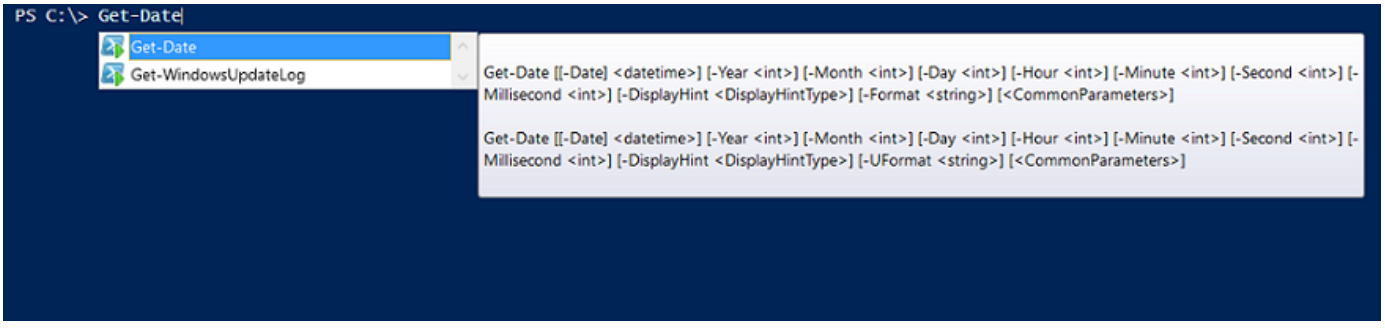
funciones OutputTypeAttribute). El siguiente código es un ejemplo de cómo agregar `OutputType` a un cmdlet:

PowerShell

```
function Get-ContosoUser {
[OutputType([String])]
param(
   [string]
   $Parameter1
)
# <script location here>
}
```



Este comportamiento es similar a la funcionalidad "type ahead" de la salida de los cmdlets del entorno del servicio de integración de PowerShell, sin necesidad de ejecutarla.



Estado del cmdlet

Cree todos los cmdlets del módulo sin estado. Pueden ejecutarse varios trabajos de runbook simultáneamente en el mismo elemento `AppDomain` y el mismo proceso y espacio aislado. Si hay algún estado compartido en esos niveles, los trabajos pueden afectarse entre sí. Este comportamiento puede provocar problemas intermitentes y difíciles de diagnosticar. Este es un ejemplo de lo que no hacer:

PowerShell

```
$globalNum = 0
function Set-GlobalNum {
   param(
       [int] $num
   )

   $globalNum = $num
}
function Get-GlobalNumTimesTwo {
   $output = $globalNum * 2

   $output
}
```

Dependencia de los módulos

Asegúrese de que el módulo está totalmente contenido en un paquete que se puede copiar mediante XCopy. Los módulos de Automation se distribuyen en los espacios aislados de Automation cuando los runbooks se ejecutan. Los módulos deben funcionar de forma independiente con respecto al host que los ejecuta.

Debe ser capaz de comprimir y mover un paquete de módulos y conseguir que funcione con normalidad cuando se importe en el entorno de PowerShell de otro host. Para que esto ocurra, asegúrese de que el módulo no dependa de ningún archivo que esté fuera de la carpeta del módulo que se comprime cuando el módulo se importa en Automation.

El módulo no debe depender de ninguna configuración de registro única en un host. Un ejemplo es la configuración que se realiza cuando se instala un producto.

Rutas de acceso a los archivos del módulo

Asegúrese de que todos los archivos del módulo tienen rutas de acceso con menos de 140 caracteres. Las rutas de acceso con más de 140 caracteres provocan problemas de importación de runbooks. Automation no puede importar ningún archivo con un tamaño de ruta de acceso superior a 140 caracteres en la sesión de PowerShell con `Import-Module`.

Importación de módulos

En esta sección se definen varias formas de importar un módulo en la cuenta de Automation.

Importación de módulos en el portal de Azure

Para importar un módulo en el portal de Azure:

1. En el portal, busque y seleccione **Cuentas de Automation**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. En **Recursos compartidos**, seleccione **Módulos**.
4. Seleccione **Agregar un módulo**.
5. Seleccione el archivo **.zip** que contiene el módulo.
6. Seleccione **Aceptar** para iniciar el proceso de importación.

Importación de módulos mediante PowerShell

Puede usar el cmdlet New-AzAutomationModule para importar un módulo en la cuenta de Automation. El cmdlet toma una dirección URL para un paquete .zip del módulo.

Azure PowerShell

```
$moduleName = "<ModuleName>"
$contentLinkUri = "<ModuleUri>"
$runtimeVersion = "<RuntimeVersion>" # 5.1 or 7.2
$resourceGroupName = "<ResourceGroupName>"
$automationAccountName = "<AutomationAccountName>"
New-AzAutomationModule -Name $moduleName -RuntimeVersion $runtimeVersion -
ContentLinkUri $contentLinkUri -ResourceGroupName $resourceGroupName -
AutomationAccountName $automationAccountName
```

También puede usar el mismo cmdlet para importar un módulo desde el Galería de PowerShell directamente. Asegúrese de obtener `ModuleName` y `ModuleVersion` del Galería de PowerShell.

Azure PowerShell

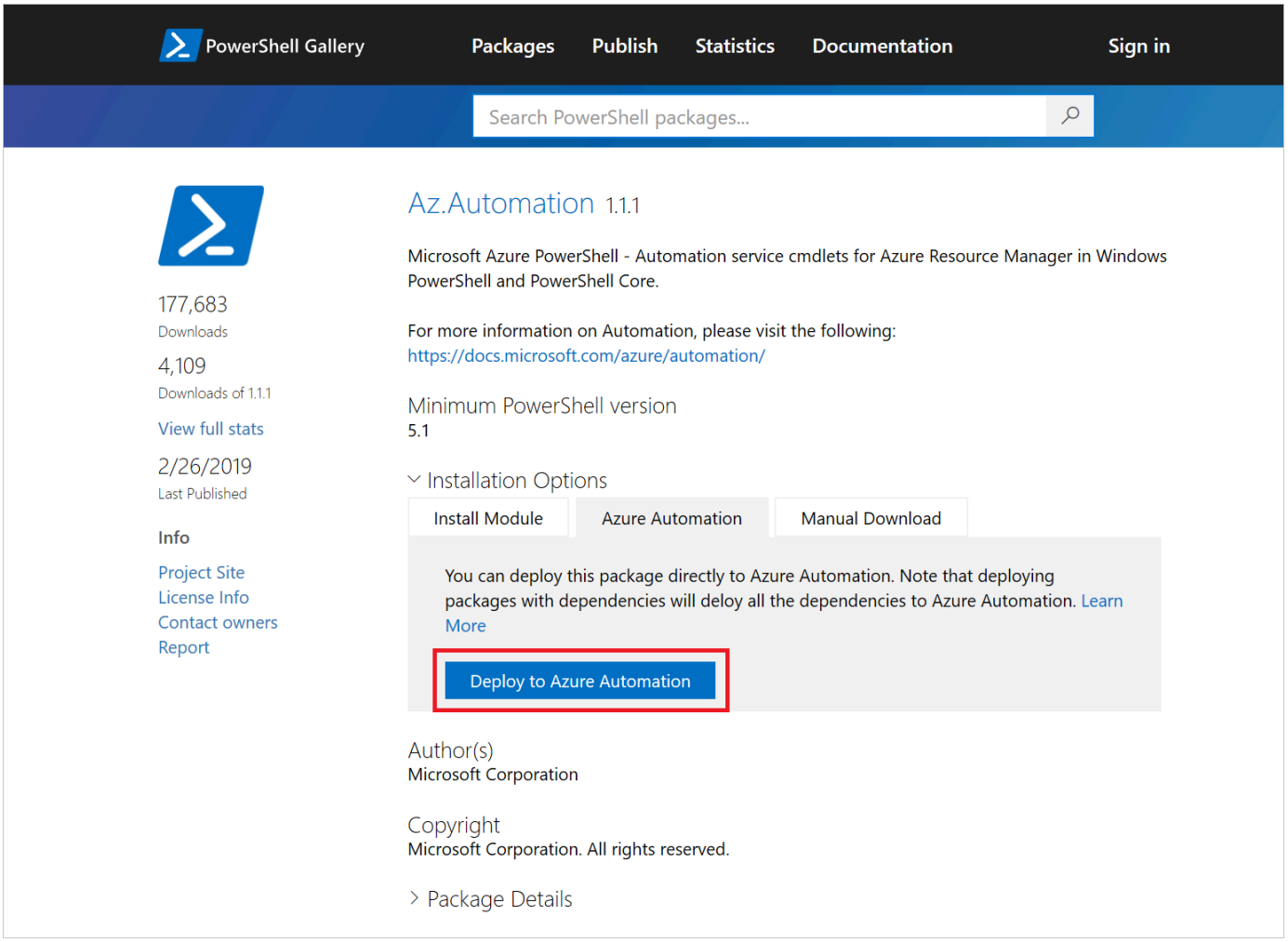
```
$moduleName = "<ModuleName>"
$moduleVersion = "<ModuleVersion>"
$runtimeVersion = "<RuntimeVersion>" # 5.1 or 7.2
$resourceGroupName = "<ResourceGroupName>"
$automationAccountName = "<AutomationAccountName>"
New-AzAutomationModule -AutomationAccountName $automationAccountName -RuntimeVersion
$runtimeVersion -ResourceGroupName $resourceGroupName -Name $moduleName -
ContentLinkUri
"https://www.powershellgallery.com/api/v2/package/$moduleName/$moduleVersion"
```

Importación de módulos desde el Galería de PowerShell

Puede importar Galería de PowerShell módulos directamente desde la Galería o desde la cuenta de Automation.

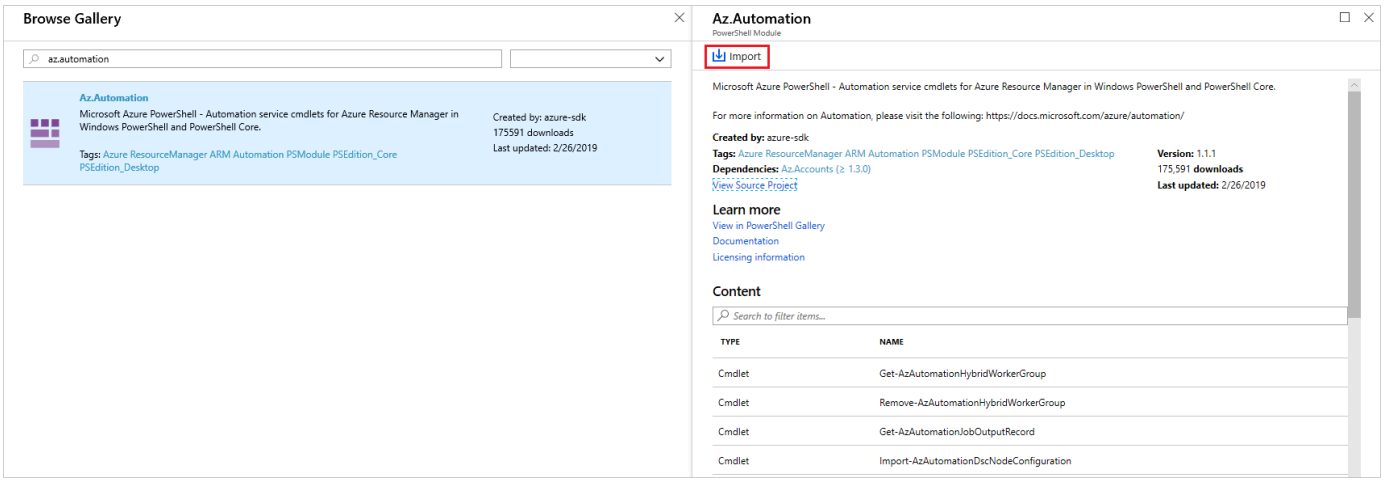
Para importar un módulo directamente desde el Galería de PowerShell:

1. Vaya a https://www.powershellgallery.com y busque el módulo que desea importar.
2. En **Opciones de instalación**, en la pestaña **Azure Automation**, seleccione **Desplegar en Azure Automation**. Esta acción abre el portal de Azure.
3. En la página Importar, seleccione su cuenta de Automation y seleccione **Aceptar**.



Para importar un módulo de Galería de PowerShell directamente desde la cuenta de Automation:

1. En el portal, busque y seleccione **Cuentas de Automation**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. En **Recursos compartidos**, seleccione **Módulos**.
4. Seleccione **Examinar galería** y busque el módulo en la galería.
5. Seleccione el módulo que quiere importar y, después, haga clic en **Importar**.
6. Seleccione **Aceptar** para iniciar el proceso de importación.



Eliminación de los módulos

Si tiene problemas con un módulo o necesita revertir a una versión anterior de un módulo, puede eliminarlo de la cuenta de Automation. No se pueden eliminar las versiones originales de los módulos predeterminados que se importan al crear una cuenta de Automation. Si el módulo que desea eliminar es una versión más reciente de uno de los módulos predeterminados, se revertirá a la versión que se instaló con la cuenta de Automation. En caso contrario, se quitará cualquier módulo que se elimine de la cuenta de Automation.

Eliminación de módulos en el portal de Azure

Para quitar un módulo en el portal de Azure:

1. En el portal, busque y seleccione **Cuentas de Automation**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. En **Recursos compartidos**, seleccione **Módulos**.
4. Seleccione el módulo que quiera quitar.
5. En la página Módulo, seleccione **Eliminar**. Si este módulo es uno de los módulos predeterminados, se revertirá a la versión que existía cuando se creó la cuenta de Automation.

Eliminación de módulos mediante PowerShell

Para quitar un módulo a través de PowerShell, ejecute el siguiente comando:

Azure PowerShell

```
Remove-AzAutomationModule -Name <moduleName> -AutomationAccountName
<automationAccountName> -ResourceGroupName <resourceGroupName>
```

Pasos siguientes

- Para obtener más información sobre el uso de módulos de Azure PowerShell, consulte Get started with Azure PowerShell.
- Para obtener más información sobre cómo crear módulos de PowerShell, consulte Escribir un módulo de PowerShell Windows.
- Para solucionar problemas relacionados con recursos compartidos, como Azure PowerShell, Python, cmdlet interno y módulos personalizados en Azure Automation, consulte Troubleshoot Azure Automation problemas de recursos compartidos.

Last updated on 18/05/2026

Actualización de módulos de Azure PowerShell en Automation

Los módulos de PowerShell más comunes se proporcionan de forma predeterminada en cada cuenta de Automation. Consulte Módulos predeterminados. A medida que el equipo de Azure actualiza los módulos de Azure periódicamente, se pueden producir cambios con los cmdlets incluidos. Estos cambios, como el cambio de nombre de un parámetro o dejar en desuso un cmdlet por completo, pueden afectar negativamente a los runbooks.

> **Nota:**
>
> No se pueden eliminar módulos globales, que son los módulos que Automation proporciona de forma integrada.

> **Importante**
>
> La nueva experiencia del entorno de runtime permite administrar módulos y paquetes configurando el entorno de ejecución del trabajo. En la nueva experiencia, las hojas Módulos y Paquetes no están disponibles. Para administrar los módulos y paquetes, consulte Administración del entorno de runtime y los runbooks asociados.

Configuración de una cuenta de Automation

Para evitar que incidan en los runbooks y los procesos que automatizan, asegúrese de probarlos y validarlos cuando realice actualizaciones. Si no tiene una cuenta de Automation dedicada pensada para este propósito, considere la posibilidad de crear una para que pueda probar muchos escenarios diferentes durante el desarrollo de los runbooks. Estas pruebas deben incluir cambios iterativos, por ejemplo, la actualización de los módulos de PowerShell.

Asegúrese de que la cuenta de Automation ha agregado una identidad administrada asignada por el sistema o una identidad administrada asignada por el usuario.

Si desarrolla sus scripts de forma local, se recomienda tener localmente las mismas versiones del módulo que tiene en la cuenta de Automation al realizar pruebas, a fin de asegurarse de que

obtendrá los mismos resultados. Una vez validados los resultados y aplicados los cambios necesarios, puede mover los cambios a producción.

> **Nota:**
>
> Es posible que una cuenta de Automation nueva no contenga los módulos más recientes.

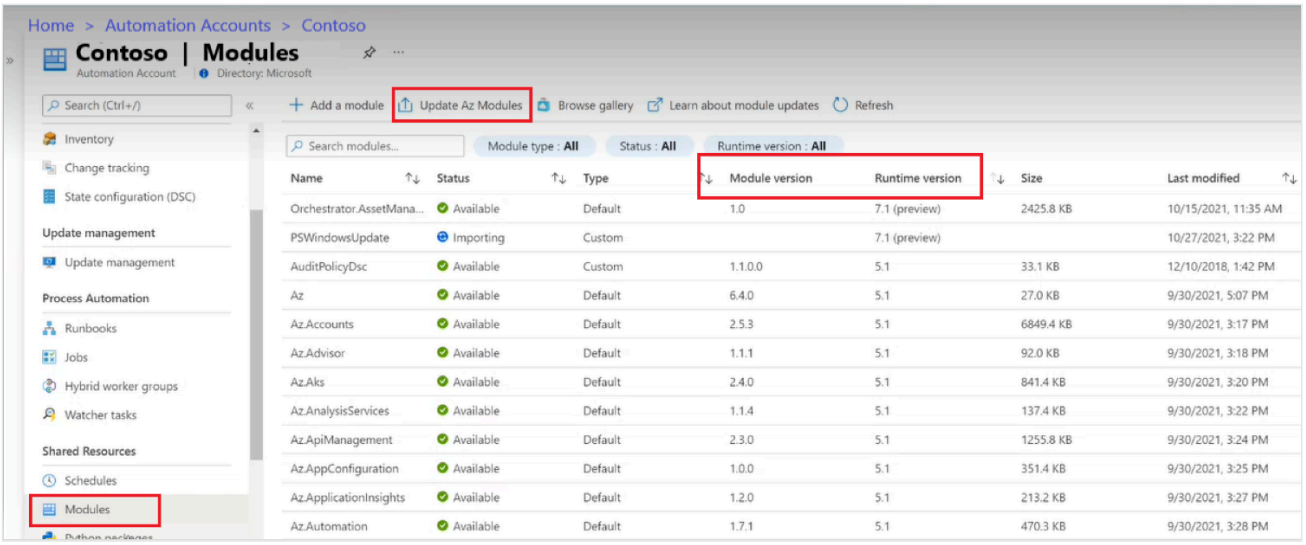
Actualización de módulos de Azure

Las siguientes secciones explican cómo puede actualizar los módulos Az a través del **portal** (recomendado) o a través del libro de ejecución.

Actualización de módulos de Az a través del portal

Actualmente, la actualización de módulos de Azure solo está disponible mediante el portal. Las actualizaciones mediante PowerShell y la plantilla de ARM estarán disponibles en el futuro. Solo se actualizarán los módulos de Azure predeterminados al realizar los pasos siguientes:

1. Inicie sesión en el portal Azure y vaya a la cuenta de Automation.
2. En **Recursos compartidos**, seleccione **Módulos**.
3. Selecciona **Actualización de módulos de Azure**.
4. Seleccione el **módulo que se va a actualizar**. De forma predeterminada, se mostrará el módulo de Azure.
5. En la lista desplegable, seleccione **Versión del módulo** y **Versión de Runtype**.
6. Seleccione **Actualizar** para actualizar el módulo de Azure a la versión que ha seleccionado. En la página **Módulos**, puede ver la lista tal como se muestra a continuación:



Si selecciona una versión anterior a la versión del módulo de Azure existente importada en la cuenta de Automation, la operación de actualización realizará una reversión a la versión inferior seleccionada.

Para comprobar la operación de actualización, compruebe la propiedad Versión del módulo y Estado de los módulos actualizados que se muestran en la lista de **módulos** en **Módulos de PowerShell**.

El equipo de Azure actualizará periódicamente la versión del módulo y proporcionará una opción para actualizar los módulos Az predeterminados seleccionando la versión del módulo en la lista desplegable.

Actualización de módulos de Az mediante runbook

Para actualizar los módulos de Azure en la cuenta de Automation:

1. Utilice el runbook Update-AutomationAzureModulesForAccount, disponible como código abierto.
2. Descargue desde el repositorio de GitHub para empezar a usar este runbook para actualizar los módulos de Azure.
3. Puede importarlo en la cuenta de Automation o ejecutarlo como un script. Para aprender a importar un runbook en la cuenta de Automation, consulte Importación de un runbook.

> **Nota:**
>
> Se recomienda actualizar los módulos de Az a través de Azure portal. También puede hacerlo mediante el `Update-AutomationAzureModulesForAccount` script, disponible como código abierto y proporcionado como referencia. Sin embargo, en caso de cualquier fallo

del runbook, debe modificar los parámetros según sea necesario o debuguear el script según el escenario.

El runbook **Update-AutomationAzureModulesForAccount** admite la actualización de los módulos Azure, AzureRM y Az de forma predeterminada. Revise Update Azure modules runbook README para obtener más información sobre cómo actualizar Az. Módulos de Automatización con este libro de ejecución. Hay otros factores importantes que debe tener en cuenta al usar los módulos Az en su cuenta de Automation. Para más información, consulte Administrar módulos en Azure Automation.

Uso del código del runbook de actualización como un script normal de PowerShell

Puede usar el código de runbook como un script normal de PowerShell en lugar de un runbook. Para ello, inicie sesión en Azure con el cmdlet Connect-AzAccount y, a continuación, pase primero `-Login $false` al script.

Utilice el libro de procedimientos de actualización en nubes soberanas

Para usar este runbook en nubes soberanas, use el parámetro `AzEnvironment` para pasar el entorno correcto al runbook. Los valores aceptables son AzureCloud (Azure nube pública), AzureChinaCloud, AzureGermanCloud y AzureUSGovernment. Estos valores se pueden recuperar con `Get-AzEnvironment | select Name`. Si no pasa un valor a este cmdlet, el runbook toma como valor predeterminado AzureCloud.

Utilice el runbook de actualización para actualizar una versión específica del módulo

Si quiere usar una versión específica del módulo de Azure PowerShell en lugar del módulo más reciente disponible en el Galería de PowerShell, pase estas versiones al parámetro opcional `ModuleVersionOverrides` del runbook **Update-AutomationAzureModulesForAccount**. Para obtener ejemplos, consulte el runbook Update-AutomationAzureModulesForAccount.ps1. Los módulos de Azure PowerShell que no se mencionan en el parámetro `ModuleVersionOverrides` se actualizan automáticamente con las versiones más recientes del módulo en Galería de PowerShell. Si no pasa nada al parámetro `ModuleVersionOverrides`, todos los módulos se actualizan con las versiones más recientes de los módulos en la Galería de PowerShell. Este comportamiento es el mismo para el botón **Update Azure Modules** en el portal de Azure.

Pasos siguientes

- Para obtener más información sobre el uso de módulos, consulte Administrar módulos en Azure Automation.
- Para obtener información sobre el runbook de actualización, consulte el Runbook de actualización de los módulos de Azure.
- Para solucionar problemas al actualizar los módulos de Azure PowerShell administrados como recursos compartidos en Azure Automation, consulte Troubleshoot Azure Automation problemas de recursos compartidos.

Last updated on 20/04/2026

Administración de programaciones en Azure Automation

Para programar un runbook en Azure Automation para que se inicie en un momento determinado, vincúlelo a una o más programaciones. Es posible configurar una programación para que se ejecute una vez o según una programación periódica diaria o por hora para los runbooks en Azure Portal. También las puede programar para que se ejecuten una vez a la semana, una vez al mes, en días específicos de la semana o del mes, o bien en un día determinado del mes. Un runbook puede vincularse a varias programaciones y una programación puede tener varios runbooks vinculados a ella.

> **Nota:**
>
> Azure Automation admite el horario de verano y lo programa de forma adecuada para las operaciones de automatización.

> **Nota:**
>
> Las programaciones no están habilitadas actualmente para las configuraciones de DSC de Azure Automation.

Cmdlets de PowerShell usados para acceder a programaciones

Los cmdlets de la tabla siguiente permiten crear y administrar programaciones de Automation con PowerShell. Se suministran como un componente de los módulos Az.

Expandir tabla

Cmdlets	Descripción
Get-AzAutomationSchedule	Recupera una programación.
Get-AzAutomationScheduledRunbook	Recupera runbooks programados.
New-AzAutomationSchedule	Crea una nueva programación.
Register-AzAutomationScheduledRunbook	Asocia un runbook con una programación.
Remove-AzAutomationSchedule	Quita una programación.

Cmdlets	Descripción
Set-AzAutomationSchedule	Establece las propiedades de una programación existente.
Unregister-AzAutomationScheduledRunbook	Anula la asociación de un runbook con una programación.

Crear una programación

Puede crear una nueva programación para los runbooks desde Azure Portal, con PowerShell o mediante una plantilla Azure Resource Manager (ARM). Para evitar que afecte a los runbooks y a los procesos que automatizan, primero debe probar los runbooks que tengan programaciones vinculadas con una cuenta de Automation dedicada para realizar pruebas. Una prueba valida que los runbooks programados sigan funcionando correctamente. Si ve un problema, puede solucionarlo y aplicar los cambios necesarios antes de migrar la versión de runbook actualizada a producción.

> ⓘ **Nota:**
>
> La cuenta de Automation no obtiene automáticamente las nuevas versiones de los módulos, a menos que se hayan actualizado manualmente seleccionando la opción Actualizar módulos de Azure en **Módulos**. Azure Automation utiliza los módulos más recientes en su cuenta de Automation al ejecutar un nuevo trabajo programado.

Creación de una programación nueva en Azure Portal

1. En la cuenta de Automation, en el panel izquierdo, seleccione **Programas** en **Recursos compartidos**.

2. En la página **Programas**, seleccione **Agregar una programación**.

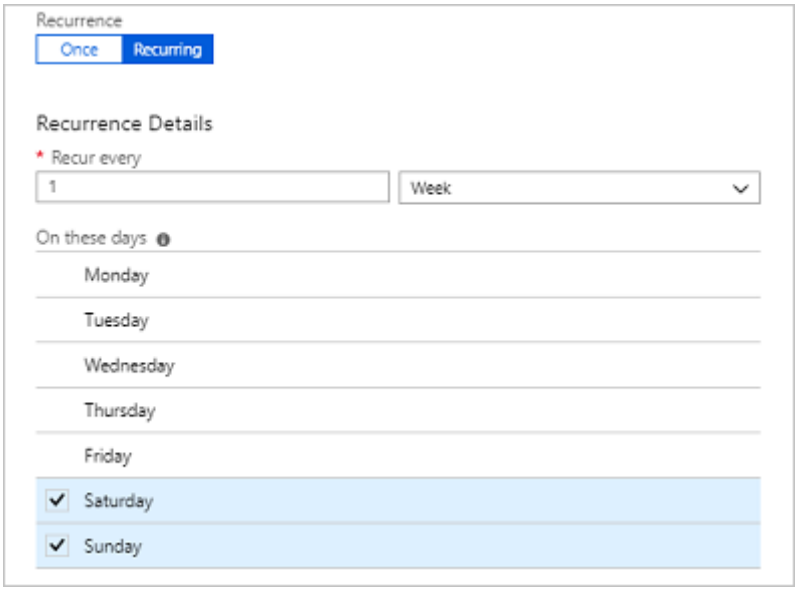
3. En la página **Nueva programación**, escriba un nombre y, opcionalmente, una descripción para la nueva programación.

   > ⓘ **Nota:**
   >
   > Las programaciones de Automation no admiten actualmente el uso de caracteres especiales en el nombre de la programación.

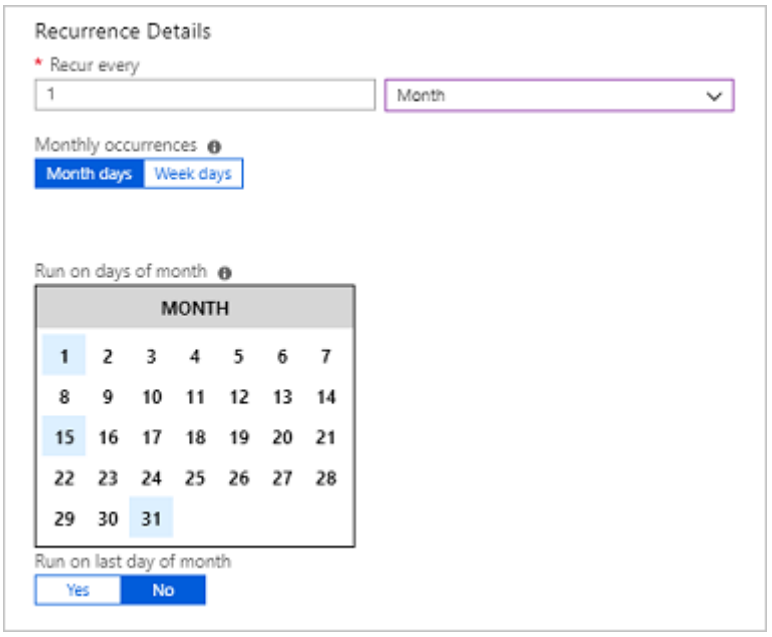
4. Elija si la programación se ejecuta una vez o de forma periódica seleccionando **Una vez** o **Periódico**. Si elige **Una vez**, especifique una hora de inicio y seleccione **Crear**. Si

selecciona **Periódica**, especifique una hora de inicio. Para **Repetir cada**, seleccione la frecuencia con la que desea que se repita el runbook. Seleccione por hora, día, semana o mes.

- Si selecciona **Semana**, se presentan los días de la semana entre los que puede elegir. Seleccione tantos días como quiera. La primera ejecución de la programación se realizará en el primer día seleccionado después de la hora de inicio. Por ejemplo, para elegir una programación durante el fin de semana, elija sábado y domingo.



- Si selecciona **Mes**, tendrá diferentes opciones. Para la opción **Repeticiones mensuales**, seleccione **Días del mes** o **Días de la semana**. Si selecciona **Días del mes**, aparece un calendario para que pueda elegir tantos días como desee. Si elige una fecha como el día 31 que no existe en el mes actual, no se ejecutará la programación. Si quiere que la programación se ejecute en el último día, seleccione **Sí** en **Ejecutar en el último día del mes**. Si selecciona **Días de la semana**, aparecerá la opción **Repetir cada**. Elija **Primero**, **Segundo**, **Tercero**, **Cuarto** o **Último**. Por último, elija un día en el que realizar la repetición.



5. Cuando haya acabado, seleccione **Crear**.

Creación de una nueva programación con PowerShell

Use el cmdlet New-AzAutomationSchedule para crear programaciones. Especifique la hora de inicio de la programación y la frecuencia con que se debe ejecutar. En los ejemplos siguientes se muestra cómo crear muchos escenarios de programación distintos.

> ⓘ **Nota:**
>
> Las programaciones de Automation no admiten actualmente el uso de caracteres especiales en el nombre de la programación.

Creación de una programación única

En el siguiente ejemplo se crea una programación única.

Azure PowerShell

```
$TimeZone = ([System.TimeZoneInfo]::Local).Id
New-AzAutomationSchedule -AutomationAccountName "ContosoAutomation" -Name
"Schedule01" -StartTime "23:00" -OneTime -ResourceGroupName "ResourceGroup01" -
TimeZone $TimeZone
```

Creación de una programación recurrente

En el ejemplo siguiente se muestra cómo crear una programación periódica que se ejecuta cada día a la 1:00 p.m. durante un año.

Azure PowerShell

```
$StartTime = Get-Date "13:00:00"
$EndTime = $StartTime.AddYears(1)
New-AzAutomationSchedule -AutomationAccountName "ContosoAutomation" -Name
"Schedule02" -StartTime $StartTime -ExpiryTime $EndTime -DayInterval 1 -
ResourceGroupName "ResourceGroup01"
```

Creación de una programación semanal recurrente

En el ejemplo siguiente se muestra cómo crear una programación semanal que se ejecute solo en días laborables.

Azure PowerShell

```
$StartTime = (Get-Date "13:00:00").AddDays(1)
[System.DayOfWeek[]]$WeekDays = @([System.DayOfWeek]::Monday..
[System.DayOfWeek]::Friday)
New-AzAutomationSchedule -AutomationAccountName "ContosoAutomation" -Name
"Schedule03" -StartTime $StartTime -WeekInterval 1 -DaysOfWeek $WeekDays -
ResourceGroupName "ResourceGroup01"
```

Creación de una programación semanal recurrente para los fines de semana

En el ejemplo siguiente se muestra cómo crear una programación semanal que se ejecute solo los fines de semana.

Azure PowerShell

```
$StartTime = (Get-Date "18:00:00").AddDays(1)
[System.DayOfWeek[]]$WeekendDays = @([System.DayOfWeek]::Saturday,
[System.DayOfWeek]::Sunday)
New-AzAutomationSchedule -AutomationAccountName "ContosoAutomation" -Name "Weekends
6PM" -StartTime $StartTime -WeekInterval 1 -DaysOfWeek $WeekendDays -
ResourceGroupName "ResourceGroup01"
```

Creación de una programación periódica para el primer día, el decimoquinto día y el último día del mes

En el ejemplo siguiente se muestra cómo crear una programación periódica que se ejecuta el primer día, el decimoquinto día y el último día de un mes.

Azure PowerShell

```
$StartTime = (Get-Date "18:00:00").AddDays(1)
New-AzAutomationSchedule -AutomationAccountName "TestAzureAuto" -Name "1st, 15th and Last" -StartTime $StartTime -DaysOfMonth @("One", "Fifteenth", "Last") -ResourceGroupName "TestAzureAuto" -MonthInterval 1
```

Creación de una programación con una plantilla de Resource Manager

En este ejemplo, se usa una plantilla de Resource Manager de Automation (ARM) que crea una nueva programación de trabajo. Para obtener información general sobre esta plantilla para administrar las programaciones de trabajos de Automation, consulte la referencia de plantillas de Microsoft.Automation automationAccounts/jobSchedules.

Copie este archivo de plantilla en un editor de texto:

JSON

```
{
  "name": "5d5f3a05-111d-4892-8dcc-9064fa591b96",
  "type": "Microsoft.Automation/automationAccounts/jobSchedules",
  "apiVersion": "2015-10-31",
  "properties": {
    "schedule": {
      "name": "scheduleName"
    },
    "runbook": {
      "name": "runbookName"
    },
    "runOn": "hybridWorkerGroup",
    "parameters": {}
  }
}
```

Edite los siguientes valores de parámetro y guarde la plantilla como un archivo JSON:

- Nombre del objeto de programación de trabajo: se usa un GUID (identificador único global) como nombre del objeto de programación de trabajo.

  > **Importante**
  >
  > Para cada programación de trabajo implementada con una plantilla de ARM, el GUID debe ser único. Incluso si va a volver a programar una programación existente, deberá cambiar el GUID. Esto se aplica incluso si ha eliminado previamente una programación de trabajo existente creada con la misma plantilla. Si se reutiliza el mismo GUID, se producirá un error en la implementación.

> Hay servicios en línea que pueden generar un nuevo GUID por usted, como este generador de GUID en línea gratuito.

- Nombre de programación: representa el nombre de la programación de trabajos de automatización que se vinculará al runbook especificado.
- Nombre del runbook: representa el nombre del runbook de Automation con el que se asociará la programación del trabajo.

Una vez guardado el archivo, puede crear la programación del trabajo del runbook con el siguiente comando de PowerShell. El comando usa el parámetro `TemplateFile` para especificar la ruta de acceso y el nombre de archivo de la plantilla.

PowerShell

```
New-AzResourceGroupDeployment -ResourceGroupName "ContosoEngineering" -TemplateFile
"<path>\RunbookJobSchedule.json"
```

Vinculación de una programación a un runbook

Un runbook puede vincularse a varias programaciones y una programación puede tener varios runbooks vinculados a ella. Si un runbook tiene parámetros, puede proporcionar valores para ellos. Debe proporcionar valores para los parámetros obligatorios y también puede proporcionar valores para los parámetros opcionales. Estos valores se usan cada vez que esta programación inicia el runbook. Puede adjuntar el mismo runbook a otra programación y especificar distintos valores de parámetros.

Vinculación de una programación a un runbook con Azure Portal

1. En Azure Portal, en la cuenta de Automation, seleccione **Runbooks** en **Automatización de procesos**.
2. Seleccione el nombre del runbook que se va a programar.
3. Si el runbook no está vinculado actualmente a una programación, se le ofrecerá la opción de crear una o de vincularlo a una existente.
4. Si el runbook tiene parámetros, puede seleccionar la opción **Modificar la configuración de ejecución (Predeterminada: Azure)** y aparece el panel **Parámetros**. Aquí puede escribir la información de los parámetros.

Vinculación de una programación a un runbook con PowerShell

Use el cmdlet Register-AzAutomationScheduledRunbook para vincular una programación. Puede especificar valores para los parámetros del runbook con el parámetro Parameters. Para más información sobre cómo especificar valores de parámetro, consulte Inicio de un runbook en Azure Automation. En el ejemplo siguiente se muestra cómo vincular una programación a un runbook mediante un cmdlet de Azure Resource Manager con parámetros.

Azure PowerShell

```
$automationAccountName = "MyAutomationAccount"
$runbookName = "Test-Runbook"
$scheduleName = "Sample-DailySchedule"
$params = @{"FirstName"="Joe";"LastName"="Smith";"RepeatCount"=2;"Show"=$true}
Register-AzAutomationScheduledRunbook -AutomationAccountName $automationAccountName
`
-Name $runbookName -ScheduleName $scheduleName -Parameters $params `
-ResourceGroupName "ResourceGroup01"
```

Programación de runbooks para que se ejecuten con más frecuencia

El intervalo de mayor frecuencia para el que se puede configurar una programación en Azure Automation es una hora. Si necesita programaciones que se ejecuten con mayor frecuencia, tiene dos opciones:

- Crear un webhook para el runbook y usar Azure Logic Apps para llamar al webhook. Azure Logic Apps ofrece una granularidad más específica para definir una programación.
- Crear cuatro programaciones que comiencen todas con una diferencia de 15 minutos entre sí y que se ejecuten una vez cada hora. Este escenario permite que el runbook se ejecute cada 15 minutos con las distintas programaciones.

Deshabilitación de una programación

Cuando se deshabilita una programación, los runbooks vinculados a ella ya no se ejecutan en ella. Puede deshabilitar una programación manualmente o establecer una fecha de expiración para las programaciones con frecuencia al crearlas. Cuando se alcanza la fecha de expiración, la programación se deshabilita.

Deshabilitación de una programación desde Azure Portal

1. En la cuenta de Automation, en el panel izquierdo, seleccione **Programas** en **Recursos compartidos**.
2. Seleccione el nombre de una programación para abrir el panel de detalles.
3. Cambie **Habilitado** a **No**.

> ⓘ **Nota:**
>
> Si quiere deshabilitar una programación con una hora de inicio en el pasado, debe cambiar la fecha de inicio a un momento en el futuro antes de guardarla.

Deshabilitación de una programación con PowerShell

Use el cmdlet Set-AzAutomationSchedule para cambiar las propiedades de una programación existente. Para deshabilitar la programación, especifique False para el parámetro `IsEnabled`.

En el ejemplo siguiente se muestra cómo deshabilitar una programación para un runbook mediante un cmdlet de Azure Resource Manager.

Azure PowerShell

```
$automationAccountName = "MyAutomationAccount"
$scheduleName = "Sample-MonthlyDaysOfMonthSchedule"
Set-AzAutomationSchedule -AutomationAccountName $automationAccountName `
-Name $scheduleName -IsEnabled $false -ResourceGroupName "ResourceGroup01"
```

Eliminación de una programación

Cuando esté preparado para quitar programaciones, puede usar Azure Portal o PowerShell. Recuerde que solo puede quitar una programación que se haya deshabilitado, tal como se describe en la sección anterior.

Eliminación de una programación con Azure Portal

1. En la cuenta de Automation, en el panel izquierdo, seleccione **Programas** en **Recursos compartidos**.
2. Seleccione el nombre de una programación para abrir el panel de detalles.
3. Haga clic en **Eliminar**.

Eliminación de una programación con PowerShell

Puede usar el cmdlet `Remove-AzAutomationSchedule` como se muestra a continuación para eliminar una programación existente.

Azure PowerShell

```
$automationAccountName = "MyAutomationAccount"
$scheduleName = "Sample-MonthlyDaysOfMonthSchedule"
Remove-AzAutomationSchedule -AutomationAccountName $automationAccountName `
-Name $scheduleName -ResourceGroupName "ResourceGroup01"
```

Pasos siguientes

- Para obtener más información sobre los cmdlets que se usan para acceder a las programaciones, consulte Administración de módulos en Azure Automation.
- Para obtener información general sobre los runbooks, consulte Ejecución de un runbook en Azure Automation.

Last updated on 17/11/2025

Administración de variables en Azure Automation

Los recursos de variables son valores que están disponibles para todos los runbooks y configuraciones de DSC en su cuenta de Automation. Puede administrarlos desde Azure Portal, PowerShell, un runbook o en una configuración de DSC.

Las variables de Automation son útiles para los siguientes escenarios:

- Uso compartido de un valor entre varios runbooks o configuraciones de DSC.
- Uso compartido de un valor entre varios trabajos del mismo runbook o configuración de DSC.
- Administración de un valor que usan los runbooks o las configuraciones de DSC desde el portal o la línea de comandos de PowerShell. Un ejemplo es un conjunto de elementos de configuración comunes, como una lista de nombres de máquinas virtuales específica, un grupo de recursos específico, un nombre de dominio de AD y más.

Azure Automation conserva las variables y hace que estén disponibles aunque se produzca un error en un runbook o en la configuración de DSC. Este comportamiento permite que un runbook o configuración de DSC establezca un valor para que otro runbook lo use, o bien que lo use el mismo runbook o la misma configuración de DSC la siguiente vez que se ejecute.

Azure Automation almacena cada variable cifrada de forma segura. Al crear una variable, puede especificar su cifrado y almacenamiento por Azure Automation como recurso seguro. Después de crear la variable, no se puede cambiar su estado de cifrado sin volver a crearla. Si tiene variables en la cuenta de Automation que almacenan datos confidenciales que todavía no están cifrados, debe eliminarlos y volver a crearlos como variables cifradas. Una recomendación de Microsoft Defender for Cloud es cifrar todas las variables de Azure Automation como se describe en Las variables de la cuenta de Automation deben estar cifradas. Si tiene variables sin cifrar que quiere excluir de esta recomendación de seguridad, consulte Exclusión de un recurso de las recomendaciones y la puntuación segura para crear una regla de exención.

> **Nota:**
>
> Los recursos protegidos en Azure Automation incluyen credenciales, certificados, conexiones y variables cifradas. Estos recursos se cifran y se almacenan en Azure Automation con una clave única que se genera para cada cuenta de Automation. Azure Automation almacena la clave en la instancia de Key Vault administrada por el sistema. Antes de almacenar un recurso seguro, Azure Automation carga la clave desde Key Vault y después la usa para cifrar el recurso.

Tipos de variables

Cuando se crea una variable con Azure Portal, debe especificar un tipo de datos en la lista desplegable para que el portal pueda mostrar el control adecuado para escribir el valor de la variable. Los siguientes tipos de variable están disponibles en Azure Automation:

- Cadena
- Entero
- DateTime
- Booleano
- Null

La variable no se limita al tipo de datos especificado. Debe establecer la variable mediante Windows PowerShell si se desea especificar un valor de un tipo diferente. Si indica `Not defined`, el valor de la variable se establece en NULL. Debe establecer el valor con el cmdlet Set-AzAutomationVariable o el cmdlet interno `Set-AutomationVariable`. Use `Set-AutomationVariable` en los runbooks que están diseñados para ejecutarse en el entorno de espacio aislado de Azure o en un Hybrid Runbook Worker de Windows.

Azure Portal no se puede usar para crear o cambiar el valor de un tipo de variable complejo. Sin embargo, puede proporcionar un valor de cualquier tipo mediante Windows PowerShell. Los tipos complejos se recuperan como Newtonsoft.Json.Linq.JProperty para un tipo de objeto complejo, en lugar de un PSCustomObject de tipo PSObject.

Puede almacenar varios valores en una única variable mediante la creación de una matriz o tabla hash y guardarlos en la variable.

> **Nota:**
>
> Las variables de nombre de máquina virtual pueden tener un máximo de 80 caracteres. Las variables de grupo de recursos pueden tener un máximo de 90 caracteres. Consulte Reglas y restricciones de nomenclatura para los recursos de Azure.

Cmdlets de PowerShell para acceder a las variables

Los cmdlets de la tabla siguiente permiten crear y administrar variables de Automation con PowerShell. Se suministran como un componente de los módulos Az.

Expandir tabla

Cmdlet	Descripción
Get-AzAutomationVariable	Recupera el valor de una variable existente. Si el valor es de un tipo simple, se recupera ese mismo tipo. Si es un tipo complejo, se recupera un tipo `PSCustomObject`. [1]
New-AzAutomationVariable	Crea una nueva variable y establece su valor.
Remove-AzAutomationVariable	Quita una variable existente.
Set-AzAutomationVariable	Establece el valor de una variable existente.

[1] Este cmdlet no se puede usar para recuperar el valor de una variable cifrada. La única forma de hacerlo es usando el cmdlet `Get-AutomationVariable` interno en un runbook o una configuración de DSC. Por ejemplo, para ver el valor de una variable cifrada, puede crear un runbook para obtener la variable y escribirla en el flujo de salida:

PowerShell

```
$encryptvar = Get-AutomationVariable -Name TestVariable
Write-output "The encrypted value of the variable is: $encryptvar"
```

Cmdlets internos para acceder a las variables

Los cmdlets internos de la tabla siguiente se usan para acceder a las variables en los runbooks y las configuraciones de DSC. Estos cmdlets se presentan con el módulo global `Orchestrator.AssetManagement.Cmdlets`. Para más información, consulte Cmdlets internos.

Expandir tabla

Cmdlet interno	Descripción
`Get-AutomationVariable`	Recupera el valor de una variable existente.
`Set-AutomationVariable`	Establece el valor de una variable existente.

Nota:

Evite usar variables en el parámetro `Name` de cmdlet `Get-AutomationVariable` en un runbook o configuración de DSC. El uso de una variable puede complicar la detección de dependencias entre runbooks y variables de Automation durante el tiempo de diseño.

Funciones de Python para acceder a las variables

Las funciones de la siguiente tabla sirven para acceder a las variables de un runbook de Python 2 y Python 3. Actualmente, los runbooks de Python 3 están en versión preliminar.

Expandir tabla

Funciones de Python	Descripción
`automationassets.get_automation_variable`	Recupera el valor de una variable existente.
`automationassets.set_automation_variable`	Establece el valor de una variable existente.

> **Nota:**
>
> Para acceder a las funciones del activo, debe importar el módulo `automationassets` en la parte superior del runbook de Python.

Creación y obtención de una variable

> **Nota:**
>
> Si va a quitar el cifrado de una variable, debe eliminarla y volver a crearla sin cifrado.

Creación y obtención de una variable con Azure Portal

1. En la cuenta de Automation, en el panel izquierdo, seleccione **Variables** en **Recursos compartidos**.
2. En la página **Variables**, seleccione **Agregar una variable**.
3. Complete las opciones de la página **Nueva variable** y, luego, haga clic en **Crear** para guardar la nueva variable.

> **Nota:**
>
> Una vez que una variable cifrada se guarde, no podrá verse en el portal. Solo se podrá actualizar.

Creación y obtención de una variable en Windows PowerShell

El runbook o la configuración de DSC usa el cmdlet `New-AzAutomationVariable` para crear una nueva variable y establecer su valor inicial. Si la variable está cifrada, la llamada debe usar el parámetro `Encrypted`. El script puede recuperar el valor de la variable mediante `Get-AzAutomationVariable`.

> ⓘ **Nota:**
>
> Un script de PowerShell no puede recuperar un valor cifrado. La única forma de hacerlo es usar el cmdlet `Get-AutomationVariable` interno.

En el siguiente ejemplo se muestra cómo crear una variable de tipo cadena y después se devuelve su valor.

PowerShell

```
$rgName = "ResourceGroup01"
$accountName = "MyAutomationAccount"
$variableValue = "My String"

New-AzAutomationVariable -ResourceGroupName "ResourceGroup01"
-AutomationAccountName "MyAutomationAccount" -Name 'MyStringVariable' `
-Encrypted $false -Value 'My String'
$string = (Get-AzAutomationVariable -ResourceGroupName "ResourceGroup01" `
-AutomationAccountName "MyAutomationAccount" -Name 'MyStringVariable').Value
```

En el siguiente ejemplo se muestra cómo crear una variable con un tipo complejo y después se recuperan sus propiedades. En este caso, se usa un objeto de máquina virtual de Get-AzVM que especifica un subconjunto de sus propiedades.

PowerShell

```
$rgName = "ResourceGroup01"
$accountName = "MyAutomationAccount"

$vm = Get-AzVM -ResourceGroupName "ResourceGroup01" -Name "VM01" | Select Name,
Location, Extensions
New-AzAutomationVariable -ResourceGroupName "ResourceGroup01" -AutomationAccountName
"MyAutomationAccount" -Name "MyComplexVariable" -Encrypted $false -Value $vm

$vmValue = Get-AzAutomationVariable -ResourceGroupName "ResourceGroup01" `
-AutomationAccountName "MyAutomationAccount" -Name "MyComplexVariable"

$vmName = $vmValue.Value.Name
$vmTags = $vmValue.Value.Tags
```

Ejemplo de un runbook textual

PowerShell

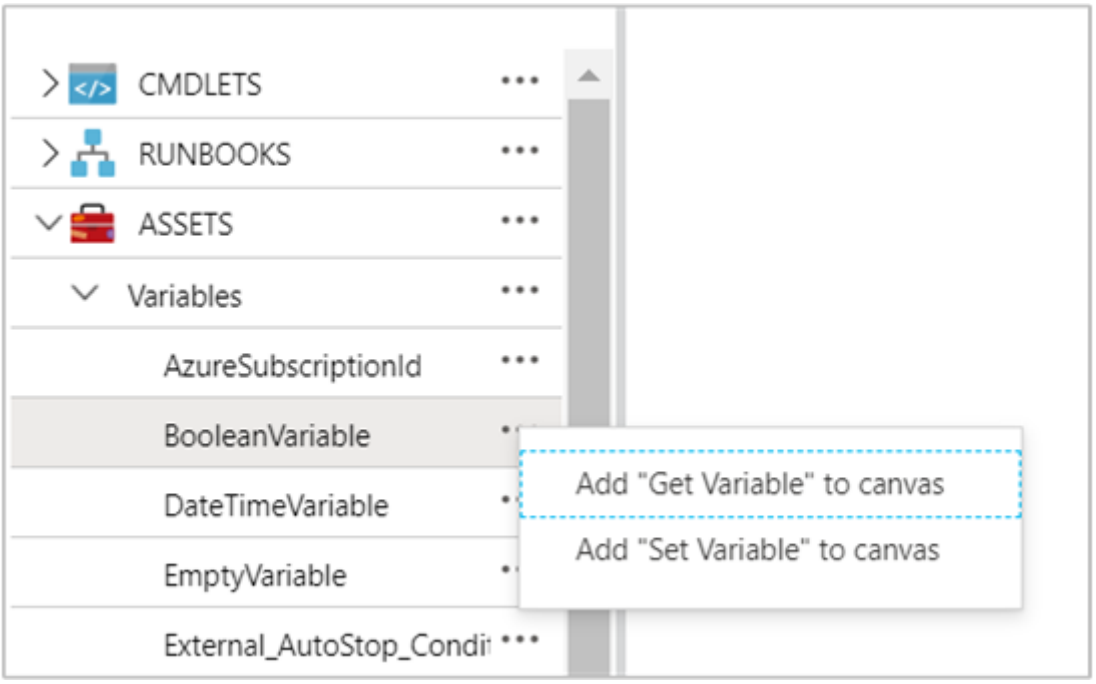
Los comandos de ejemplo siguientes muestran cómo establecer y recuperar una variable en un runbook textual. En este ejemplo se da por hecho que ha creado variables de entero con el nombre **numberOfIterations** y **numberOfRunnings**, así como una variable de cadena con el nombre **sampleMessage**.

PowerShell

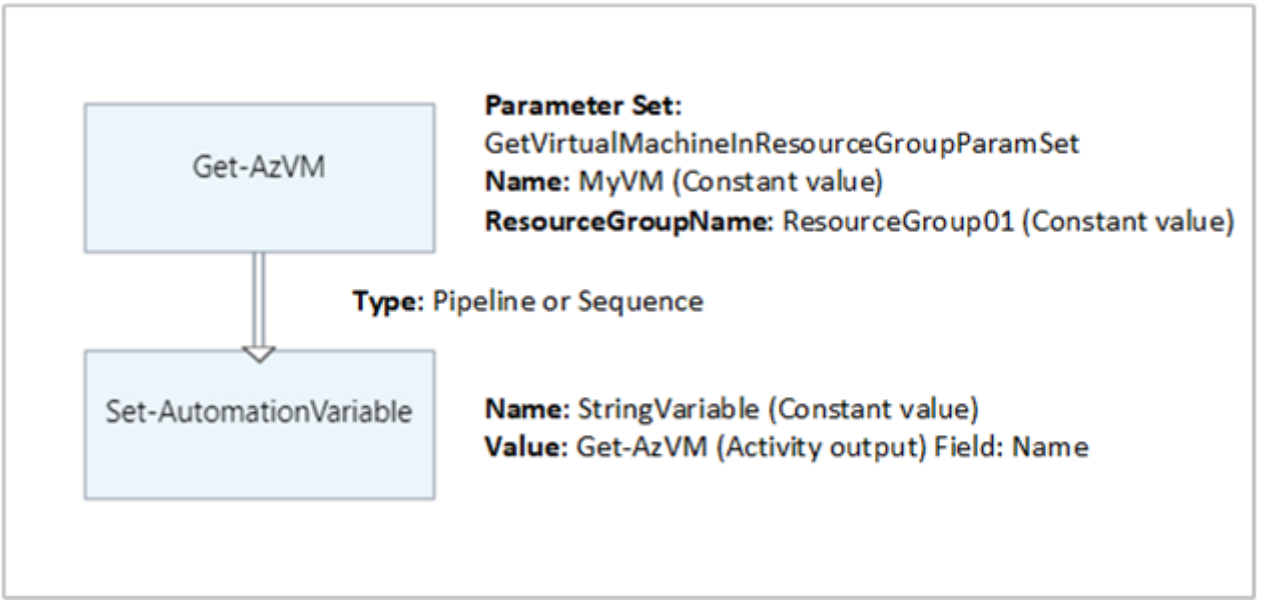
```
$rgName = "ResourceGroup01"
$accountName = "MyAutomationAccount"

$numberOfIterations = Get-AutomationVariable -Name "numberOfIterations"
$numberOfRunnings = Get-AutomationVariable -Name "numberOfRunnings"
$sampleMessage = Get-AutomationVariable -Name "sampleMessage"

Write-Output "Runbook has been run $numberOfRunnings times."

for ($i = 1; $i -le $numberOfIterations; $i++) {
    Write-Output "$i`: $sampleMessage"
}
Set-AutomationVariable -Name numberOfRunnings -Value ($numberOfRunnings += 1)
```

Ejemplos de runbook gráfico

En un runbook gráfico, puede agregar actividades para **Get-AutomationVariable** o **Set-AutomationVariable** de los cmdlet internos. Basta con hacer clic con el botón derecho en cada variable en el panel Biblioteca del editor gráfico y seleccionar la actividad que se va a agregar.



La imagen siguiente muestra las actividades de ejemplo para actualizar una variable con un valor simple en un runbook gráfico. En este ejemplo, la actividad para `Get-AzVM` recupera una única máquina virtual de Azure y guarda el nombre del equipo en una variable de cadena de Automation existente. No importa si el vínculo es una canalización o una secuencia, ya que el código solo espera un único objeto en la salida.



Pasos siguientes

- Para más información sobre los cmdlets que se usan para acceder a las variables, consulte Administración de módulos en Azure Automation.
- Para obtener información general sobre los runbooks, consulte Ejecución de un runbook en Azure Automation.

- Para más información sobre las configuraciones DSC, vea Introducción a State Configuration de Azure Automation.

Last updated on 17/11/2025

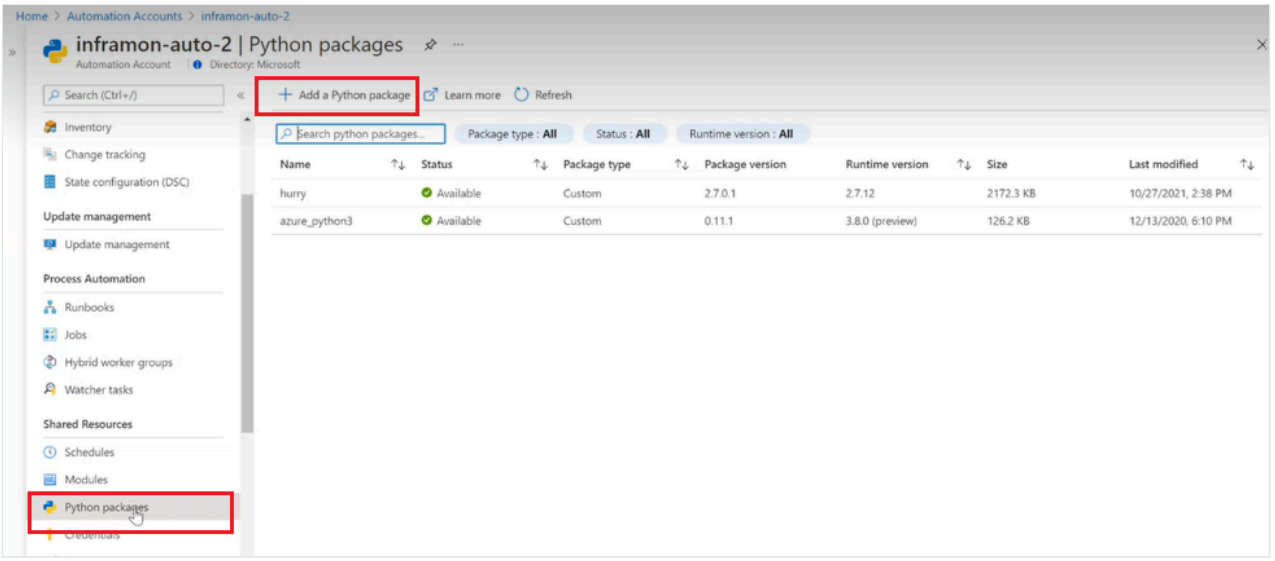
Administración de paquetes de Python 2 en Azure Automation

En este artículo se describe cómo importar, administrar y usar paquetes de Python 2 en Azure Automation que se ejecutan en el entorno de espacio aislado de Azure e Hybrid Runbook Worker. Para simplificar los runbooks, puede usar paquetes de Python para importar los módulos que necesita.

Para obtener información sobre cómo administrar paquetes de Python 3, consulte Administración de paquetes de Python 3.

Importación de paquetes

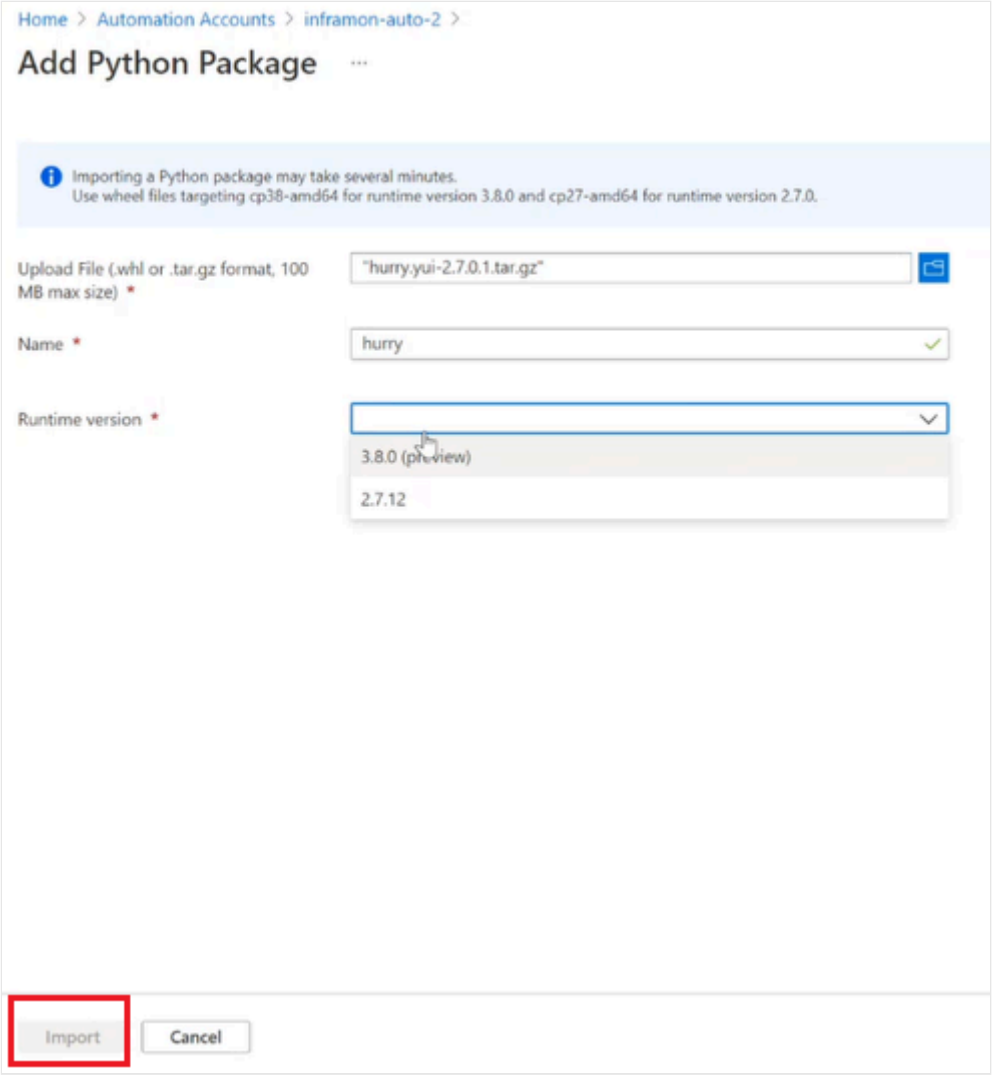
1. Seleccione **Paquetes de Python** en **Recursos compartidos**, en la cuenta de Automation. Seleccione **+ Agregar un paquete de Python**.



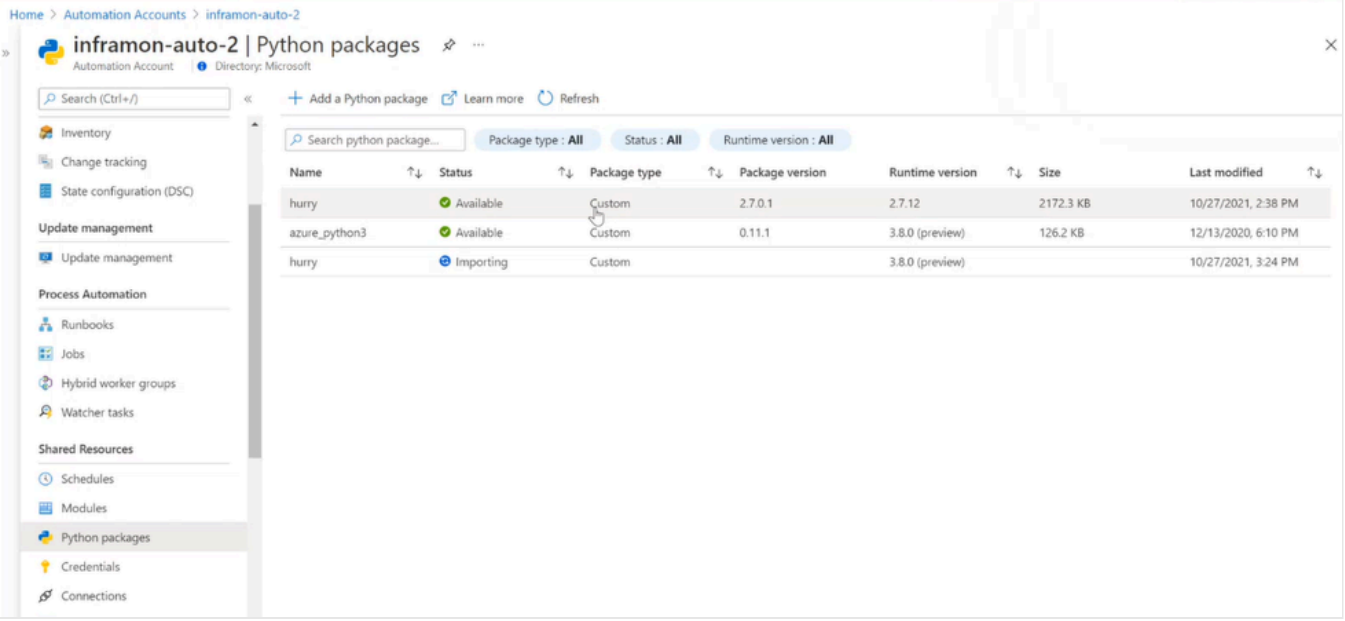
2. En la página **Agregar un paquete de Python**, seleccione un paquete local para cargar. El paquete puede ser un archivo **.whl**.

3. Escriba el nombre y seleccione la **versión del entorno de ejecución** como 2.x.x.

4. Seleccione **Import** (Importar).



Una vez que se haya importado un paquete, este se muestra en la página de **Paquetes de Python** en su cuenta de Automation. Para quitar un paquete, selecciónelo y luego seleccione **Eliminar**.



Importación de paquetes con dependencias

Azure Automation no resuelve las dependencias de los paquetes de Python durante el proceso de importación. Hay dos formas de importar un paquete con todas sus dependencias. Solo es necesario realizar uno de los pasos siguientes para importar los paquetes en su cuenta de Automation.

Descarga manual

En una máquina Windows de 64 bits con Python 2.7 y pip instalados, ejecute el siguiente comando para descargar un paquete y todas sus dependencias:

Símbolo del sistema de Windows

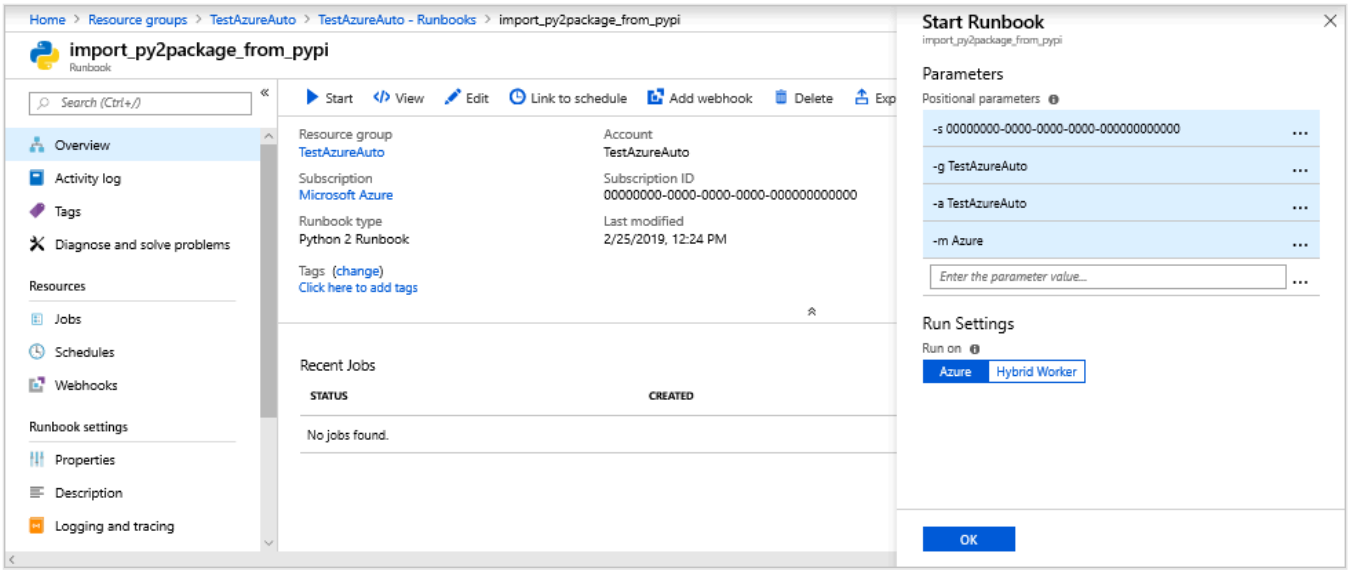
```
C:\Python27\Scripts\pip2.7.exe download -d <output dir> <package name>
```

Una vez que se han descargado los paquetes, puede importarlos en la cuenta de Automation.

Runbook

Para obtener un runbook, importe paquetes de Python 2 desde pypi a la cuenta de Azure Automation desde la organización GitHub de Azure Automation a la cuenta de Automation. Asegúrese de que los parámetros de ejecución están establecidos en **Azure** e inicie el runbook con los parámetros. Asegúrese de que la identidad administrada esté habilitada para la cuenta de Automation y que tenga acceso de colaborador de Automation para la importación correcta del paquete. Asegúrese de iniciar cada parámetro con el modificador, como se ve en la lista e imagen siguientes:

- -s <subscriptionId>
- -g <resourceGroup>
- -a <automationAccount>
- -m <modulePackage>



El runbook permite especificar el paquete que se va a descargar. Por ejemplo, el uso del parámetro `Azure` descarga todos los módulos de Azure y todas las dependencias (aproximadamente 105). Una vez completado el runbook, puede consultar la página **Paquetes de Python** en **Recursos compartidos** en su cuenta de Automation para comprobar que el paquete se importó correctamente.

Usar un paquete en un runbook

Si tiene un paquete importado, puede usarlo en un runbook. Agregue el código siguiente para mostrar todos los grupos de recursos de una suscripción de Azure.

Python

```python
#!/usr/bin/env python
import os
import requests
# printing environment variables
endPoint = os.getenv('IDENTITY_ENDPOINT') + "?
resource=https://management.azure.com/"
identityHeader = os.getenv('IDENTITY_HEADER')
payload = {}
headers = {
    'X-IDENTITY-HEADER': identityHeader,
    'Metadata': 'True'
}
response = requests.request("GET", endPoint, headers=headers, data=payload)
print response.text
```

Desarrollar y probar runbooks sin conexión

Para desarrollar y probar los runbooks de Python 2 desconectado, puede usar el módulo Recursos emulados de Python de Azure Automation en GitHub. Este módulo le permite hacer referencia a recursos compartidos como credenciales, variables, conexiones y certificados.

Pasos siguientes

Para preparar un runbook de Python, consulte Creación de un runbook de Python.

Last updated on 17/11/2025

Administración de paquetes de Python 3 en Azure Automation

15/08/2025

En este artículo se describe cómo importar, administrar y usar paquetes de Python 3 en Azure Automation que se ejecutan en el entorno de espacio aislado de Azure y Hybrid Runbook Worker. Los paquetes de Python deben descargarse en instancias de Hybrid Runbook Worker para la ejecución correcta del trabajo. Para simplificar los runbooks, puede usar paquetes de Python para importar los módulos que necesita.

Para obtener información sobre cómo administrar paquetes de Python 2, consulte Administración de paquetes de Python 2.

> **Nota**
>
> Python 2.7 y Python 3.8 ya no son compatibles con python de la plataforma principal. Se recomienda crear runbooks solo en Python 3.10.

Paquetes de Python predeterminados

Para admitir runbooks de Python 3.8 en el servicio Automation, algunos paquetes de Python se instalan de manera predeterminada. Aquí hay una lista de estos paquetes. La versión predeterminada se puede invalidar si se importan los paquetes de Python en la cuenta de Automation.

Asimismo, se da preferencia a la versión importada en la cuenta de Automation. Para importar un solo paquete, consulte Importación de un paquete. Para importar un paquete con varios paquetes, consulte Importación de un paquete con dependencias.

> **Nota**
>
> No hay paquetes predeterminados instalados para Python 3.10.

Paquetes como archivos de origen

Azure Automation solo admite un paquete de Python que solo contenga código de Python y no incluya otras extensiones de lenguaje ni código en otros lenguajes. Sin embargo, es posible

que el entorno de espacio aislado de Azure no tenga los compiladores necesarios para los archivos binarios de C/C++, por lo que se recomienda usar en su lugar archivos wheel.

> **Nota**
>
> Actualmente, Python 3.10 solo admite archivos wheel.

El Índice de paquetes de Python (PyPI) es un repositorio de software para el lenguaje de programación Python. Al seleccionar un paquete de Python 3 para importarlo a su cuenta de Automation desde PyPI, tenga en cuenta los siguientes elementos del nombre de archivo:

Seleccione una versión de Python:

Python 3.10

Expandir tabla

Elemento de nombre de archivo	Descripción
cp310	Automation admite **trabajos de Python 3.10** for Cloud.
manylinux_x86_64	Los procesos de espacio aislado de Azure son una arquitectura de 64 bits basada en Linux para runbooks de Python 3.10.

Por ejemplo:

- Para importar pandas, puede seleccionar un archivo wheel con un nombre similar a `pandas-1.5.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl`.

Algunos paquetes de Python disponibles en PyPI no proporcionan un archivo wheel. En este caso, descargue el origen (archivo .zip o .tar.gz) y genere el archivo wheel mediante pip.

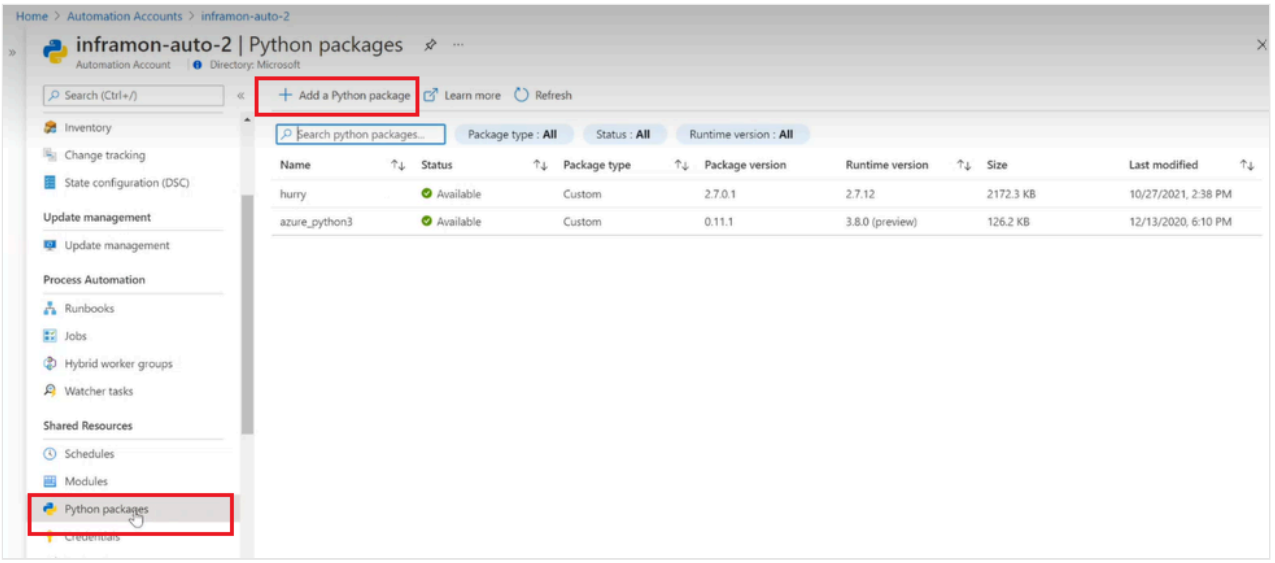
Realice los pasos siguientes con una máquina Linux de 64 bits con Python 3.10.x y el paquete wheel instalado:

1. Descargue el archivo de código fuente `pandas-1.2.4.tar.gz.`.
2. Ejecute pip para obtener el archivo wheel con el siguiente comando: `pip wheel --no-deps pandas-1.2.4.tar.gz`.

Importación de un paquete

Para importar un paquete, siga estos pasos:

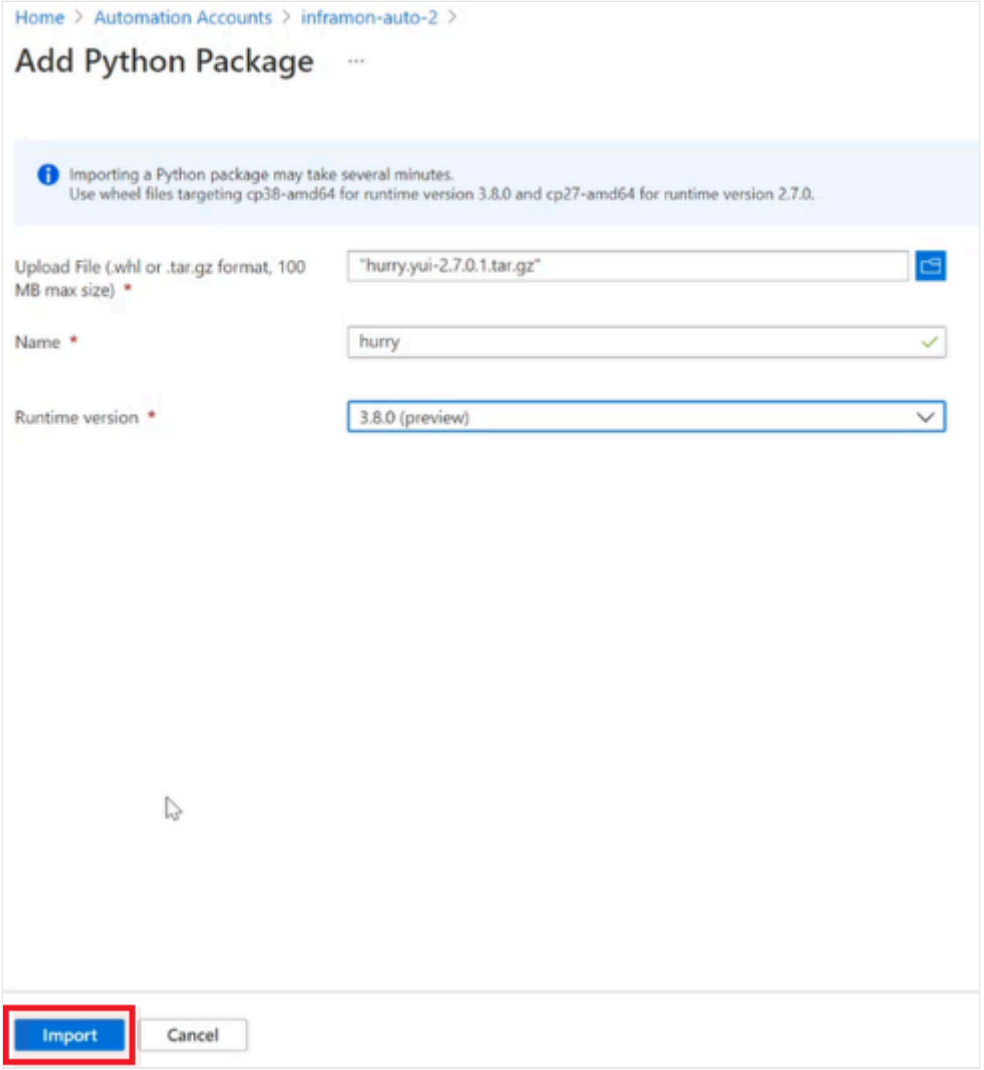
1. Seleccione **Paquetes de Python** en **Recursos compartidos**, en la cuenta de Automation. A continuación, seleccione **+ Agregar un paquete de Python**.



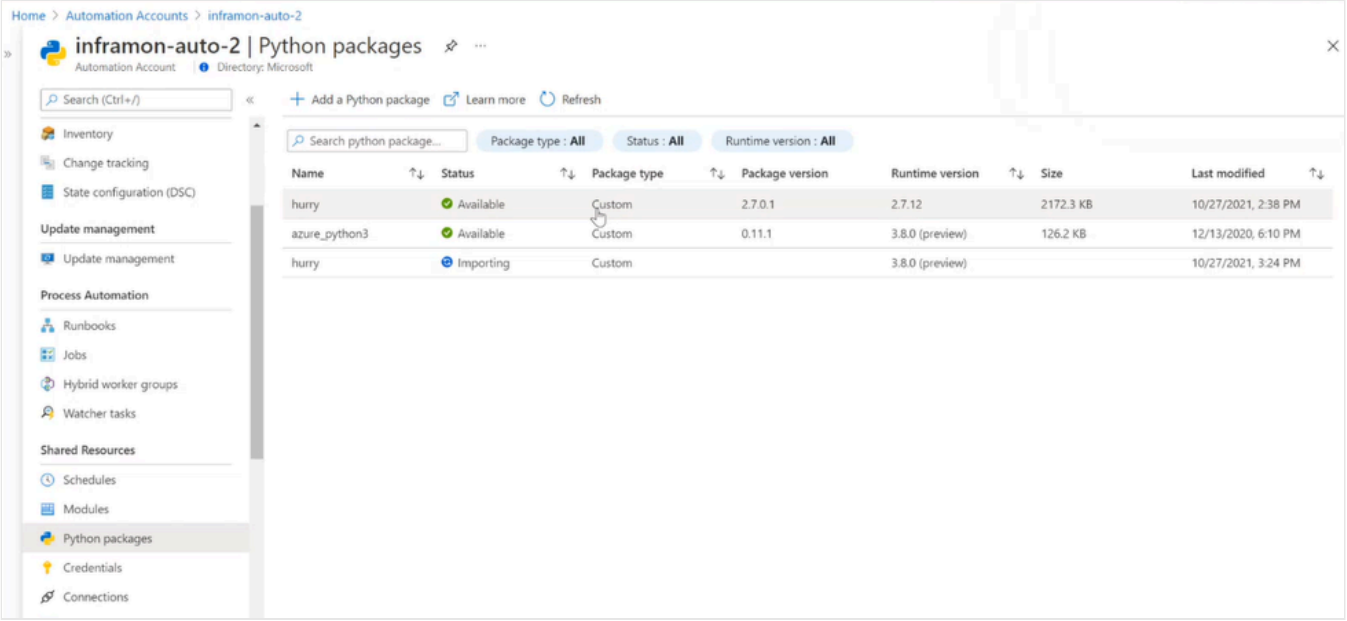
2. En la página **Agregar un paquete de Python**, seleccione un paquete local para cargar. El paquete puede ser **.whl** o **.tar.gz** archivo para python 3.8 y **.whl** para Python 3.10.

3. Escriba un nombre y seleccione la **versión en tiempo de ejecución** como Python 3.8 o Python 3.10.

4. Seleccione **importar**.



Una vez que se haya importado un paquete, este se muestra en la página de Paquetes de Python en su cuenta de Automation. Para quitar un paquete, selecciónelo y luego seleccione **Eliminar**.



Importación de un paquete con dependencias

Para importar un paquete de Python 3.8 y sus dependencias, importe el siguiente script de Python en un runbook de Python 3.8. Asegúrese de que la identidad administrada esté habilitada para la cuenta de Automation y que tenga acceso de colaborador de Automation para la importación correcta del paquete.

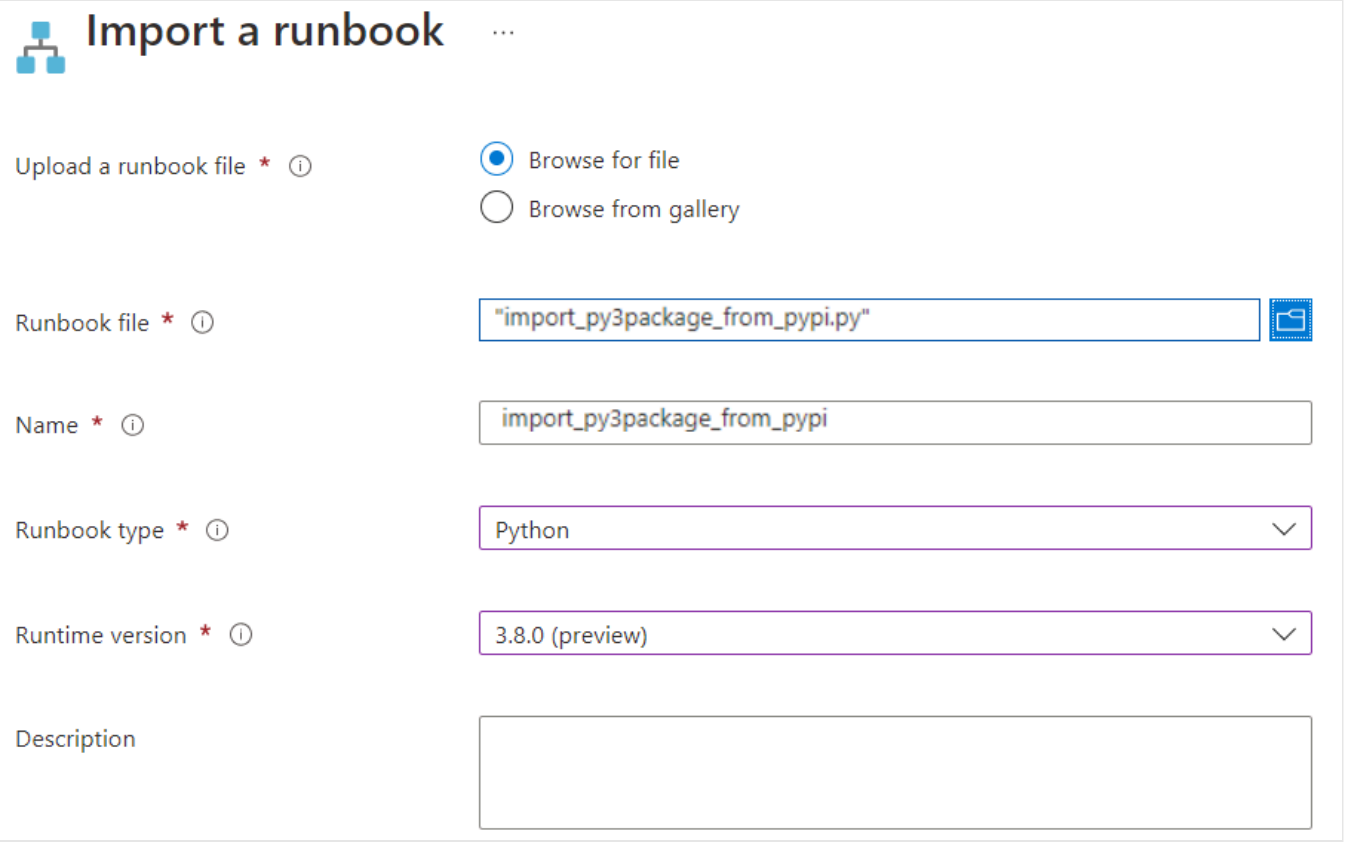
Identidad administrada asignada por el sistema

```
https://github.com/azureautomation/runbooks/blob/master/Utility/Python/import_
py3package_from_pypi.py
```

Importación del script en un runbook

Para información sobre cómo importar el runbook, consulte Importación de un runbook desde Azure Portal. Copie el archivo de GitHub en el almacenamiento al que se pueda acceder desde el portal antes de realizar la importación.

La página **Importar un runbook** adopta como valor predeterminado el nombre del runbook para que coincida con el nombre del script. Si tiene acceso al campo, puede cambiar el nombre. **Tipo de runbook** puede adoptar como valor predeterminado **Python 2.7**. Si es así, asegúrese de cambiarlo a **Python 3.8**.



Ejecute el runbook para importar el paquete y las dependencias.

Después de crear y publicar el runbook, ejecútelo para importar el paquete. Consulte Inicio de un runbook en Azure Automation para más información sobre la ejecución del runbook.

El script `import_py3package_from_pypi.py` requiere los siguientes parámetros:

Expandir tabla

Parámetro	Descripción
subscription_id	Identificador de suscripción de la cuenta de Automation
resource_group	Nombre del grupo de recursos en el que se define la cuenta de Automation
automation_account	Nombre de la cuenta de Automation
module_name	Nombre del módulo que se va a importar desde `pypi.org`
module_version	Versión del módulo

El valor del parámetro debe proporcionarse como una sola cadena en el formato siguiente:

-s <subscription_id> -g <resource_group> -a<automation_account> -m <module_name> -v module_version <>

Para más información sobre el uso de parámetros con runbooks, consulte Trabajar con parámetros del runbook.

Usar un paquete en un runbook

Puede usar el paquete importado en un runbook. Agregue el código siguiente para mostrar todos los grupos de recursos de una suscripción de Azure.

Python

```python
#!/usr/bin/env python3
import os
import requests
# printing environment variables
endPoint = os.getenv('IDENTITY_ENDPOINT')+"?
resource=https://management.azure.com/"
identityHeader = os.getenv('IDENTITY_HEADER')
payload={}
headers = {
  'X-IDENTITY-HEADER': identityHeader,
  'Metadata': 'True'
}
```

```
response = requests.request("GET", endPoint, headers=headers, data=payload)
print(response.text)
```

> **Nota**
>
> El paquete `automationassets` de Python no está disponible en pypi.org, por lo que tampoco se puede importar a un trabajo de runbook híbrido de Windows.

Identificación de los paquetes disponibles en el espacio aislado

Use el código siguiente para enumerar los módulos instalados predeterminados:

Python

```python
#!/usr/bin/env python3

import pkg_resources
installed_packages = pkg_resources.working_set
installed_packages_list = sorted(["%s==%s" % (i.key, i.version)
   for i in installed_packages])

for package in installed_packages_list:
    print(package)
```

Cmdlets de PowerShell de Python 3.8

Adición de un nuevo paquete de Python 3.8

Python

```
New-AzAutomationPython3Package -AutomationAccountName tarademo  -ResourceGroupName
mahja -Name requires.io -ContentLinkUri
https://files.pythonhosted.org/packages/7f/e2/85dfb9f7364cbd7a9213caea0e91fc948da3
c912a2b222a3e43bc9cc6432/requires.io-0.2.6-py2.py3-none-any.whl

Response
ResourceGroupName     : mahja
AutomationAccountName : tarademo
Name                  : requires.io
IsGlobal              : False
Version               :
SizeInBytes           : 0
```

```
ActivityCount         : 0
CreationTime          : 9/26/2022 1:37:13 PM +05:30
LastModifiedTime      : 9/26/2022 1:37:13 PM +05:30
ProvisioningState     : Creating
```

Enumeración de todos los paquetes de Python 3.8

Python

```
Get-AzAutomationPython3Package -AutomationAccountName tarademo  -ResourceGroupName
mahja

Response :
ResourceGroupName     : mahja
AutomationAccountName : tarademo
Name                  : cryptography
IsGlobal              : False
Version               :
SizeInBytes           : 0
ActivityCount         : 0
CreationTime          : 9/26/2022 11:52:28 AM +05:30
LastModifiedTime      : 9/26/2022 12:11:00 PM +05:30
ProvisioningState     : Failed
ResourceGroupName     : mahja
AutomationAccountName : tarademo
Name                  : requires.io
IsGlobal              : False
Version               :
SizeInBytes           : 0
ActivityCount         : 0
CreationTime          : 9/26/2022 1:37:13 PM +05:30
LastModifiedTime      : 9/26/2022 1:39:04 PM +05:30
ProvisioningState     : ContentValidated
ResourceGroupName     : mahja
AutomationAccountName : tarademo
Name                  : sockets
IsGlobal              : False
Version               : 1.0.0
SizeInBytes           : 4495
ActivityCount         : 0
CreationTime          : 9/20/2022 12:46:28 PM +05:30
LastModifiedTime      : 9/22/2022 5:03:42 PM +05:30
ProvisioningState     : Succeeded
```

Obtener detalles sobre un paquete específico

Python

```
Get-AzAutomationPython3Package -AutomationAccountName tarademo  -ResourceGroupName
mahja -Name sockets


Response
ResourceGroupName     : mahja
AutomationAccountName : tarademo
Name                  : sockets
IsGlobal              : False
Version               : 1.0.0
SizeInBytes           : 4495
ActivityCount         : 0
CreationTime          : 9/20/2022 12:46:28 PM +05:30
LastModifiedTime      : 9/22/2022 5:03:42 PM +05:30
ProvisioningState     : Succeeded
```

Eliminación del paquete de Python 3.8

Python

```
Remove-AzAutomationPython3Package -AutomationAccountName tarademo  -
ResourceGroupName mahja -Name sockets
```

Actualización del paquete de Python 3.8

Python

```
Set-AzAutomationPython3Package -AutomationAccountName tarademo  -ResourceGroupName
mahja -Name requires.io -ContentLinkUri
https://files.pythonhosted.org/packages/7f/e2/85dfb9f7364cbd7a9213caea0e91fc948da3
c912a2b222a3e43bc9cc6432/requires.io-0.2.6-py2.py3-none-any.whl


ResourceGroupName     : mahja
AutomationAccountName : tarademo
Name                  : requires.io
IsGlobal              : False
Version               : 0.2.6
SizeInBytes           : 10109
ActivityCount         : 0
CreationTime          : 9/26/2022 1:37:13 PM +05:30
LastModifiedTime      : 9/26/2022 1:43:12 PM +05:30
ProvisioningState     : Creating
```

Pasos siguientes

Para preparar un runbook de Python, consulte Creación de un runbook de Python.

Uso de los runbooks y módulos existentes

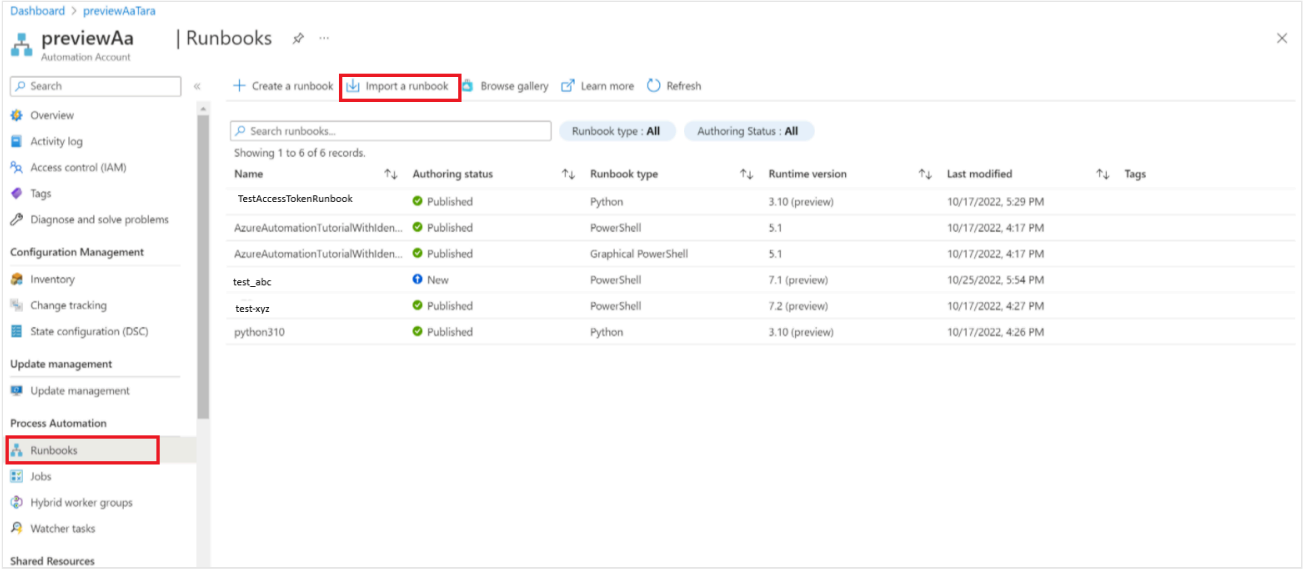
En lugar de crear sus propios runbooks y módulos en Azure Automation, puede acceder a escenarios que ya se han creado mediante Microsoft y la comunidad. Puede obtener runbooks de Python y PowerShell relacionados con Azure en la galería de runbooks de Azure Portal, y módulos y runbooks (que pueden ser específicos de Azure o no) en la galería de PowerShell. También puede contribuir a la comunidad si comparte los escenarios que desarrolle.

> **Nota:**
>
> El Centro de scripts de TechNet se va a retirar. Todos los runbooks del centro de scripts de la galería de runbooks se han migrado a la organización GitHub de Automation. Para más información, consulteMigración de runbooks de Azure Automation a GitHub.

Importación de runbooks desde GitHub con el portal de Azure

> **Nota:**
>
> - La opción **Browse gallery** en el portal de Azure tiene una experiencia de usuario mejorada.
> - En la hoja **Automatización de procesos**>**Runbook**, puede importar runbooks mediante la opción **Importar un runbook** o **Examinar galería** y la página **Runbooks** muestra dos columnas nuevas: **Versión en entorno de ejecución** y **Tipo de runbook**.

1. En el portal de Azure, abra la cuenta de Automation.
2. Seleccione la hoja **Runbooks** en **Automatización de procesos**.
3. Haga clic en **Importar un runbook** en la página **Runbooks**.

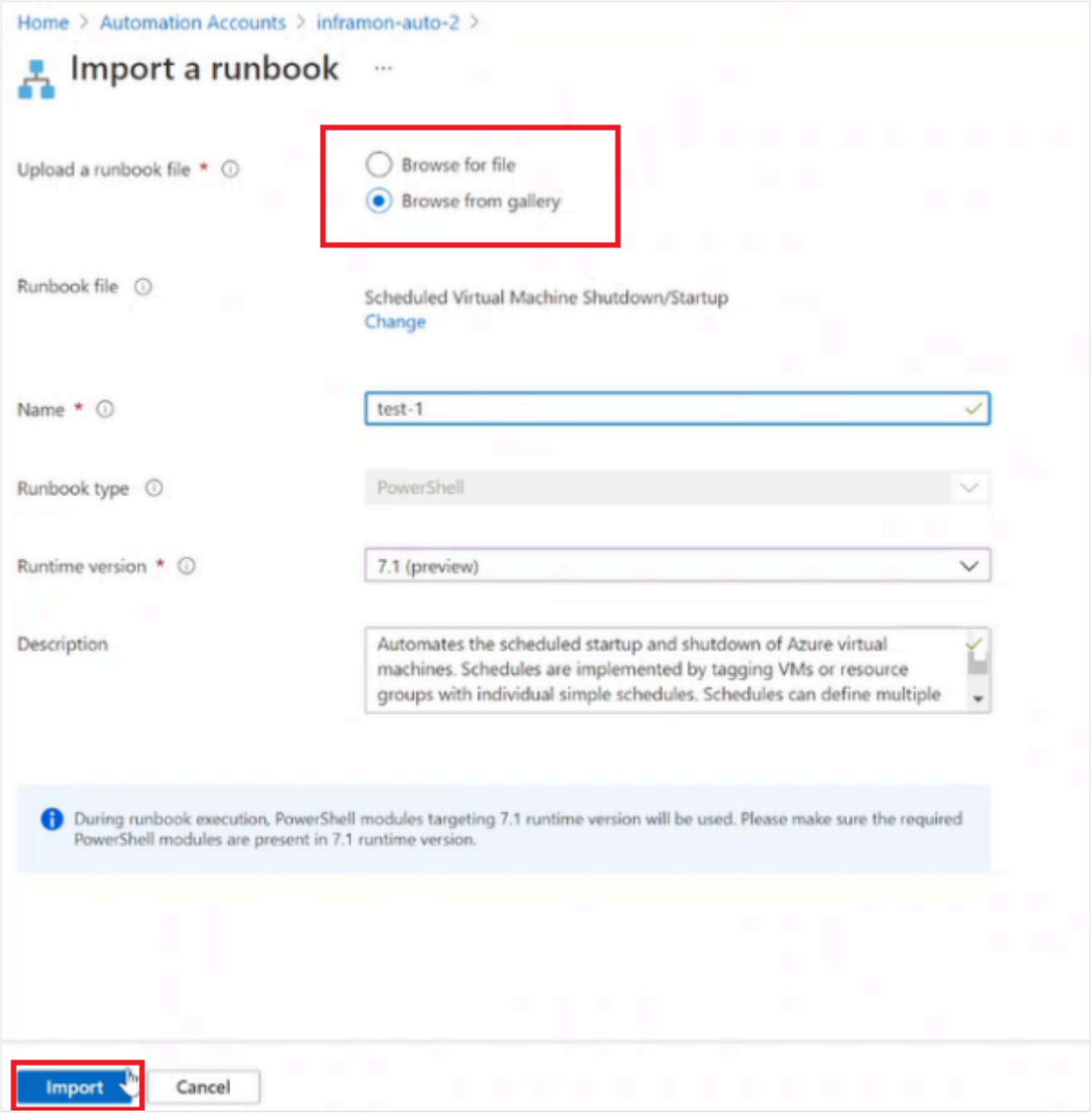


4. En la página **Importar un runbook**, puede importar un archivo almacenado en el equipo local o desde GitHub mediante **Buscar archivo** o **Examinar la galería**, respectivamente.

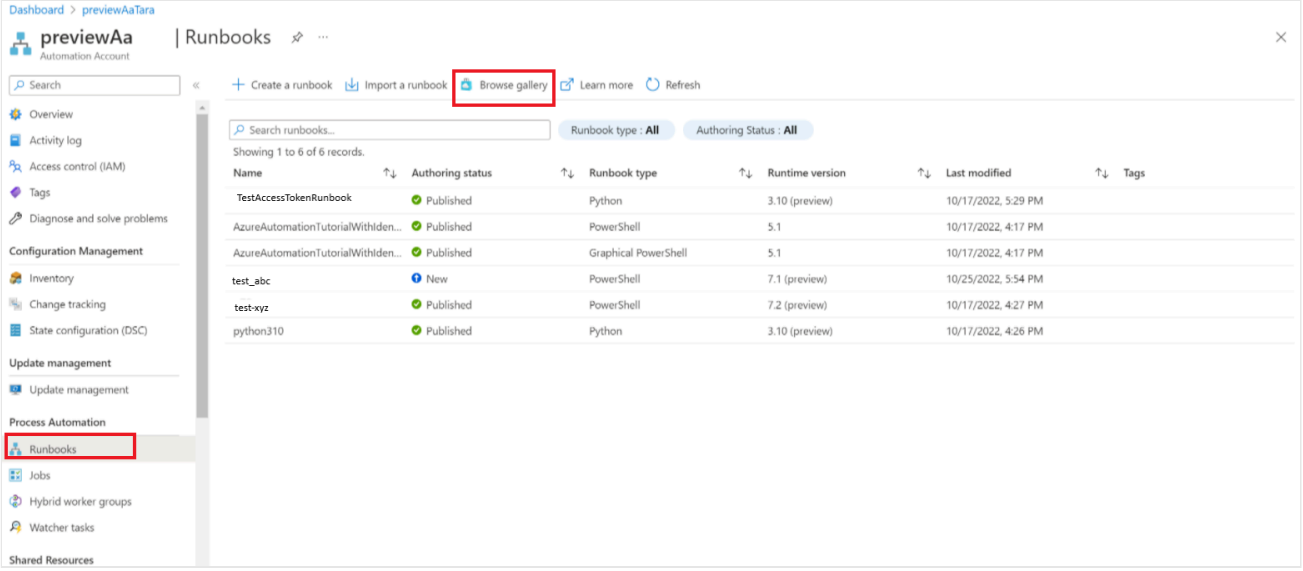
5. Seleccione el archivo.

6. Escriba el **nombre**, la **versión del entorno de ejecución** y la **descripción**.

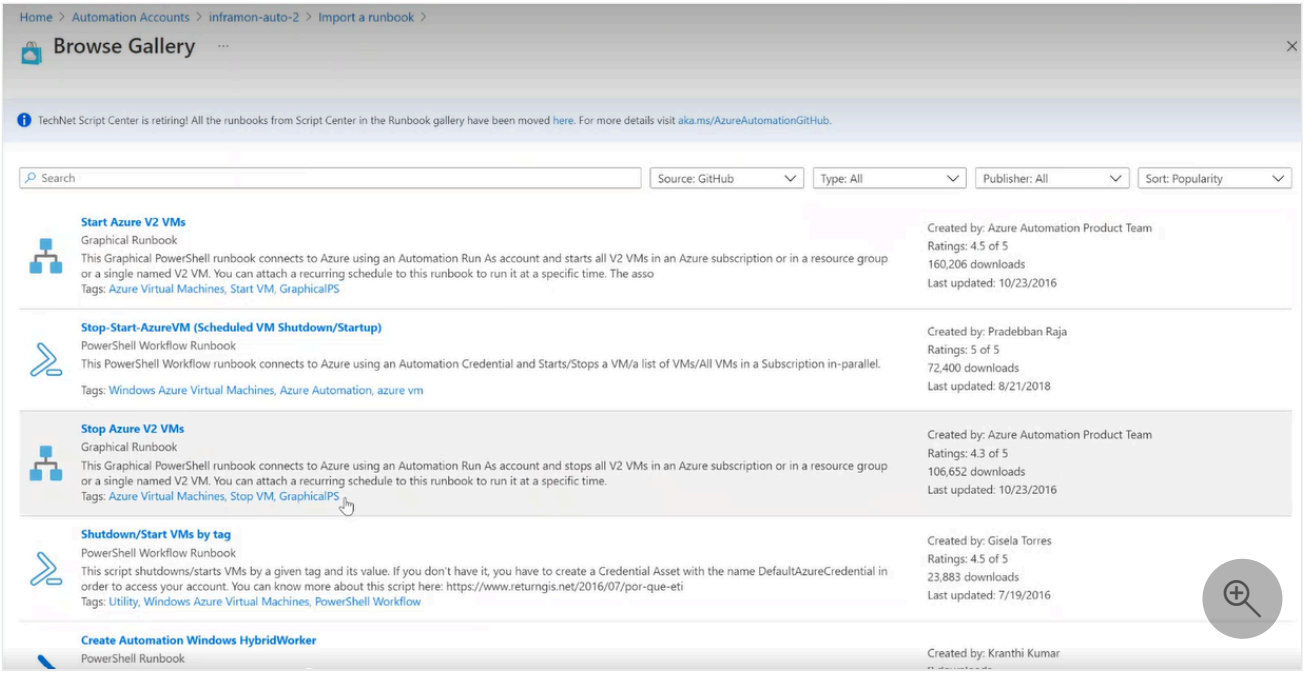
7. Haga clic en **Import**.



8. Como alternativa, seleccione **Explorar galería** en la página **Runbooks** para revisar los runbooks disponibles.



9. Puede usar los filtros situados encima de la lista para restringir la presentación por publicador, tipo y ordenación. Busque el elemento de la galería que desee y selecciónelo para ver sus detalles.

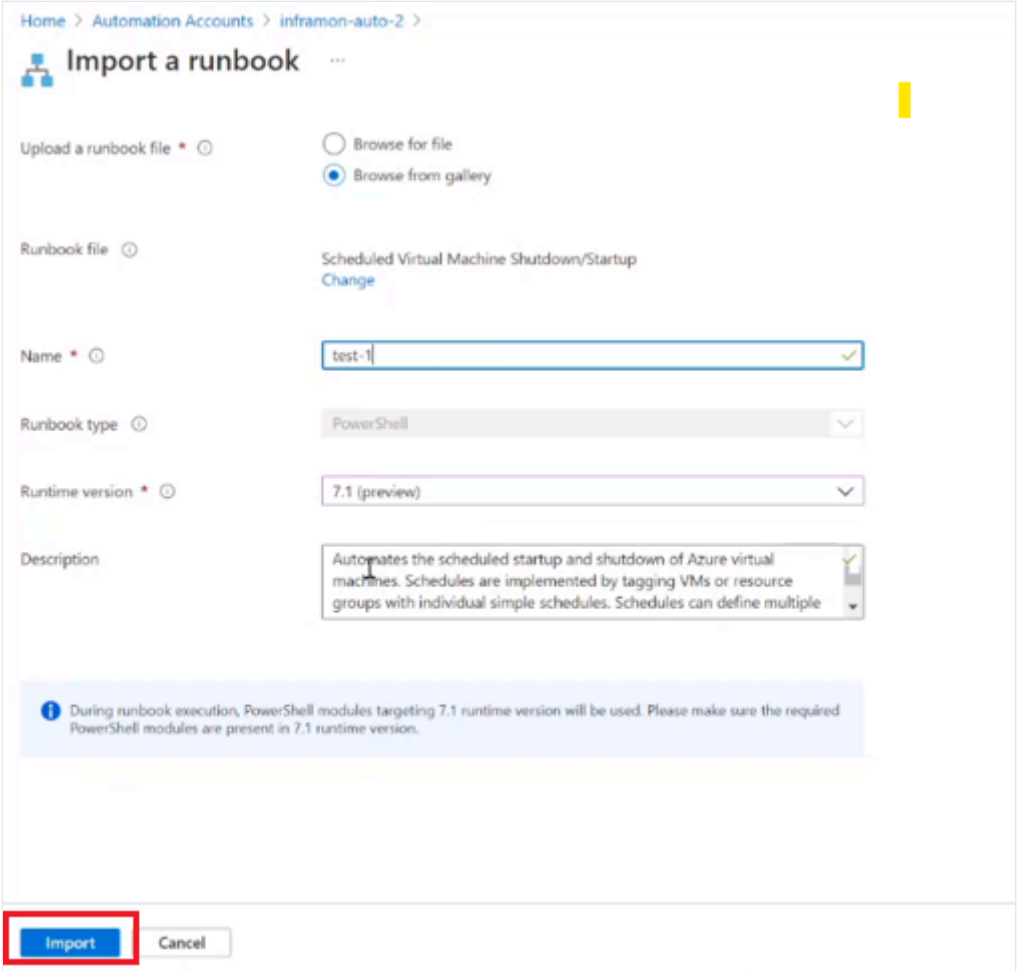


10. Haga clic en **Seleccionar** para seleccionar el runbook que haya elegido.

11. En la página **Importar un runbook**, escriba el **nombre** y seleccione las **versiones del entorno de ejecución**.

12. El **tipo de Runbook** y la **descripción** se rellenan automáticamente.

13. Haga clic en **Import**.



14. El runbook aparecerá en la pestaña **Runbooks** para la cuenta de Automation.

Runbooks de la galería de PowerShell

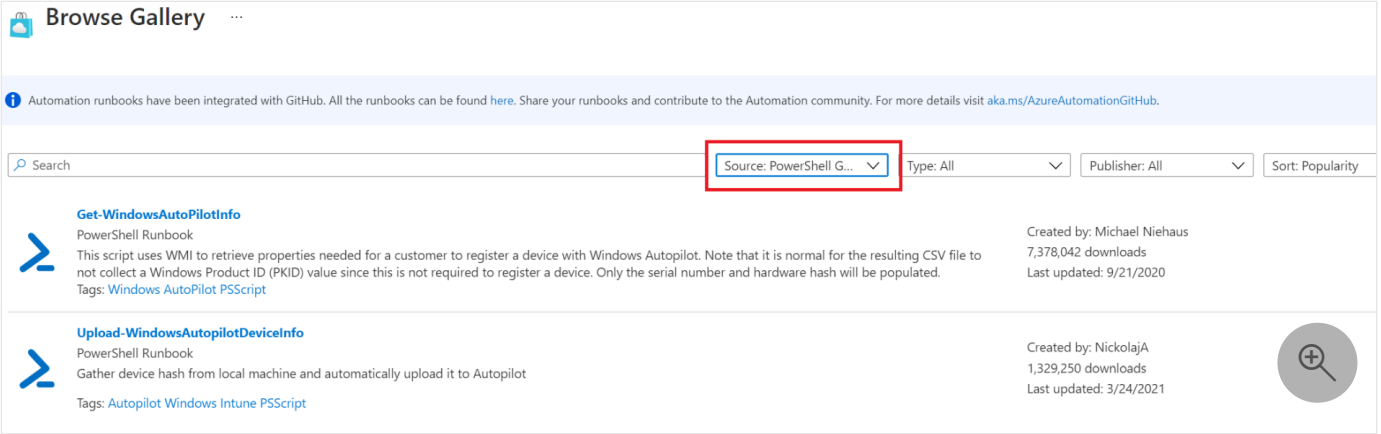
> **Importante**
>
> Debe comprobar el contenido de cualquier runbook que obtenga de la galería de PowerShell. Tenga mucho cuidado al instalarlos y ejecutarlos en un entorno de producción.

La galería de PowerShell ofrece diversos runbooks de Microsoft y de la comunidad que se pueden importar a Azure Automation. Puede usar uno al descargarlo de la galería, o puede importar runbooks directamente desde la galería o de la cuenta de Automation en Azure Portal.

> **Nota:**
>
> Los runbooks gráficos no son compatibles en Galería de PowerShell.

Solo puede importar directamente desde el Galería de PowerShell mediante el portal de Azure. No puede realizar esta función mediante PowerShell. El procedimiento es el mismo que se

muestra en Import runbooks de GitHub con el portal de Azure, salvo que el **Source** será **Galería de PowerShell**.



Módulos del Galería de PowerShell

Los módulos de PowerShell contienen cmdlets que puede utilizar en sus runbooks. Los módulos existentes que puede instalar en Azure Automation están disponibles en el Galería de PowerShell. Puede iniciar esta galería desde el portal de Azure e instalar los módulos directamente en Azure Automation, o bien puede descargarlos e instalarlos manualmente.

También puede encontrar módulos para importar en el portal de Azure. Se muestran para la cuenta de Automation en la opción **Módulos** en **Recursos compartidos**.

> **Importante**
>
> No incluya la palabra clave "AzureRm" en ningún script diseñado para ejecutarse con el módulo Az. La inclusión de la palabra clave, incluso en un comentario, puede hacer que AzureRm se cargue y, a continuación, entre en conflicto con el módulo Az.

Escenarios comunes disponibles en el Galería de PowerShell

La lista siguiente contiene algunos runbooks que admiten escenarios comunes. Para obtener una lista completa de runbooks creados por el equipo de Azure Automation, consulte AzureAutomationTeam profile.

- Update-ModulesInAutomationToLatestVersion: importa la versión más reciente de todos los módulos de una cuenta de Automation desde Galería de PowerShell.

- Enable-AzureDiagnostics: configura Azure Diagnostics y Log Analytics para recibir registros de Azure Automation que contienen flujos de trabajo y estado del trabajo.
- Copy-ItemFromAzureVM: copia un archivo remoto desde una máquina virtual Windows Azure.
- Copy-ItemToAzureVM: copia un archivo local en una máquina virtual Azure.

Contribución a la comunidad

Le animamos encarecidamente a contribuir y ayudar a crecer la comunidad de Azure Automation. Comparta los asombrosos runbooks que haya creado con la comunidad. Se agradecen sus contribuciones.

Adición de un runbook a la galería de runbooks de GitHub

Puede agregar nuevos runbooks de PowerShell o Python a la galería de runbooks con este flujo de trabajo de GitHub.

1. Cree un repositorio público en GitHub y agregue el runbook y cualquier otro archivo necesario (como readme.md, descripción, etc.).

2. Agregue el tema `azureautomationrunbookgallery` para asegurarse de que nuestro servicio detecta el repositorio, de modo que se pueda mostrar en la galería de runbooks de Automation.

3. Si el runbook que creó es un flujo de trabajo de PowerShell, agregue el tema `PowerShellWorkflow`. Si es un runbook de Python 3, agregue `Python3`. No se requieren otros temas específicos para Azure runbooks, pero le animamos a agregar otros temas que se pueden usar para la categorización y la búsqueda en la Galería de Runbook.

   > **Nota:**
   >
   > Consulte los runbooks existentes en la galería para ver aspectos como el formato, los encabezados y las etiquetas existentes que puede usar (como `Azure Automation` o `Linux Azure Virtual Machines`).

Para sugerir cambios en un runbook existente, realice una solicitud de incorporación de cambios en él.

Si decide clonar y editar un runbook existente, el procedimiento recomendado es darle un nombre diferente. Si vuelve a usar el nombre anterior, aparecerá dos veces en la lista de la galería

de runbooks.

> ⓘ **Nota:**
>
> Permita al menos 12 horas para la sincronización entre GitHub y automation Runbook Gallery, tanto para runbooks actualizados como nuevos.

Adición de un runbook de PowerShell a la galería de PowerShell

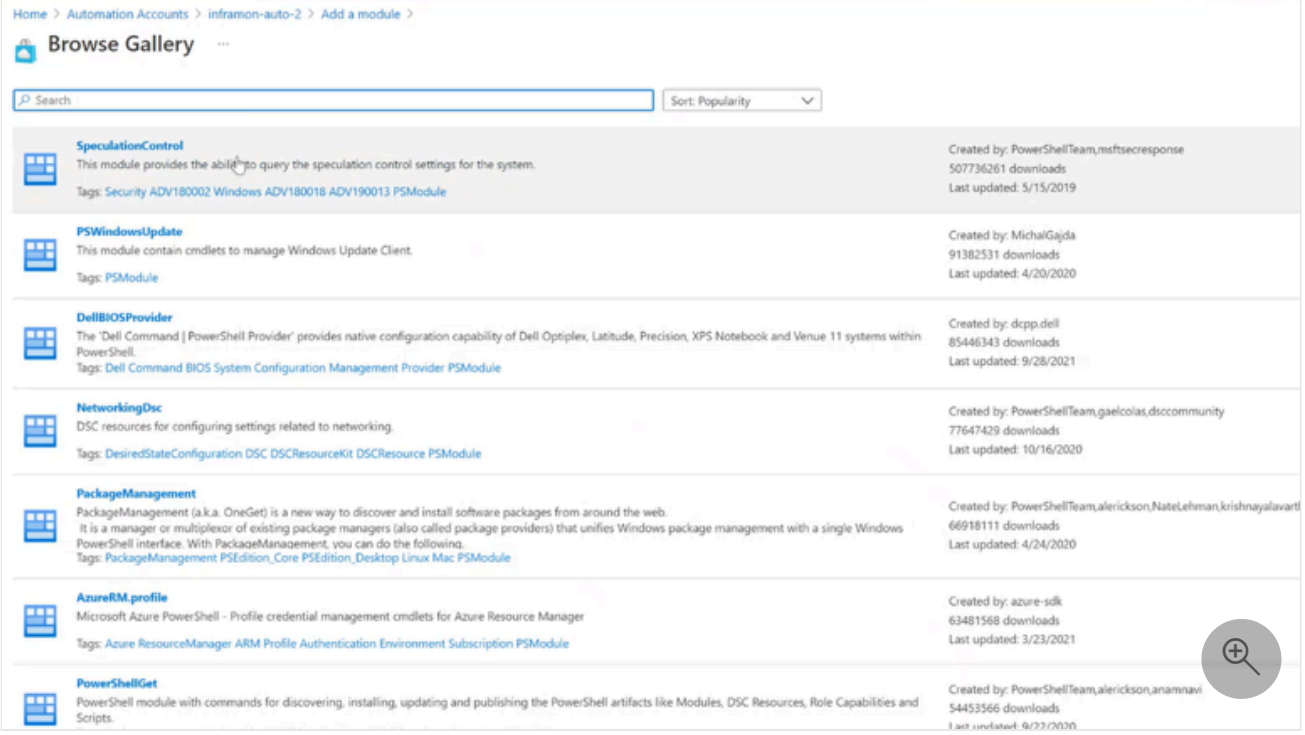
Microsoft recomienda agregar a la Galería de PowerShell aquellos runbooks que piense que podrían ser útiles para otros clientes. El Galería de PowerShell acepta módulos de PowerShell y scripts de PowerShell. Puede agregar un runbook cargándolo en la Galería de PowerShell.

Importación de un módulo desde la galería módulos en el portal de Azure

1. En el portal de Azure, abra la cuenta de Automation.

2. En **Recursos compartidos**, seleccione **Módulos**.

3. En la página **Módulos**, seleccione **Examinar galería** para abrir la lista de módulos.

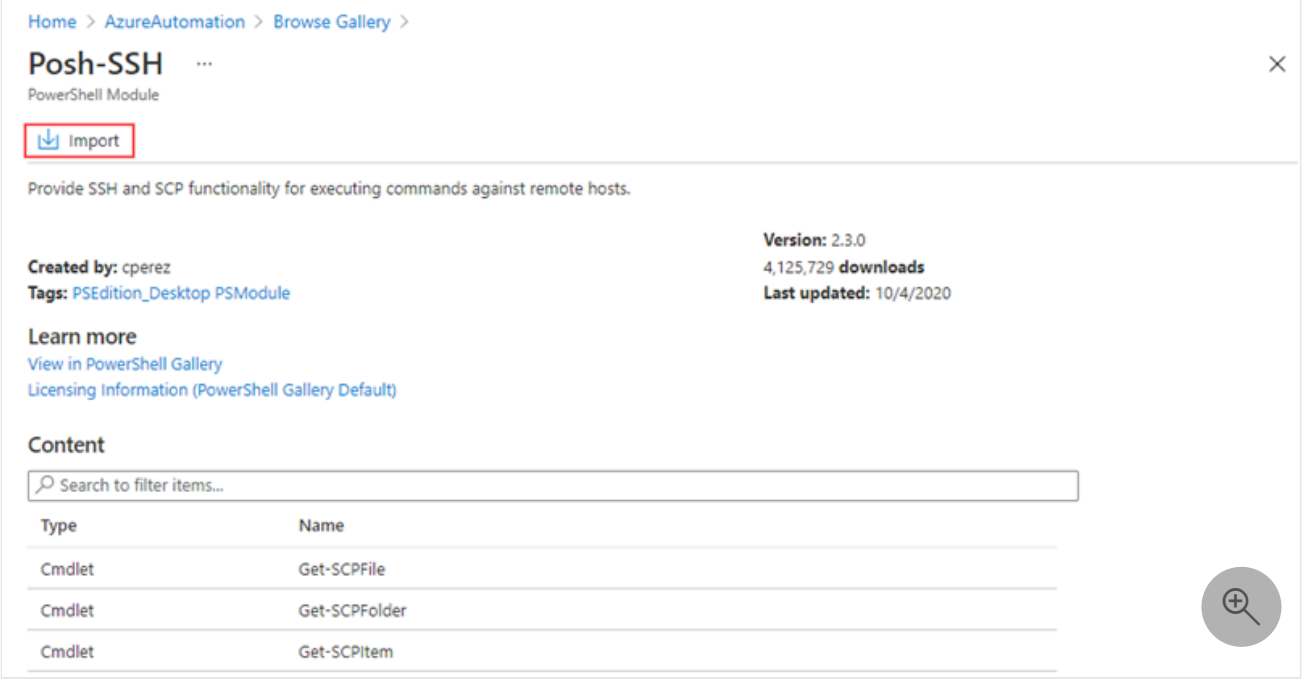


4. En la página Examinar galería, puede buscar por los campos siguientes:

- Nombre del módulo
- Etiquetas
- Autor
- Nombre de cmdlet/recurso de DSC

5. Busque un módulo que le interese y selecciónelo para ver sus detalles.

   Al explorar un módulo específico, puede ver más información. Esta información incluye un vínculo a la Galería de PowerShell, las dependencias necesarias y todos los cmdlets o recursos de DSC que contiene el módulo.



6. Para instalar el módulo directamente en Azure Automation, haga clic en **Import**.

7. En el panel Importar, puede ver el nombre del módulo que se va a importar. Si están instaladas todas las dependencias, el botón **Aceptar** estará activo. Si faltan dependencias, debe importarlas antes de importar este módulo.

8. En la página Importar, haga clic en **Aceptar** para importar el módulo. Aunque Azure Automation importa un módulo a su cuenta, extrae metadatos sobre el módulo y los cmdlets. Esta acción puede tardar un par de minutos, ya que debe extraerse cada actividad.

9. Recibirá una notificación inicial indicando que el módulo se está implementado y otra notificación cuando se haya completado.

10. Una vez importado el módulo, puede ver las actividades disponibles. Puede utilizar los recursos del módulo en sus runbooks y recursos de DSC.

> **Nota:**
>
> Los módulos que solo admiten núcleos de PowerShell no se admiten en Azure Automation y no se pueden importar en el portal de Azure ni se implementan directamente desde el Galería de PowerShell.

Solicitud de un runbook o módulo

Puede enviar solicitudes a Voz de usuario. Si necesita ayuda para escribir un manual de ejecución o tiene una pregunta sobre PowerShell, publique una pregunta en nuestra página de preguntas de Microsoft Q&A.

Pasos siguientes

- Para empezar a trabajar con runbooks de PowerShell, consulte Tutorial: Creación de un runbook de PowerShell.
- Para trabajar con runbooks, consulte Administrar runbooks en Azure Automation.
- Para más información sobre el scripting de PowerShell, consulte la documentación de PowerShell.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.
- Para solucionar problemas al usar runbooks y recursos compartidos desde la Galería de Runbooks o Galería de PowerShell en Azure Automation, consulte Solucionar problemas de recursos compartidos en Azure Automation.
- Para solucionar problemas relacionados con el uso de runbooks existentes en Azure Automation, consulte Troubleshoot runbook issues.

Last updated on 01/05/2026

Información sobre el flujo de trabajo de PowerShell para Azure Automation

Los runbooks de Azure Automation se implementan como flujos de trabajo de Windows PowerShell, scripts de Windows PowerShell que usan Windows Workflow Foundation. Un flujo de trabajo es una secuencia de pasos conectados y programados que realizan tareas de larga duración o requieren de la coordinación de pasos múltiples a través de varios dispositivos o nodos administrados.

Aunque el flujo de trabajo esté escrito con sintaxis de Windows PowerShell y se inicie mediante Windows PowerShell, se procesa mediante Windows Workflow Foundation. Las ventajas de un flujo de trabajo en un script normal incluyen el rendimiento simultáneo de una acción en varios dispositivos y la recuperación automática en caso de errores.

> **Nota:**
>
> Este artículo es aplicable a PowerShell 5.1; PowerShell 7.1 (versión preliminar) y PowerShell 7.2 (versión preliminar) no admiten flujos de trabajo. Un flujo de trabajo de PowerShell es muy similar a un script de Windows PowerShell, pero presenta algunas diferencias importantes que pueden resultar confusas para un nuevo usuario. Por lo tanto, se recomienda escribir los runbooks con el flujo de trabajo de PowerShell solo si hay que usar puntos de control.

Para más información sobre los temas de este artículo, consulte Introducción al flujo de trabajo de Windows PowerShell.

Uso de la palabra clave workflow

El primer paso para convertir un script de PowerShell en un flujo de trabajo de PowerShell es delimitarlo con la palabra clave `Workflow`. Un flujo de trabajo comienza con la palabra clave `Workflow`, seguida del cuerpo del script entre llaves. El nombre del flujo de trabajo sigue a la palabra clave `Workflow`, tal y como se muestra en la sintaxis siguiente:

PowerShell

```
Workflow Test-Workflow
{
    <Commands>
}
```

El nombre del flujo de trabajo debe coincidir con el nombre del runbook de Automation. Si se va a importar el runbook, el nombre de archivo debe coincidir con el nombre del flujo de trabajo y debe terminar en **.ps1**.

Para agregar parámetros al flujo de trabajo, use la palabra clave `Param` igual que lo haría en un script.

Información sobre las diferencias entre el código de flujo de trabajo de PowerShell y el código de script de PowerShell

El código de flujo de trabajo de PowerShell es casi idéntico al código de script de PowerShell salvo por algunos cambios importantes. En las secciones siguientes se describen los cambios que debe realizar en un script de PowerShell para que se ejecute en un flujo de trabajo.

Actividades

Una actividad es una tarea específica en un flujo de trabajo que se realiza en una secuencia. El flujo de trabajo de Windows PowerShell convierte automáticamente muchos de los cmdlets de Windows PowerShell en actividades cuando se ejecuta un flujo de trabajo. Cuando se especifica uno de estos cmdlets en el runbook, Windows Workflow Foundation ejecuta la actividad correspondiente.

Para los cmdlets sin actividad correspondiente, el flujo de trabajo de Windows PowerShell ejecuta automáticamente el cmdlet en una actividad InlineScript. Algunos cmdlets están excluidos y no se pueden usar en un flujo de trabajo a menos que se incluyan explícitamente en un bloque de InlineScript. Para más información, consulte Uso de actividades en flujos de trabajo de script.

Las actividades de flujo de trabajo comparten un conjunto de parámetros comunes para configurar su funcionamiento. Consulte about_WorkflowCommonParameters.

Parámetros posicionales

No puede usar parámetros posicionales con actividades y cmdlets en un flujo de trabajo. Por lo tanto, debe usar nombres de parámetro. Considere el siguiente código que obtiene todos los servicios en ejecución:

```
Azure PowerShell
```

```
Get-Service | Where-Object {$_.Status -eq "Running"}
```

Si intenta ejecutar este código en un flujo de trabajo, recibirá un mensaje similar a `Parameter set cannot be resolved using the specified named parameters.`. Para corregir este problema, proporcione el nombre del parámetro, como en el ejemplo siguiente:

PowerShell

```
Workflow Get-RunningServices
{
    Get-Service | Where-Object -FilterScript {$_.Status -eq "Running"}
}
```

Objetos deserializados

Los objetos de los flujos de trabajo se deserializan, lo que significa que sus propiedades siguen estando disponibles, pero no sus métodos. Por ejemplo, considere el siguiente código de PowerShell, que detiene un servicio con el método `Stop` del objeto `Service`.

Azure PowerShell

```
$Service = Get-Service -Name MyService
$Service.Stop()
```

Si intenta ejecutarlo en un flujo de trabajo, recibirá un error que indica `Method invocation is not supported in a Windows PowerShell Workflow.`.

Una opción es ajustar estas dos líneas de código en un bloque de InlineScript. En este caso, `Service` representa un objeto de servicio dentro del bloque.

PowerShell

```
Workflow Stop-Service
{
    InlineScript {
        $Service = Get-Service -Name MyService
        $Service.Stop()
    }
}
```

Otra opción es usar otro cmdlet que tenga la misma funcionalidad que el método, si hay uno disponible. En nuestro ejemplo, el cmdlet `Stop-Service` proporciona la misma funcionalidad que el método `Stop`, y se podría usar el siguiente código para un flujo de trabajo.

PowerShell

```
Workflow Stop-MyService
{
    $Service = Get-Service -Name MyService
    Stop-Service -Name $Service.Name
}
```

Uso de InlineScript

La actividad `InlineScript` es útil cuando necesita ejecutar uno o más comandos como un script tradicional de PowerShell en lugar de como un flujo de trabajo de PowerShell. Mientras que los comandos de un flujo de trabajo se envían a Windows Workflow Foundation para su procesamiento, los comandos de un bloque de InlineScript se procesan mediante Windows PowerShell.

InlineScript usa la sintaxis que se muestra a continuación.

PowerShell

```
InlineScript
{
    <Script Block>
} <Common Parameters>
```

Puede devolver resultados de InlineScript asignando el resultado a una variable. El siguiente ejemplo detiene un servicio y, a continuación, envía el nombre de servicio.

PowerShell

```
Workflow Stop-MyService
{
    $Output = InlineScript {
        $Service = Get-Service -Name MyService
        $Service.Stop()
        $Service
    }

    $Output.Name
}
```

Puede pasar valores en un bloque de InlineScript, pero debe usar el modificador de ámbito **$Using** . El ejemplo siguiente es idéntico al ejemplo anterior salvo que se proporciona el nombre de servicio mediante una variable.

PowerShell

```
Workflow Stop-MyService
{
    $ServiceName = "MyService"

    $Output = InlineScript {
        $Service = Get-Service -Name $Using:ServiceName
        $Service.Stop()
        $Service
    }

    $Output.Name
}
```

Aunque las actividades InlineScript puedan ser críticas en determinados flujos de trabajo, no admiten construcciones de flujo de trabajo. Solo debe usarlos cuando sea necesario por los siguientes motivos:

- No puede usar puntos de control dentro de un bloque de InlineScript. Si se produce un error dentro del bloque, se debe reanudar desde el principio del bloque.
- No puede usar la ejecución en paralelo dentro de un bloque de InlineScript.
- InlineScript afecta a la escalabilidad del flujo de trabajo, ya que retiene la sesión de Windows PowerShell durante todo el bloque de InlineScript.

Para más información sobre el uso de InlineScript, vea Ejecutar comandos de Windows PowerShell en un flujo de trabajo y about_InlineScript.

Uso del procesamiento paralelo

Una ventaja de los flujos de trabajo de Windows PowerShell es la capacidad para realizar un conjunto de comandos en paralelo en lugar de hacerlo secuencialmente como con un script típico.

Puede usar la palabra clave `Parallel` para crear un bloque de scripts con varios comandos que se ejecutan simultáneamente. Esto usa la siguiente sintaxis que se muestra a continuación. En este caso, Activity1 y Activity2 se inician al mismo tiempo. Activity3 se inicia después de que se hayan completado Activity1 y Activity2.

PowerShell

```
Parallel
{
    <Activity1>
    <Activity2>
}
<Activity3>
```

Por ejemplo, considere los siguientes comandos de PowerShell que copian varios archivos a un destino de red. Estos comandos se ejecutan secuencialmente de modo que un archivo debe terminar de copiarse antes de que comience el siguiente.

Azure PowerShell

```
Copy-Item -Path C:\LocalPath\File1.txt -Destination \\NetworkPath\File1.txt
Copy-Item -Path C:\LocalPath\File2.txt -Destination \\NetworkPath\File2.txt
Copy-Item -Path C:\LocalPath\File3.txt -Destination \\NetworkPath\File3.txt
```

El flujo de trabajo siguiente ejecuta estos mismos comandos en paralelo para que todos ellos empiezan a copiarse al mismo tiempo. Una vez que todos se han copiado, se muestra el mensaje de finalización.

PowerShell

```
Workflow Copy-Files
{
    Parallel
    {
        Copy-Item -Path "C:\LocalPath\File1.txt" -Destination "\\NetworkPath"
        Copy-Item -Path "C:\LocalPath\File2.txt" -Destination "\\NetworkPath"
        Copy-Item -Path "C:\LocalPath\File3.txt" -Destination "\\NetworkPath"
    }

    Write-Output "Files copied."
}
```

Puede utilizar la construcción `ForEach -Parallel` para procesar comandos para cada elemento de una colección simultáneamente. Los elementos de la colección se procesan en paralelo mientras que los comandos del bloque de scripts se ejecutan secuencialmente. Este proceso usa la siguiente sintaxis que se muestra a continuación. En este caso, Activity1 se inicia al mismo tiempo para todos los elementos de la colección. Para cada elemento, Activity2 se inicia una vez completado Activity1. Activity3 se inicia únicamente después de que se hayan completado Activity1 y Activity2 para todos los elementos. Se usa el parámetro `ThrottleLimit` para limitar el paralelismo. Un valor `ThrottleLimit` demasiado alto puede causar problemas. El valor ideal para el parámetro `ThrottleLimit` depende de muchos factores del entorno. Empiece con un valor bajo y pruebe distintos valores crecientes hasta que encuentre uno que funcione para su caso concreto.

PowerShell

```
ForEach -Parallel -ThrottleLimit 10 ($<item> in $<collection>)
{
    <Activity1>
    <Activity2>
```

```
}
<Activity3>
```

El ejemplo siguiente es similar al ejemplo anterior, en el que se copian archivos en paralelo. En este caso, se muestra un mensaje para cada archivo después de copiarse. Una vez que todos se han copiado, se muestra el mensaje de finalización.

PowerShell

```
Workflow Copy-Files
{
    $files =
@("C:\LocalPath\File1.txt","C:\LocalPath\File2.txt","C:\LocalPath\File3.txt")

    ForEach -Parallel -ThrottleLimit 10 ($File in $Files)
    {
        Copy-Item -Path $File -Destination \\NetworkPath
        Write-Output "$File copied."
    }

    Write-Output "All files copied."
}
```

> ⓘ **Nota:**
>
> No se recomienda ejecutar runbooks secundarios en paralelo, ya que se ha demostrado que no tiene unos resultados confiables. A veces el resultado del runbook secundario no se muestra, y la configuración de un runbook secundario puede afectar a los demás runbooks secundarios paralelos. Variables tales como `VerbosePreference`, `WarningPreference` y otras podrían no propagarse a los runbooks secundarios. Y si el runbook secundario cambia estos valores, es posible que no se restauren correctamente después de su invocación.

Uso de puntos de control en un flujo de trabajo

Un punto de control es una instantánea del estado actual del flujo de trabajo que incluye el valor actual de las variables y cualquier salida generada en ese punto. Si un flujo de trabajo termina en error o se suspende, comienza desde su último punto de control la próxima vez que se ejecute, en lugar de comenzar desde el principio.

Puede establecer un punto de control en un flujo de trabajo con la actividad `Checkpoint-Workflow`. Azure Automation tiene una característica denominada distribución equilibrada, por la que se descarga cualquier runbook que se ejecute durante tres horas para permitir la

ejecución de otros runbooks. Finalmente, se recarga el runbook descargado. Cuando es así, se reanuda la ejecución desde el último punto de control tomado en el runbook.

Con el fin de garantizar que el runbook finalmente se complete, debe agregar puntos de control en intervalos que se ejecuten durante menos de tres horas. Si durante cada ejecución se agrega un nuevo punto de control, y si el runbook se expulsa después de tres horas debido a un error, el runbook se reanudará indefinidamente.

En el ejemplo siguiente, se produce una excepción después de Activity2, lo que provoca la finalización del flujo de trabajo. Cuando se vuelve a ejecutar el flujo de trabajo, se inicia ejecutando Activity2, ya que estaba inmediatamente después del último punto de control establecido.

PowerShell

```
<Activity1>
Checkpoint-Workflow
<Activity2>
<Exception>
<Activity3>
```

Establezca puntos de control en un flujo de trabajo después de actividades que puedan ser propensas a la excepción y que no deben repetirse si se reanuda el flujo de trabajo. Por ejemplo, el flujo de trabajo puede crear una máquina virtual. Puede establecer un punto de control antes y después de los comandos para crear la máquina virtual. Si se produce un error en la creación, los comandos se repiten si el flujo de trabajo se vuelve a iniciar. Si el flujo de trabajo produce un error después de que la creación se realice correctamente, la máquina virtual no se vuelve a crear cuando se reanude el flujo de trabajo.

El siguiente ejemplo copia varios archivos en una ubicación de red y establece un punto de comprobación después de cada archivo. Si la ubicación de red se pierde, el flujo de trabajo finaliza con error. Cuando se vuelve a iniciar, se reanuda en el último punto de control. Solo se omiten los archivos que ya se han copiado.

PowerShell

```
Workflow Copy-Files
{
    $files =
@("C:\LocalPath\File1.txt","C:\LocalPath\File2.txt","C:\LocalPath\File3.txt")

    ForEach ($File in $Files)
    {
        Copy-Item -Path $File -Destination \\NetworkPath
        Write-Output "$File copied."
        Checkpoint-Workflow
    }
```

```
    Write-Output "All files copied."
}
```

Como las credenciales de nombre de usuario no se conservan después de llamar a la actividad Suspend-Workflow o después del último punto de control, tiene que establecer las credenciales en NULL y, después, recuperarlas de nuevo desde el almacén de recursos tras llamar a `Suspend-Workflow` o al punto de control. En caso contrario, es posible que reciba el siguiente mensaje de error: `The workflow job cannot be resumed, either because persistence data could not be saved completely, or saved persistence data has been corrupted. You must restart the workflow.`.

El mismo código de abajo muestra cómo controlar esta operación en los runbooks del flujo de trabajo de PowerShell.

PowerShell

```
workflow CreateTestVms
{
    $Cred = Get-AzAutomationCredential -Name "MyCredential"
    $null = Connect-AzAccount -Credential $Cred

    $VmsToCreate = Get-AzAutomationVariable -Name "VmsToCreate"

    foreach ($VmName in $VmsToCreate)
        {
        # Do work first to create the VM (code not shown)

        # Now add the VM
        New-AzVM -VM $Vm -Location "WestUs" -ResourceGroupName "ResourceGroup01"

        # Checkpoint so that VM creation is not repeated if workflow suspends
        $Cred = $null
        Checkpoint-Workflow
        $Cred = Get-AzAutomationCredential -Name "MyCredential"
        $null = Connect-AzAccount -Credential $Cred
        }
}
```

Nota:

En el caso de los runbooks de PowerShell no gráficos, `Add-AzAccount` y `Add-AzureRMAccount` son alias de Connect-AzAccount. Puede usar estos cmdlets o bien actualizar los módulos de la cuenta de Automation a las versiones más recientes. Es posible que deba actualizar los módulos incluso si acaba de crear una nueva cuenta de Automation.

Para obtener más información acerca de los puntos de control, consulte Adición de puntos de control a un flujo de trabajo de scripts.

Pasos siguientes

- Para obtener información sobre los runbooks de flujo de trabajo de PowerShell, consulte Tutorial: Creación de un runbook de flujo de trabajo de PowerShell.

Last updated on 17/11/2025

Administración del entorno en tiempo de ejecución y runbooks asociados

09/07/2025

En este artículo se proporciona información sobre cómo crear el entorno en tiempo de ejecución y realizar varias operaciones mediante Azure Portal y la API de REST.

Requisitos previos

Antes de asignar permisos a identidades administradas, asegúrese de cumplir estos requisitos previos:

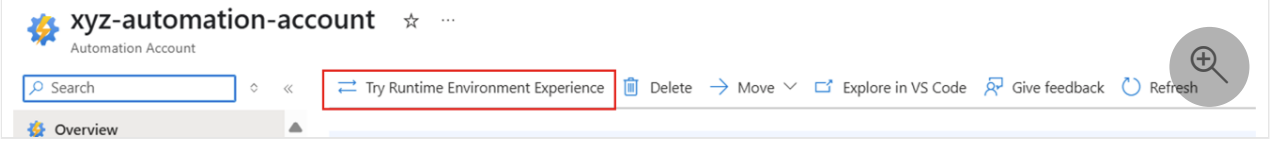
- Una cuenta de Azure Automation en todas las regiones públicas, excepto Brasil Sudeste y las nubes gubernamentales.

Cambio entre el entorno en tiempo de ejecución y la experiencia anterior

Experiencia del entorno en tiempo de ejecución

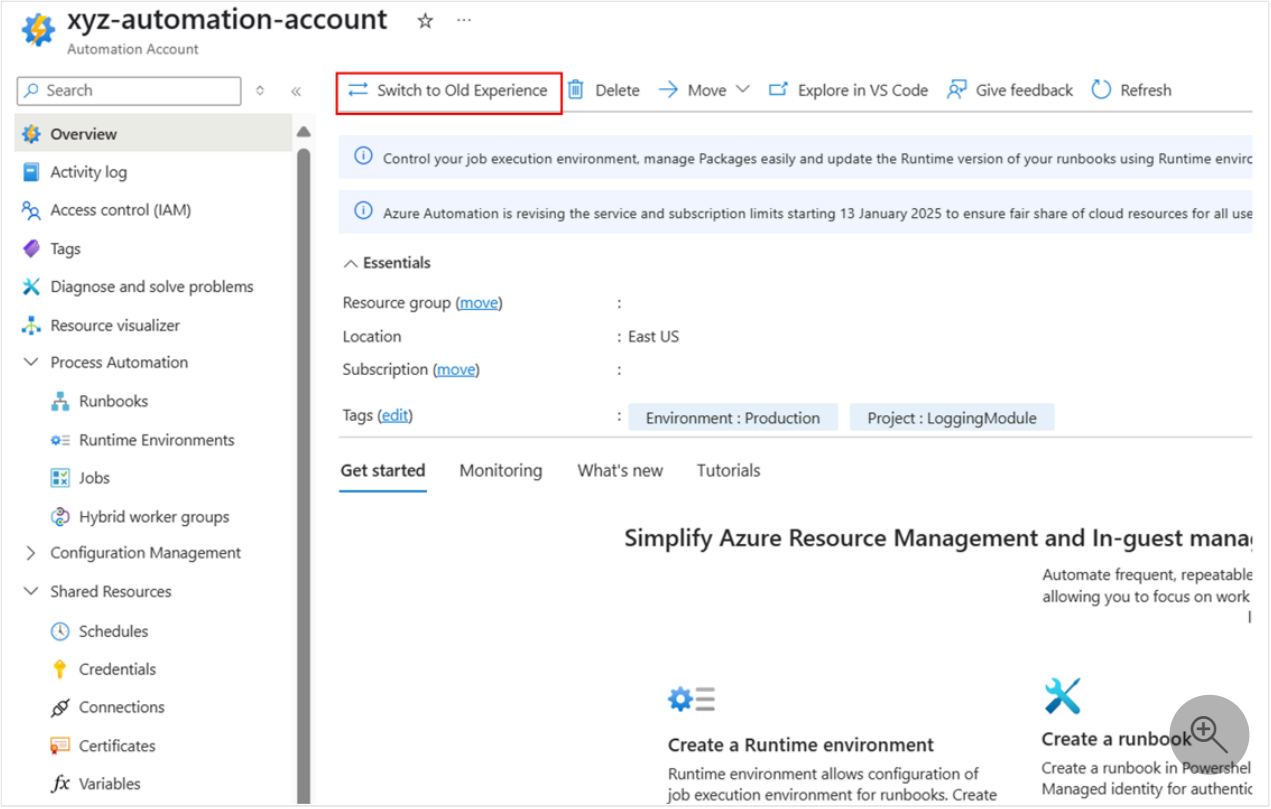
Para cambiar al entorno en tiempo de ejecución, siga estos pasos:

1. Seleccione **Información general** y, a continuación, **Probar experiencia del entorno en tiempo de ejecución**.



2. En **Automatización de procesos**, tiene los **entornos de ejecución** y los **módulos** y **paquetes de Python** se eliminan en **Recursos compartidos** porque el entorno de ejecución permite la gestión de paquetes necesarios durante la ejecución de Runbook.

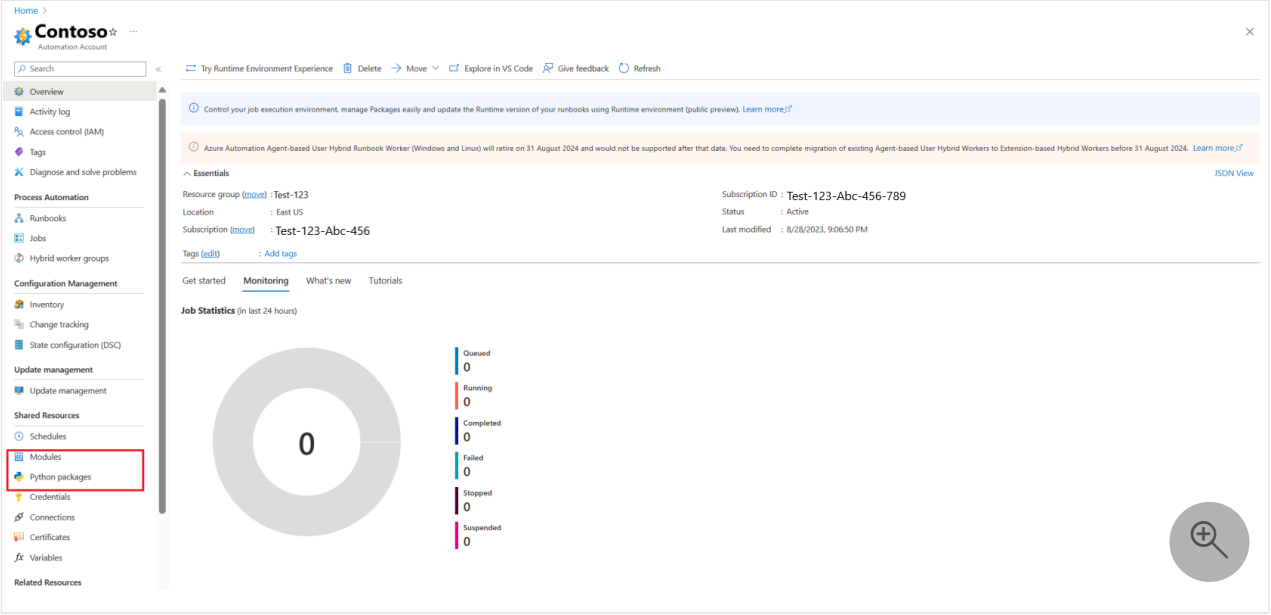
3. Para revertir a la experiencia anterior para administrar módulos y paquetes, seleccione **Información general** en el panel izquierdo y, a continuación, **Cambiar a la experiencia anterior**.



Experiencia anterior

Para cambiar a la experiencia anterior, siga este paso:

1. Vaya a la cuenta de Automation, en **Automatización** de procesos, los **entornos** en tiempo de ejecución se quitan y, en **Recursos compartidos**, se pueden ver **módulos** y **paquetes de Python** .



> ⓘ **Nota**

Las actualizaciones del runbook persisten entre la nueva experiencia del entorno en runtime y la experiencia anterior. Los cambios realizados en el entorno en tiempo de ejecución vinculados a un runbook se conservarían durante la ejecución del runbook en la experiencia anterior.

Administración de entornos en tiempo de ejecución

Creación de entornos de ejecución

Azure Portal

Para crear un entorno en tiempo de ejecución, siga estos pasos:

1. Inicie sesión en Azure Portal y seleccione su cuenta de Automation.
2. En la página **Información general**, seleccione **Probar la experiencia del entorno de ejecución**, si aún no está en la nueva experiencia.
3. En **Automatización de** procesos, seleccione **Entornos en tiempo de ejecución** y, a continuación, seleccione **Crear**.
4. En la pestaña **Datos básicos**, escriba la siguientes información:

   a. **Nombre** del entorno en tiempo de ejecución. Debe comenzar con una letra y solo puede contener letras, números, caracteres de subrayado y guiones.

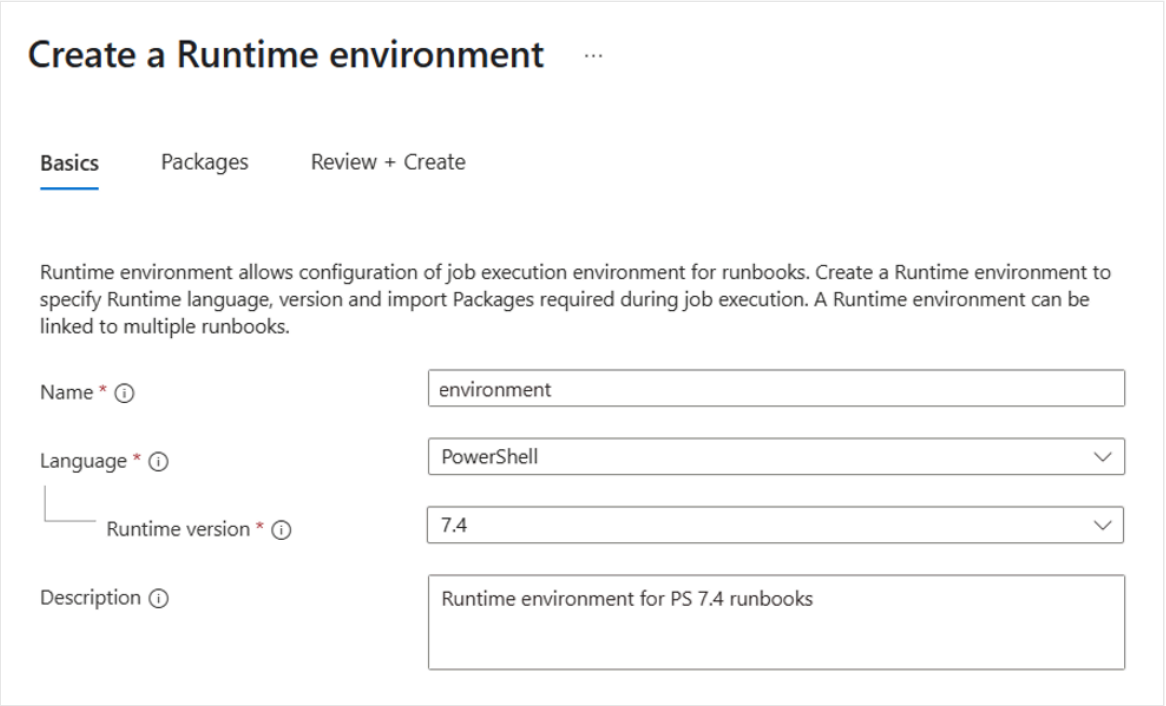
   b. En la lista desplegable **Lenguaje**, seleccione el lenguaje de scripting para el entorno en tiempo de ejecución.

      - Elija **PowerShell** para el lenguaje de scripting de PowerShell o **Python** para el lenguaje de scripting de Python.

   c. Seleccione **Versión en tiempo de ejecución** para el lenguaje de scripting.

      - Para PowerShell: elija 5.1, 7.4.
      - Para Python: elija 3.10.

   d. Escriba la **descripción** adecuada.



5. Seleccione **Siguiente** y, en la pestaña **Paquetes**, cargue los paquetes necesarios durante la ejecución del runbook. El *paquete Az de PowerShell* se carga de forma predeterminada para todos los entornos en tiempo de ejecución de PowerShell, lo que incluye todos los cmdlets para administrar recursos de Azure.

> **Nota**
>
> Los comandos de la CLI de Azure se admiten en runbooks asociados con el entorno en tiempo de ejecución de PowerShell 7.4. Los comandos de la CLI de Azure versión 2.64.0 están disponibles como un paquete predeterminado en el entorno en tiempo de ejecución de PowerShell 7.4.

6. Para cargar más paquetes necesarios durante la ejecución del runbook: Seleccione **Agregar un archivo** para agregar los archivos almacenados localmente en el equipo o seleccione **Agregar desde la galería** para cargar paquetes desde la Galería de PowerShell.

> **Nota**
>
> - La importación de un paquete podría tardar varios minutos. El tamaño total máximo de los archivos que puede importar es de 100 MB.
> - Utiliza archivos *.zip* para tipos de runbook de PowerShell, como se indica aquí.

- Para los paquetes de Python 3.10, use archivos .whl destinados al sistema operativo Linux cp310.

7. Seleccione **Siguiente** y, en la pestaña **Revisar y crear**, compruebe que la configuración sea correcta. Al seleccionar **Crear**, Azure ejecuta la validación en la configuración del entorno en tiempo de ejecución que ha elegido. Si se supera la validación, puede proceder a crear el entorno en tiempo de ejecución; de lo contrario, el portal indica la configuración que necesita modificar.

En la página **Entornos** de tiempo de ejecución, puede ver el entorno en tiempo de ejecución recién creado para la cuenta de Automation. Si no encuentra los entornos en tiempo de ejecución recién creados en la lista, seleccione **Actualizar**.

Visualización del entorno en tiempo de ejecución

Obtenga las propiedades del entorno en tiempo de ejecución de la cuenta de Automation.

REST

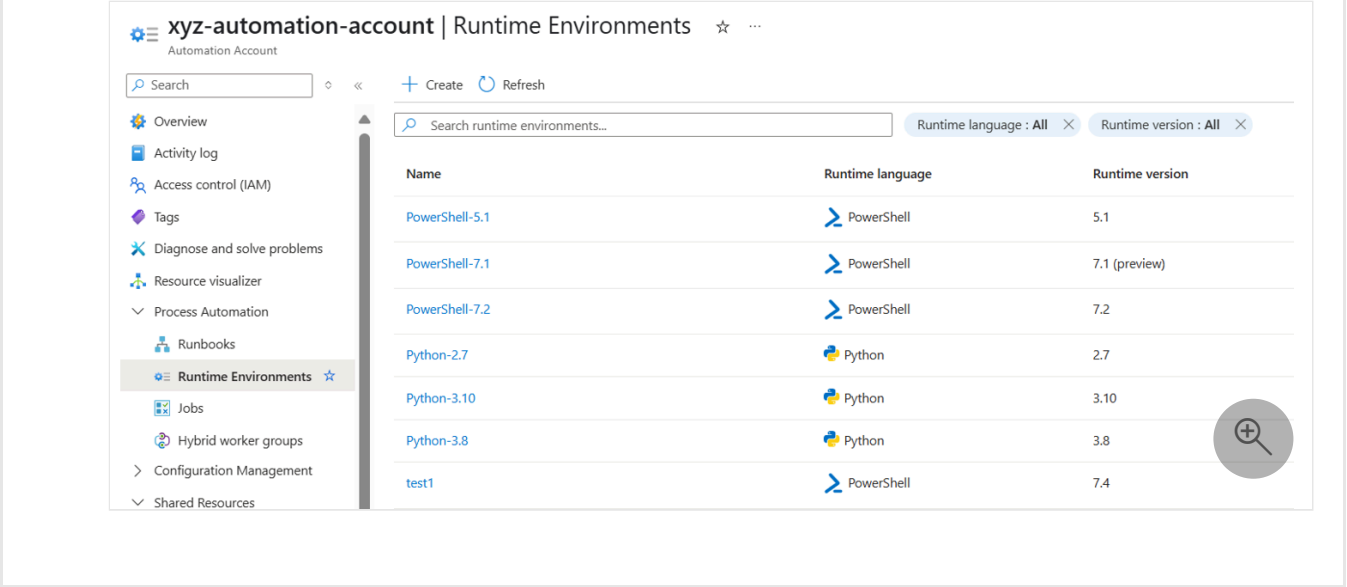
```
GET
https://management.azure.com/subscriptions/<subscriptionId>/resourceGroups/<resourceGroup>/providers/Microsoft.Automation/automationAccounts/<accountName>/runtimeEnvironments/<runtimeEnvironmentName>?api-version=2024-10-23
```

Enumeración de entornos en tiempo de ejecución

Azure Portal

Para enumerar todos los entornos en tiempo de ejecución de la cuenta de Automation, siga este paso:

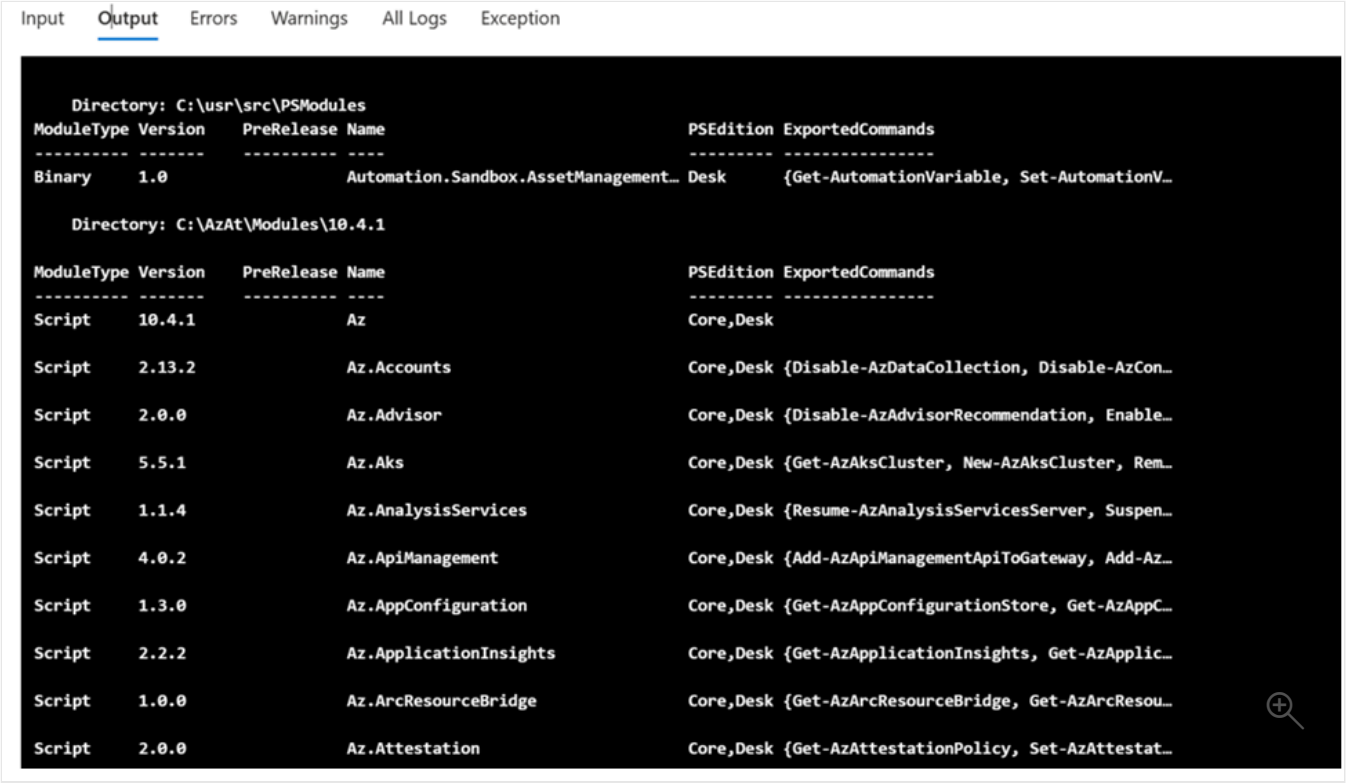
1. En tu cuenta de Automatización, bajo **Automatización de procesos**, selecciona **Entornos de ejecución**.



Visualización de componentes del paquete Az de PowerShell

Para ver todos los paquetes de componentes del *paquete Az de PowerShell*, ejecute el cmdlet `Get-Module -ListAvailable` en un runbook de PowerShell.

La salida del trabajo mostraría todos los paquetes de componentes y sus versiones.

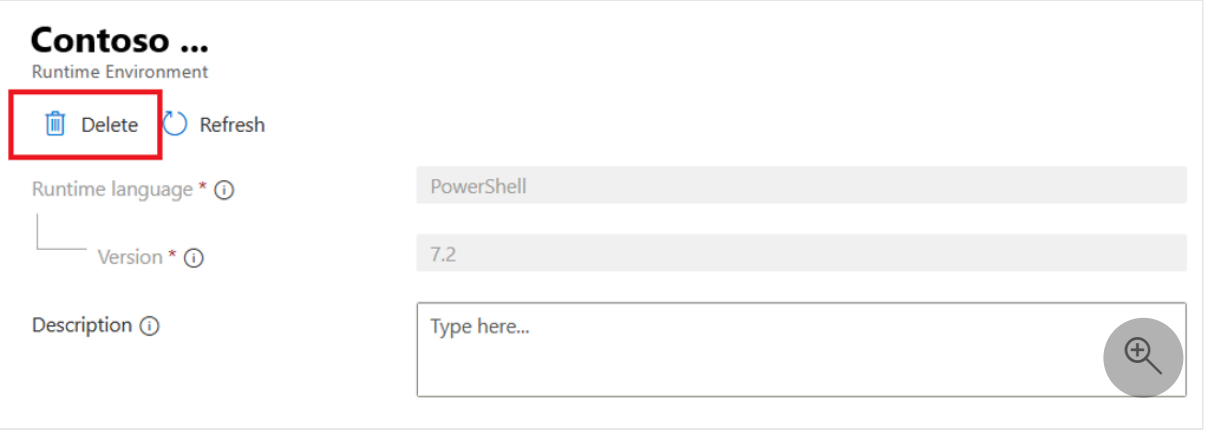


Eliminación de entornos en tiempo de ejecución

Azure Portal

Para eliminar el entorno en tiempo de ejecución de la cuenta de Automation, siga estos pasos:

1. En su cuenta de Automation, bajo **Process Automation**, seleccione **Entornos en tiempo de ejecución**.
2. Seleccione el entorno en tiempo de ejecución que desea eliminar.
3. Seleccione **Eliminar** para eliminar el entorno en tiempo de ejecución.



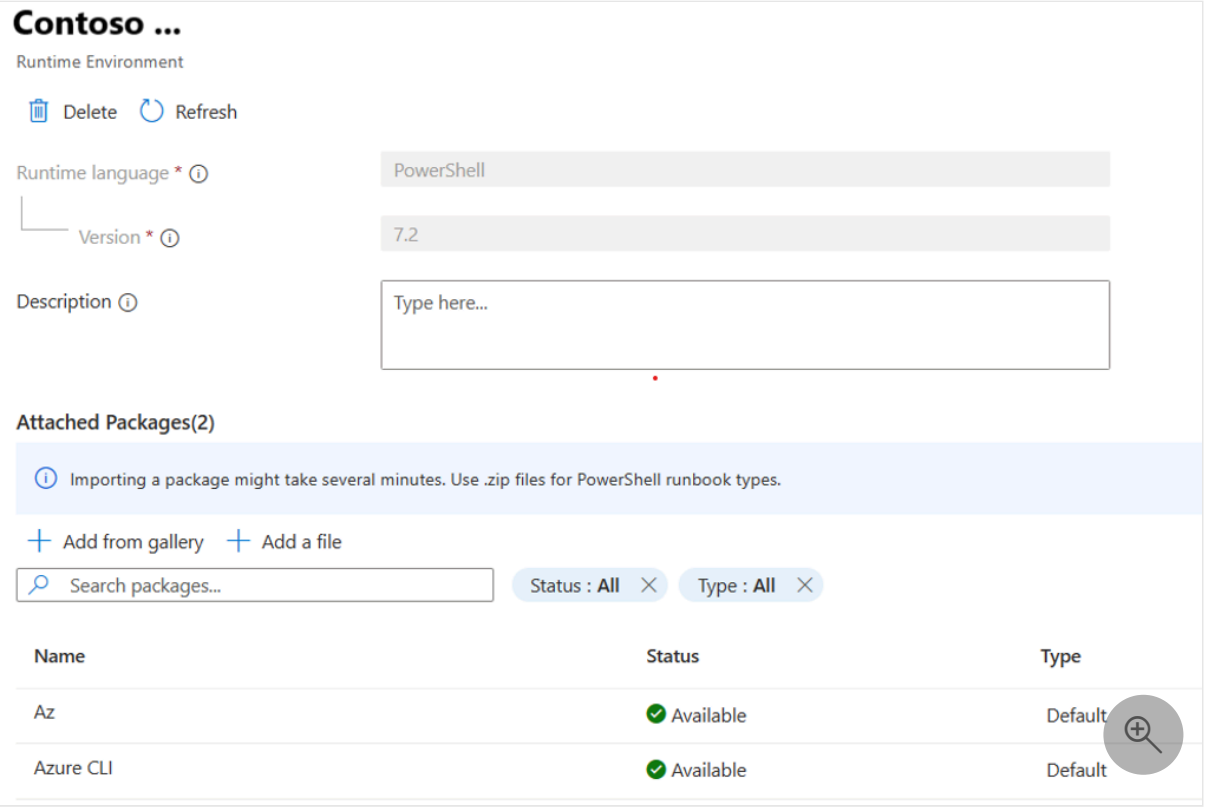
Actualización de entornos en tiempo de ejecución

El lenguaje en tiempo de ejecución y la versión en tiempo de ejecución son propiedades inmutables. Sin embargo, puede actualizar la versión de los módulos y agregar o quitar paquetes en el entorno en tiempo de ejecución. Los runbooks vinculados al entorno en tiempo de ejecución se actualizan automáticamente con la nueva configuración.

Azure Portal

Para actualizar un entorno en tiempo de ejecución, siga estos pasos:

1. En su cuenta de Automatización, bajo **Automatización de procesos**, seleccione **Entornos de ejecución**.
2. Seleccione el entorno en tiempo de ejecución que desea actualizar.
3. Seleccione la versión de la lista desplegable para actualizar la versión de los paquetes existentes.
4. Seleccione **Guardar**.



5. Seleccione **Agregar un archivo** para cargar paquetes desde el equipo local o **Agregar desde la galería** para cargar paquetes desde la Galería de PowerShell.

> **Nota**
>
> Puede agregar hasta 10 paquetes a la vez al entorno en tiempo de ejecución. Asegúrese de hacer clic en **Guardar** después de agregar 10 paquetes.

Administración de runbooks vinculados al entorno en tiempo de ejecución

Crear Runbook

Puede crear un nuevo runbook de PowerShell configurado con el entorno en tiempo de ejecución.

Azure Portal

Requisito previo

- Asegúrese de que ha creado un entorno en tiempo de ejecución antes de crear un runbook.

Para crear un nuevo runbook vinculado al entorno en tiempo de ejecución, siga estos pasos:

1. En la cuenta de Automation, en **Automatización de procesos**, seleccione **Runbooks**.

2. Seleccione **Crear**.

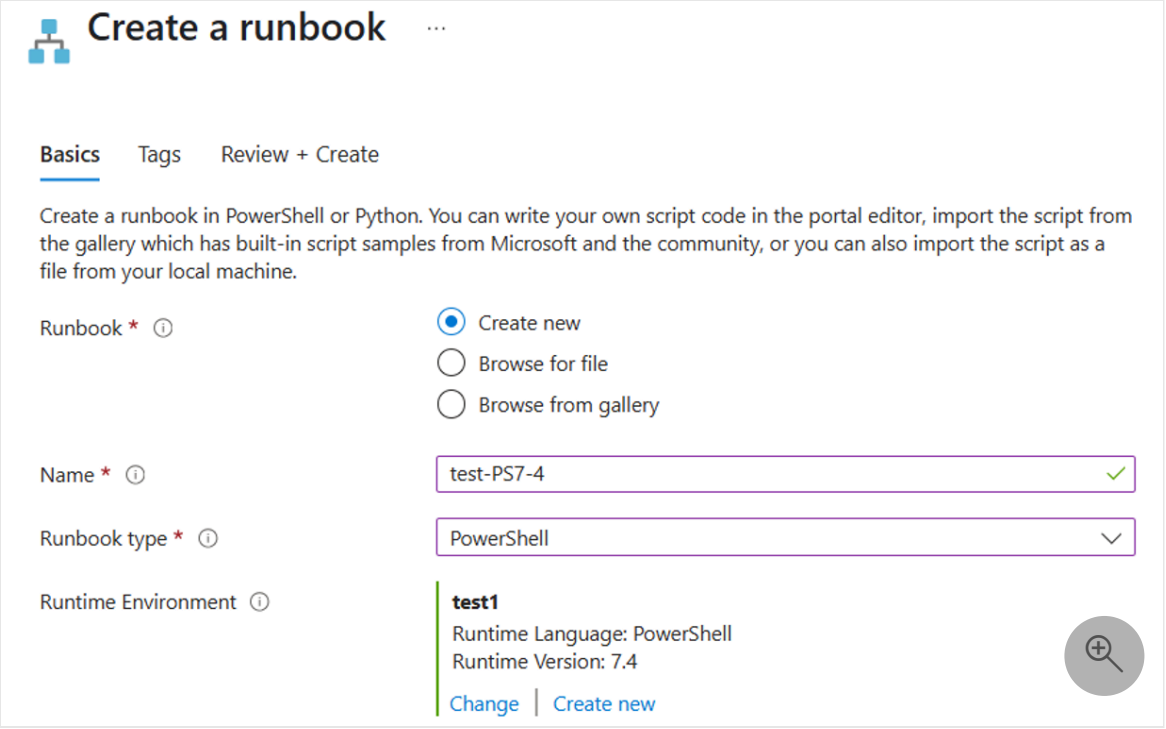
3. En la pestaña **Básicos**, puede crear un nuevo runbook o cargar un archivo desde el equipo local o desde la galería de PowerShell.

   a. Escriba un **Nombre** para el runbook. El nombre debe empezar por una letra, y debe contener solo letras, números, caracteres de subrayado y guiones.

   b. En la lista desplegable **Tipo de runbook**, seleccione el tipo de runbook que desea crear.

   c. Seleccione el **entorno en tiempo de ejecución** que se va a configurar para el runbook. Puede hacer clic en **Seleccionar entre los existentes** para seleccionar entornos en tiempo de ejecución ya existentes o **Crear nuevo** para crear un nuevo entorno en tiempo de ejecución y vincularlo al runbook. La lista de entornos en tiempo de ejecución se rellena en función del *tipo de runbook* seleccionado en el paso b.

   d. Escriba la **descripción** adecuada.



> **Nota**
>
> En **Tipo de runbook de PowerShell**, solo se mostrará y puede elegir el entorno en tiempo de ejecución de PowerShell. En **Tipo de runbook de Python**, solo se mostrarán y se pueden elegir los entornos en tiempo de ejecución de Python.

4. Agregue **etiquetas** al runbook, revise la configuración y seleccione **Crear** para crear un nuevo runbook.

Este runbook está vinculado al entorno en tiempo de ejecución seleccionado. Todos los paquetes del entorno en tiempo de ejecución elegido estarían disponibles durante la ejecución del runbook.

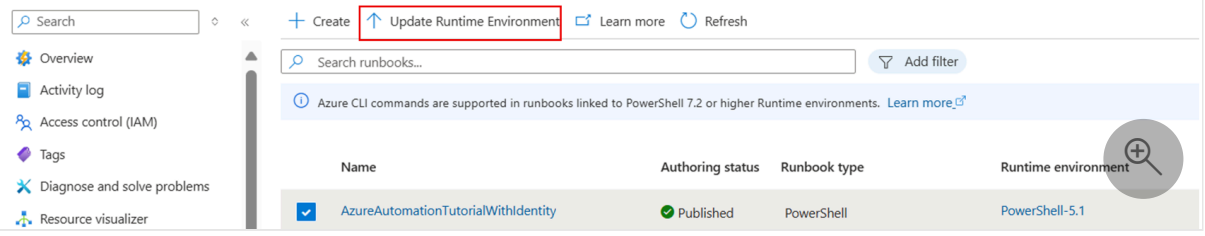
Actualización del runbook

Puede actualizar el runbook cambiando el entorno en tiempo de ejecución vinculado. Puede elegir runbooks únicos o varios para la actualización. El runbook en estado de ejecución no se vería afectado al cambiar el entorno en tiempo de ejecución vinculado a ese runbook.

Azure Portal

Para actualizar un runbook, siga estos pasos:

1. En la cuenta de Automation, en **Automatización de procesos**, seleccione **Runbooks**.
2. Active la casilla de los runbook que desea actualizar.
3. Seleccione **Actualizar entorno de tiempo de ejecución**.



4. En la lista desplegable, seleccione el entorno en tiempo de ejecución al que desea vincular los runbooks.
5. Seleccione **Actualizar** para actualizar los runbooks seleccionados con el nuevo entorno en tiempo de ejecución.

Compruebe si el runbook se ejecuta según lo previsto después de la actualización. Si el runbook no proporciona el resultado esperado, puede actualizar de nuevo el entorno en

tiempo de ejecución a la experiencia anterior siguiendo los pasos del 1 al 4.

> **Nota**
>
> Las actualizaciones del runbook persisten entre la nueva experiencia del entorno en runtime y la experiencia anterior. Los cambios realizados en el entorno en tiempo de ejecución vinculados a un runbook se conservarían durante la ejecución del runbook en la experiencia anterior.

Prueba de la actualización del runbook

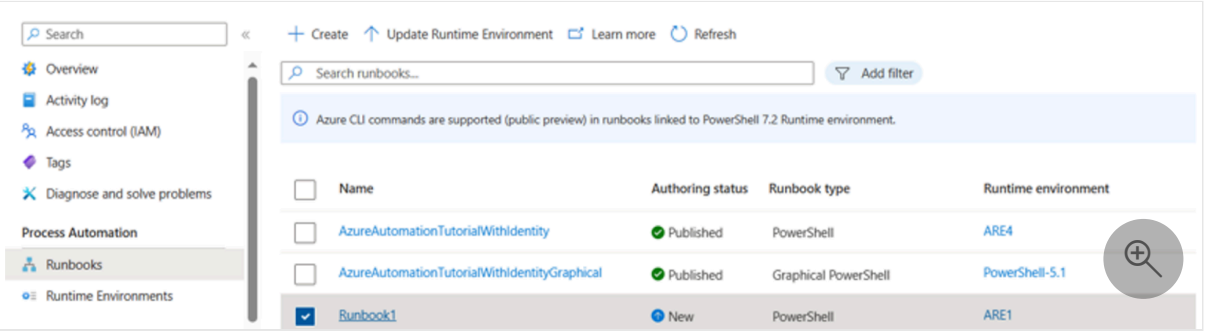
Ejecute un trabajo de prueba para un runbook con un entorno en tiempo de ejecución diferente. Este escenario es útil cuando un runbook debe probarse con un entorno en tiempo de ejecución antes actualizarlo.

Azure Portal

Puede actualizar runbooks cambiando el entorno en tiempo de ejecución vinculado a ese runbook. Se recomienda probar la ejecución del runbook antes de publicar los cambios. Esto es para asegurarse de que el runbook funciona según lo previsto.

Para probar la ejecución del runbook antes de publicar cambios en el entorno en tiempo de ejecución, siga estos pasos:

1. Vaya a la página **Runbooks** y seleccione el runbook para el que desea actualizar el entorno de ejecución.



2. En la página **Editar runbook**, seleccione **Editar en el portal** para abrir el editor de texto.

3. El campo Entorno de tiempo de ejecución muestra la configuración existente. Seleccione el nuevo entorno en tiempo de ejecución en la lista desplegable.

4. Seleccione **Panel de prueba** para probar la ejecución del runbook con el entorno en tiempo de ejecución actualizado. Seleccione **Iniciar** para iniciar la prueba.
5. Cierre el panel de prueba y realice los cambios necesarios en el código del runbook o en el entorno en tiempo de ejecución si es necesario.
6. Seleccione **Publicar** para actualizar el entorno en tiempo de ejecución vinculado al runbook.
7. Seleccione **Sí** para invalidar la versión publicada anteriormente y confirmar los cambios.

Creación de trabajos en la nube

Azure Portal

Puede ejecutar runbooks vinculados al entorno en tiempo de ejecución tanto en Azure como en Hybrid Worker.

Vinculación de runbooks existentes a entornos en tiempo de ejecución generados por el sistema

Todos los runbooks existentes de la cuenta de Azure Automation se vincularían automáticamente a entornos en tiempo de ejecución generados por el sistema. Estos entornos se crean en función del lenguaje en tiempo de ejecución, la versión y los módulos o paquetes presentes en la cuenta de Azure Automation correspondiente. Learn more. Para actualizar los runbooks existentes, cambie el entorno en tiempo de ejecución siguiendo los pasos mencionados aquí.

Pasos siguientes

- Para obtener información general, consulte Entorno en tiempo de ejecución.
- Para solucionar problemas con la ejecución de un runbook, vea Solución de problemas relativos a errores con runbooks.

Administración de límites y cuotas de suscripción de Azure Automation

En este artículo se proporcionan pasos para ver los límites actuales y solicitar el aumento o la disminución de la cuota.

Requisitos previos para los cambios de cuota

- Una cuenta de Azure Automation activa
- Rol de colaborador para cuenta de Azure Automation.

> **Nota:**
>
> Las suscripciones gratuitas, incluidas Azure cuenta gratuita y Azure for Students no son aptas para cambios de límite o cuota.

Visualizar límites actuales y solicitud de cambios de cuota

Puede ver los límites y el uso actuales, solicitar un aumento o disminución de la cuota para el número de cuentas de Automation por suscripción y el número de trabajos que se ejecutan simultáneamente por cuenta de Automation. Para más información, consulte la lista completa de límites de servicio

Siga los pasos para ver la cuota actual y solicitar cambios en la cuota:

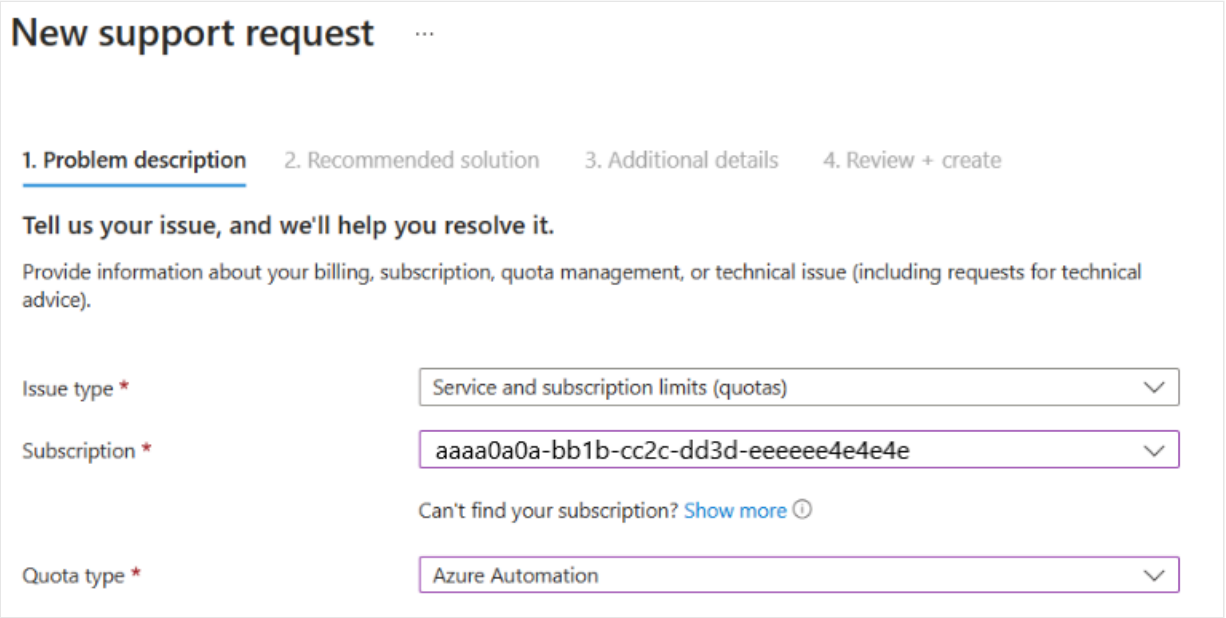
1. Seleccione **Ayuda y soporte técnico** y, a continuación, **Nueva solicitud de soporte técnico**.

2. En la página **Nueva solicitud de soporte técnico**, seleccione la pestaña **Descripción del problema** y proporcione los detalles siguientes:

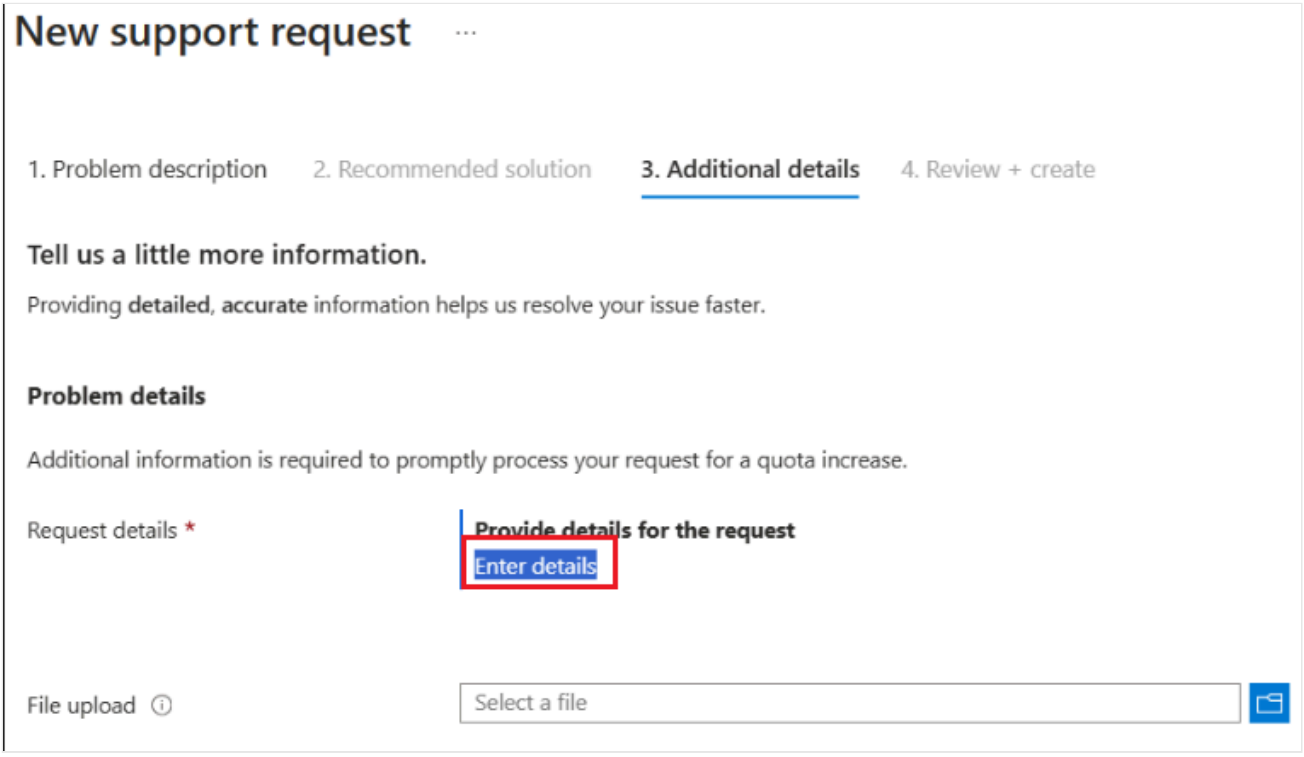
    - En **Tipo de problema**, seleccione *Límites de servicio y suscripción (cuotas)*.

    - En **Suscripción**, seleccione la suscripción para la que desea comprobar la cuota

    - En **Tipo de cuota**, seleccione *Azure Automation*.



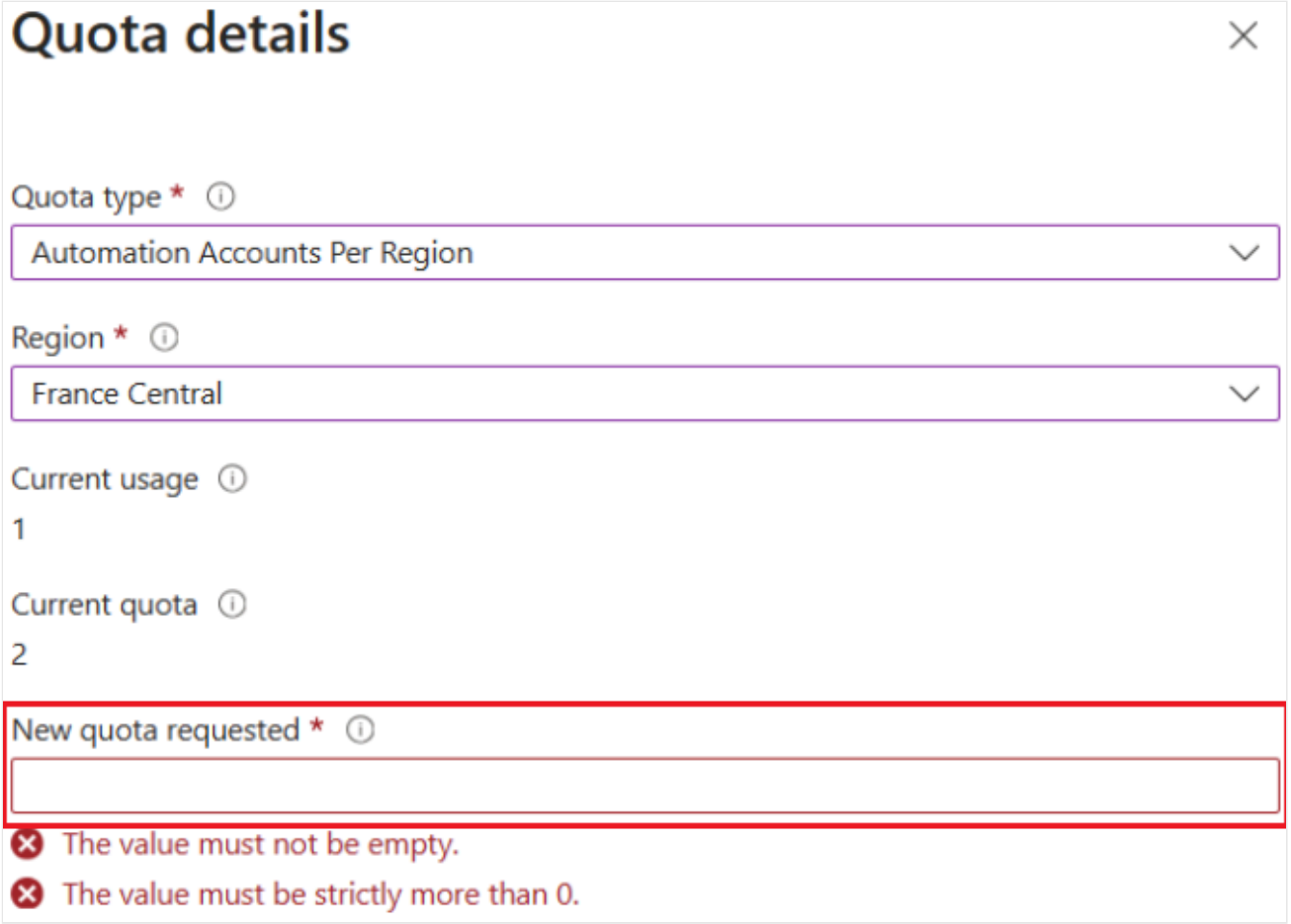
3. En la pestaña **Detalles adicionales**, en la sección **Detalles del problema**, seleccione **Especificar detalles**



4. En el panel lateral, seleccione el recurso para el que desea ver la cuota actual y solicitar el aumento o la disminución de la cuota. Puede seleccionar *la Cuenta de Automatización por región o Trabajos simultáneos por cuenta*.

5. Seleccione la **Región** para ver el **Uso actual** y la **Cuota actual** del recurso seleccionado.

6. Proporcione un valor en la **Nueva cuota solicitada** según sus necesidades para aumentar o disminuir la cuota.



7. Seleccione **Guardar y continuar**

8. Proporcione los detalles restantes para crear la solicitud de soporte técnico.

9. Seleccione **Siguiente:Revisar y crear** para validar la información proporcionada y, a continuación, seleccione **Crear** para crear una solicitud de soporte técnico.

> ⓘ **Nota:**
>
> Los aumentos de cuota están sujetos a disponibilidad de recursos en la región seleccionada.

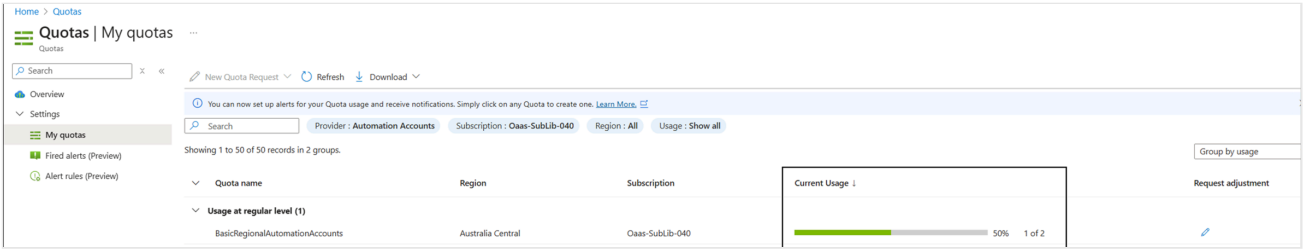
Visualización de los límites actuales y solicitud de aumento en la página Cuotas

También puede ver el uso actual y solicitar el aumento o la disminución de la cuota para el número de cuentas de Automation por suscripción en la página Cuotas de Azure Portal. Esta funcionalidad no está disponible actualmente para ver el número de trabajos que se ejecutan simultáneamente en la cuenta de Automation.

Siga estos pasos para ver los límites actuales y solicitar el aumento de la cuota:

1. Vaya a la página Mis cuotas y elija el proveedor **Cuentas de Automatización**. Las opciones de filtro de la parte superior de la página permiten filtrar por ubicación, suscripción y uso.

2. Puede ver el uso actual y el límite del número de Cuentas de Automatización.



3. Seleccione el icono de lápiz de la columna **Ajuste de solicitud** para solicitar cuota adicional.

4. En el panel **Solicitud de cuota nueva**, escriba el **nuevo límite** para el número de cuentas de Automation según sus necesidades empresariales.

5. Seleccione **Submit** (Enviar). La solicitud puede tardar unos minutos en procesarse.

    - Si se rechaza la solicitud, seleccione **Crear una solicitud de soporte técnico**. Algunos campos se rellenan automáticamente. Complete el resto de detalles de la solicitud de soporte técnico y envíela.

Pasos siguientes

- Obtenga más información sobre las cuotas o límites predeterminados que se ofrecen a distintos recursos en Azure Automation.
- Para solucionar problemas relacionados con los límites de suscripción y cuotas de la cuenta de Azure Automation, consulte Solución de incidencias de runbooks.

Last updated on 07/05/2026

Administración de runbooks en Azure Automation

Para agregar un runbook a Azure Automation, puede crear uno o importar uno existente desde un archivo o desde la Galería de runbooks. En este artículo se proporciona información sobre la administración de un runbook, así como sobre patrones recomendados y procedimientos recomendados para el diseño de runbooks. Puede encontrar todos los detalles sobre el acceso a los módulos y los runbooks de la comunidad en Galerías de runbooks y módulos para Azure Automation.

Crear un manual de procedimientos

Cree un runbook nuevo en Azure Automation mediante Azure Portal o PowerShell. Una vez creado el runbook, se puede editar siguiendo la información de las secciones siguientes:

- Editar un runbook textual en Azure Automation
- Aprendizaje de los conceptos clave del flujo de trabajo de PowerShell para los runbooks de Automation
- Administración de paquetes de Python 2 en Azure Automation
- Administración de paquetes de Python 3 (versión preliminar) en Azure Automation

Crear un runbook en el portal de Azure

1. Inicie sesión en Azure Portal.
2. Busque y seleccione **Cuentas de automatización**.
3. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
4. Desde la cuenta de Automation, seleccione **Runbooks** en **Automatización de procesos** para abrir la lista de runbooks.
5. Haga clic en **Crear un libro de procedimientos**.
   a. Asigne un nombre al runbook.
   b. Desde el menú desplegable **Tipo de runbook**. Seleccione su tipo. El nombre del runbook debe empezar por una letra y puede tener letras, números, guiones bajos y guiones.
   c. Seleccione la **versión del entorno de ejecución**.
   d. Escriba la **descripción** correspondiente.
6. Haga clic en **Crear** para crear el runbook.

Creación de un runbook con PowerShell

Use el cmdlet New-AzAutomationRunbook para crear un runbook vacío. Use el parámetro `Type` para especificar uno de los tipos de runbook definidos para `New-AzAutomationRunbook`.

En el ejemplo siguiente se muestra cómo crear un nuevo runbook vacío.

Azure PowerShell

```
$params = @{
    AutomationAccountName = 'MyAutomationAccount'
    Name                  = 'NewRunbook'
    ResourceGroupName     = 'MyResourceGroup'
    Type                  = 'PowerShell'
}
New-AzAutomationRunbook @params
```

Importación de un runbook

Puede importar un script de PowerShell o de flujo de trabajo de PowerShell (**.ps1**), un runbook gráfico (**.graphrunbook**) o un script de Python 2 o Python 3 (**.py**) para crear su propio runbook. Especifique el tipo de runbook que se creará durante la importación teniendo en cuenta las consideraciones siguientes.

- Puede importar un archivo **.ps1** que no contenga un flujo de trabajo, ya sea en un runbook de PowerShell o en un runbook de flujo de trabajo de PowerShell. Si lo importa en un runbook de flujo de trabajo de PowerShell, se convertirá en un flujo de trabajo. En este caso, se incluyen comentarios en el runbook para describir los cambios realizados.
- Solo puede importar un archivo **.ps1** que contenga un PowerShell Workflow en un runbook de PowerShell Workflow. Si el archivo contiene varios flujos de trabajo de PowerShell, se produce un error en la importación. Debe guardar cada flujo de trabajo en su propio archivo e importar cada uno por separado.
- No debe importar un archivo **.ps1** que contiene un flujo de trabajo de PowerShell en un runbook de PowerShell, ya que el motor de scripts de PowerShell no lo reconoce.
- Importe solo un archivo **.graphrunbook** en un runbook gráfico nuevo.

Importación de un runbook desde Azure Portal

Puede usar el siguiente procedimiento para importar un archivo de script en Azure Automation.

> ⓘ **Nota:**
>
> Solo puede importar un archivo **.ps1** en un runbook de flujo de trabajo de PowerShell mediante el portal.

1. En el portal de Azure, busque y seleccione **Cuentas de automatización**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. Desde la cuenta de Automation, seleccione **Runbooks** en **Automatización de procesos** para abrir la lista de runbooks.
4. Haga clic en **Importar un runbook**. Puede seleccionar cualquiera de las siguientes opciones:
   a. **Buscar un archivo**: esta opción selecciona un archivo de la máquina local.
   b. **Examinar desde la galería**: puede examinar y seleccionar un runbook existente de la galería.
5. Seleccione el archivo.
6. Si el campo **Nombre** está habilitado, tiene la opción de cambiar el nombre del runbook. El nombre debe empezar por una letra y puede tener letras, números, guiones bajos y guiones.
7. El **tipo de libro de procedimientos** se completa automáticamente, pero puede cambiar el tipo después de tener en cuenta las restricciones aplicables.
8. La **versión del entorno de ejecución** se rellena automáticamente o puede elegir la versión de la lista desplegable.
9. Haga clic en **Import**. El nuevo runbook aparece en la lista de runbooks de la cuenta de Automation.
10. Tiene que publicar el runbook antes de poder ejecutarlo.

> ⓘ **Nota:**
>
> Después de importar un runbook gráfico, puede convertirlo en otro tipo, pero no puede convertir un runbook gráfico en un runbook de texto.

Importar un runbook con PowerShell

Use el cmdlet Import-AzAutomationRunbook para importar un archivo de script como un runbook de borrador. Si ya existe el runbook, se producirá un error de importación, a menos que utilice el parámetro `Force` con el cmdlet.

En los ejemplos siguientes se muestra cómo importar un archivo de script en un runbook.

Azure PowerShell

```
$params = @{
    AutomationAccountName = 'MyAutomationAccount'
    Name                  = 'Sample_TestRunbook'
    ResourceGroupName     = 'MyResourceGroup'
    Type                  = 'PowerShell'
    Path                  = 'C:\Runbooks\Sample_TestRunbook.ps1'
}
Import-AzAutomationRunbook @params
```

Gestionar recursos

Si el runbook crea un recurso, el script debe comprobar que el recurso ya exista antes de intentar crearlo. Este es un ejemplo básico.

PowerShell

```
$vmName = 'WindowsVM1'
$rgName = 'MyResourceGroup'
$myCred = Get-AutomationPSCredential 'MyCredential'

$vmExists = Get-AzResource -Name $vmName -ResourceGroupName $rgName
if (-not $vmExists) {
    Write-Output "VM $vmName does not exist, creating"
    New-AzVM -Name $vmName -ResourceGroupName $rgName -Credential $myCred
} else {
    Write-Output "VM $vmName already exists, skipping"
}
```

Recuperación de detalles del registro de actividad

Puede recuperar detalles del runbook, como la persona o la cuenta que ha iniciado un runbook, del registro de actividad para la cuenta de Automation. En el siguiente ejemplo de PowerShell se proporciona el último usuario que ha ejecutado el runbook especificado.

PowerShell

```
$rgName = 'MyResourceGroup'
$accountName = 'MyAutomationAccount'
$runbookName = 'MyRunbook'
$startTime = (Get-Date).AddDays(-1)

$params = @{
    ResourceGroupName = $rgName
    StartTime         = $startTime
}
$JobActivityLogs = (Get-AzLog @params).Where( { $_.Authorization.Action -eq
```

```
'Microsoft.Automation/automationAccounts/jobs/write' })

$JobInfo = @{}
foreach ($log in $JobActivityLogs) {
    # Get job resource
    $JobResource = Get-AzResource -ResourceId $log.ResourceId

    if ($null -eq $JobInfo[$log.SubmissionTimestamp] -and
$JobResource.Properties.Runbook.Name -eq $runbookName) {
        # Get runbook
        $jobParams = @{
            ResourceGroupName     = $rgName
            AutomationAccountName = $accountName
            Id                    = $JobResource.Properties.JobId
        }
        $Runbook = Get-AzAutomationJob @jobParams | Where-Object RunbookName -EQ
$runbookName

        # Add job information to hashtable
        $JobInfo.Add($log.SubmissionTimestamp, @($Runbook.RunbookName, $Log.Caller,
$JobResource.Properties.jobId))
    }
}
$JobInfo.GetEnumerator() | Sort-Object Key -Descending | Select-Object -First 1
```

Seguimiento del progreso

Se recomienda crear los runbooks para que tengan una naturaleza modular, con una lógica que se pueda reutilizar y reiniciar fácilmente. El seguimiento del progreso de un runbook ayuda a garantizar que la lógica del runbook se ejecute correctamente si surgen incidencias.

Puede realizar un seguimiento del progreso de un runbook mediante un origen externo, como una cuenta de almacenamiento, una base de datos o archivos compartidos. Cree lógica en el runbook para comprobar primero el estado de la última acción que se ha realizado. Luego, en función de los resultados de la comprobación, la lógica puede omitir o continuar tareas específicas del runbook.

Evitación de trabajos simultáneos

Algunos runbooks se comportan de forma extraña si se ejecutan en varios trabajos al mismo tiempo. En este caso, es importante que un runbook implemente la lógica para establecer si ya hay un trabajo en ejecución. Este es un ejemplo básico.

PowerShell

```
# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process
```

```
# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext

# Check for already running or new runbooks
$runbookName = "runbookName"
$resourceGroupName = "resourceGroupName"
$automationAccountName = "automationAccountName"

$jobs = Get-AzAutomationJob -ResourceGroupName $resourceGroupName -
AutomationAccountName $automationAccountName -RunbookName $runbookName -DefaultProfile
$AzureContext

# Ranking all the active jobs
$activeJobs = $jobs | where {$_.status -eq 'Running' -or $_.status -eq 'Queued' -or
$_.status -eq 'New' -or $_.status -eq 'Activating' -or $_.status -eq 'Resuming'} |
Sort-Object -Property CreationTime
$jobRanking = @()
$rank = 0
ForEach($activeJob in $activeJobs)
{
    $rank = $rank + 1
    $activeJob | Add-Member -MemberType NoteProperty -Name jobRanking -Value $rank -
Force
    $jobRanking += $activeJob
}

$AutomationJobId = $PSPrivateMetadata.JobId.Guid
$currentJob = $activeJobs | where {$_.JobId -eq $AutomationJobId}
$currentJobRank = $currentJob.jobRanking

# Only allow the Job with Rank = 1 to start processing.
If($currentJobRank -ne "1")
{
    Write-Output "$(Get-Date -Format yyyy-MM-dd-hh-mm-ss.ffff) Concurrency check
failed as Current Job Ranking is not 1 but $($currentJobRank) therefore exiting..."
    Exit
} Else
{
    Write-Output "$(Get-Date -Format yyyy-MM-dd-hh-mm-ss.ffff) Concurrency check
passed. Start processing.."
}
```

Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:

1. En la línea 5, elimine `$AzureContext = (Connect-AzAccount -Identity).context`,

2. Sustitúyalo por `$AzureContext = (Connect-AzAccount -Identity -AccountId <ClientId>).context`, y
3. Escriba el id. de cliente.

> **Nota:**
>
> En el caso de los trabajos híbridos de PowerShell 7.2, realice cambios en la línea 28. Reemplace `$PSPrivateMetadata.JobId.Guid` por `$env:PSPrivateMetaData`.

Control de errores transitorios en un script dependiente del tiempo

Los runbooks deben ser sólidos y capaces de controlar los errores, incluidos los transitorios, que pueden provocar que se reinicien o fallen. Si se produce un error en un runbook, Azure Automation vuelve a intentarlo.

Si su runbook normalmente se ejecuta dentro de un límite de tiempo, haga que el script incluya lógica para controlar el tiempo de ejecución. Esta comprobación garantiza la ejecución de operaciones, como el inicio, el apagado o el escalado horizontal, solo durante horas específicas.

> **Nota:**
>
> La hora local en el proceso de espacio aislado de Azure se establece en UTC. Al realizar cálculos de fecha y hora en sus runbooks, debe tenerse en cuenta este hecho.

Lógica de reintento en el manual de procedimientos para evitar fallos transitorios

Los runbooks suelen realizar llamadas a sistemas remotos, como Azure a través de ARM, Azure Resource Graph, servicios SQL y otros servicios web. Cuando el sistema al que llaman los runbooks está ocupado, no está disponible temporalmente o aplica limitación bajo carga, las llamadas pueden producir errores en tiempo de ejecución. Para crear resistencia en los runbooks, debe implementar la lógica de reintento al realizar las llamadas para que los runbooks puedan controlar un problema transitorio sin errores.

Para obtener más información, consulte patrón de reintento y REST general y directrices de reintento.

Ejemplo 1: Si el runbook realiza solo una o dos llamadas

PowerShell

```
$searchServiceURL = "https://$searchServiceName.search.windows.net"
$resource = Get-AzureRmResource -ResourceType "Microsoft.Search/searchServices" -
ResourceGroupName $searchResourceGroupName -ResourceName  $searchServiceName -
ApiVersion 2015-08-19
$searchAPIKey = (Invoke-AzureRmResourceAction -Action listAdminKeys -ResourceId
$resource.ResourceId -ApiVersion 2015-08-19 -Force).PrimaryKey
```

Al llamar a `Invoke-AzureRmResourceAction`, puede observar errores transitorios. En este escenario, se recomienda implementar el siguiente patrón básico en torno a la llamada al cmdlet.

PowerShell

```
$searchServiceURL = "https://$searchServiceName.search.windows.net"
$resource = Get-AzureRmResource -ResourceType "Microsoft.Search/searchServices" -
ResourceGroupName $searchResourceGroupName -ResourceName  $searchServiceName -
ApiVersion 2015-08-19

    # Adding in a retry
    $Stoploop = $false
    $Retrycount = 0

    do {
        try   {
               $searchAPIKey = (Invoke-AzureRmResourceAction -Action listAdminKeys -
ResourceId $resource.ResourceId -ApiVersion 2015-08-19 -Force).PrimaryKey
               write-verbose "Invoke-AzureRmResourceAction on $resource.ResourceId
completed"
               $Stoploop = $true
              }
        catch {
               if ($Retrycount -gt 3)
                 {
                  Write-verbose "Could not Invoke-AzureRmResourceAction on
$resource.ResourceId after 3 retrys."
                  $Stoploop = $true
                 }
               else
                 {
                  Write-verbose "Could not Invoke-AzureRmResourceAction on
$resource.ResourceId retrying in 30 seconds..."
                  Start-Sleep -Seconds 30
                  $Retrycount = $Retrycount + 1
                 }
               }
        }
    While ($Stoploop -eq $false)
```

> ⓘ **Nota:**
>
> El reintento de la llamada es de hasta tres veces, suspendiendo durante 30 segundos cada vez.

Ejemplo 2: si el runbook realiza llamadas remotas frecuentes

Si el runbook realiza llamadas remotas frecuentes, podría experimentar problemas transitorios de tiempo de ejecución. Cree una función que implemente la lógica de reintento para cada llamada que se realiza y pase la llamada como un bloque de script que se va a ejecutar.

PowerShell

```
Function ResilientRemoteCall {

         param(
               $scriptblock
               )

         $Stoploop = $false
         $Retrycount = 0

         do {
             try   {
                    Invoke-Command -scriptblock $scriptblock
                    write-verbose "Invoked $scriptblock completed"
                    $Stoploop = $true
                   }
             catch {
                    if ($Retrycount -gt 3)
                      {
                       Write-verbose "Invoked $scriptblock failed 3 times and we will
not try again."
                       $Stoploop = $true
                      }
                    else
                      {
                       Write-verbose "Invoked $scriptblock failed  retrying in 30
seconds..."
                       Start-Sleep -Seconds 30
                       $Retrycount = $Retrycount + 1
                      }
                    }
             }
         While ($Stoploop -eq $false)
}
```

A continuación, puede pasar cada llamada remota a la función como

`ResilientRemoteCall { Get-AzVm }` o

`ResilientRemoteCall { $searchAPIKey = (Invoke-AzureRmResourceAction -Action listAdminKeys -ResourceId $resource.ResourceId -ApiVersion 2015-08-19 -Force).PrimaryKey}`

Trabajo con varias suscripciones

El runbook debe poder trabajar con suscripciones. Por ejemplo, para controlar varias suscripciones, el runbook debe usar el cmdlet Disable-AzContextAutosave. Este cmdlet garantiza que no se recupere el contexto de autenticación desde otro runbook que se ejecute en el mismo espacio aislado.

PowerShell

```
# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process

# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription `
    -DefaultProfile $AzureContext

$childRunbookName = 'childRunbookDemo'
$resourceGroupName = "resourceGroupName"
$automationAccountName = "automationAccountName"

$startParams = @{
    ResourceGroupName     = $resourceGroupName
    AutomationAccountName = $automationAccountName
    Name                  = $childRunbookName
    DefaultProfile        = $AzureContext
}
Start-AzAutomationRunbook @startParams
```

Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:

1. En la línea 5, quite `$AzureContext = (Connect-AzAccount -Identity).context`,
2. Sustitúyalo por `$AzureContext = (Connect-AzAccount -Identity -AccountId <ClientId>).context`, y
3. Escriba el id. de cliente.

Trabajo con un script personalizado

> ⓘ **Nota:**
>
> Normalmente no se pueden ejecutar scripts y runbooks personalizados en el host con un agente de Log Analytics instalado.

Para utilizar un script personalizado, haga lo siguiente:

1. Crear una cuenta de Automation.
2. Implemente el rol Hybrid Runbook Worker.

Prueba de un runbook

Al probar un runbook, se ejecuta la versión de borrador y se completará cualquier acción que este realice. No se crea ningún historial de trabajos, pero los flujos salida y advertencia y error se muestran en el panel de **salida de la prueba**. Los mensajes del flujo detallado solo se muestran en el panel de salida si la variable VerbosePreference está establecida en `Continue`.

Aunque se esté ejecutando la versión de borrador, el runbook sigue ejecutándose con normalidad y realiza cualquier acción sobre los recursos del entorno. Por este motivo, solo debe probar runbooks en recursos no pertenecientes a entornos de producción.

> ⓘ **Nota:**
>
> Todas las acciones de ejecución de runbooks se registran en el **registro de actividad** de la cuenta de Automation con el nombre de la operación **Crear un trabajo de Azure Automation**. Sin embargo, la ejecución de runbooks en un panel de prueba donde se ejecuta la versión de borrador del runbook se registraría en los registros de actividad con el nombre de la operación **Escribir un borrador de runbook de Azure Automation**. Seleccione las pestañas **Operación** y **JSON** para ver el ámbito que termina en .. */runbooks/(nombre de runbook)/draft/testjob*.

El procedimiento para probar cada tipo de runbook es el mismo. No hay diferencias entre realizar las pruebas en el editor de texto o en el editor gráfico de Azure Portal.

1. Abra la versión de borrador del runbook en el editor de texto o el editor gráfico.
2. Haga clic en **Prueba** para abrir la página de **prueba**.

3. Si el runbook tiene parámetros, se enumeran en el panel izquierdo, donde puede proporcionar los valores que se usarán para la prueba.
4. Si quiere ejecutar la prueba en una instancia de Hybrid Runbook Worker, cambie los **Parámetros de ejecución** a **Hybrid Worker** y seleccione el nombre del grupo de destino. De lo contrario, mantenga el valor predeterminado **Azure** para ejecutar la prueba en la nube.
5. Haga clic en **Iniciar** para iniciar la prueba.
6. Puede usar los botones en el panel de **salida** para detener o suspender un runbook de flujo de trabajo de PowerShell o gráfico mientras se está probando. Cuando se suspende el runbook, este completa la actividad que está realizando en ese momento antes de suspenderse. Una vez suspendido, puede detener o reiniciar el runbook.
7. Inspeccione la salida del runbook en el panel **Output**.

Publicar un manual de procedimientos

Cuando crea o importa un nuevo runbook, debe publicarlo antes de poder ejecutarlo. Cada runbook de Azure Automation tiene una versión de borrador y una versión publicada. Solo es posible ejecutar la versión publicada y solo es posible editar la versión de borrador. Los cambios realizados en la versión de borrador no afectan la versión publicada. Cuando la versión en borrador deba ponerse a disposición, la publica, sobrescribiendo la versión publicada actual con la versión en borrador.

Publicar un runbook en el portal de Azure

1. En el portal de Azure, busque y seleccione **Cuentas de automatización**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. Abra el runbook en la cuenta de Automation.
4. Haga clic en **Editar**.
5. Haga clic en el botón **Publicar** y, a continuación, seleccione **Sí** en el mensaje de comprobación.

Publicación de un runbook mediante PowerShell

Utilice el cmdlet Publish-AzAutomationRunbook para publicar su runbook.

Azure PowerShell

```
$accountName = "MyAutomationAccount"
$runbookName = "Sample_TestRunbook"
$rgName = "MyResourceGroup"
```

```
$publishParams = @{
    AutomationAccountName = $accountName
    ResourceGroupName     = $rgName
    Name                  = $runbookName
}
Publish-AzAutomationRunbook @publishParams
```

Programación de un runbook en Azure Portal

Una vez publicado el runbook, puede programarlo para que funcione:

1. En el portal de Azure, busque y seleccione **Cuentas de automatización**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. Seleccione el runbook de la lista de runbooks.
4. Seleccione **Programas** en **Recursos**.
5. Seleccione **Agregar una programación**.
6. En el panel **Programar runbook**, seleccione **Vincular una programación al runbook**.
7. En el panel **Programación**, seleccione **Crear una nueva programación**.
8. Escriba un nombre, una descripción y otros parámetros en el panel **Nueva programación**.
9. Una vez creada la programación, resáltela y haga clic en **Aceptar**. Ahora debería estar vinculada a su runbook.
10. Busque un correo electrónico en el buzón que le informa sobre el estado del runbook.

Restauración del runbook eliminado

Puede recuperar un runbook eliminado mediante scripts de PowerShell. Para recuperar un runbook, asegúrese de que se cumplan las condiciones siguientes:

- Los runbooks que se van a restaurar se eliminaron en los últimos 29 días.
- La cuenta de Automation para ese runbook existe.
- El permiso de rol *colaborador de Automation* se concede a la identidad administrada asignada por el sistema de la cuenta de Automation.

Script de PowerShell

- Ejecute el script de PowerShell como un trabajo en la cuenta de Automation para restaurar los runbooks eliminados.
- Descargue el script de PowerShell desde GitHub. Como alternativa, puede importar el script de PowerShell llamado *Restore Automation runbook* desde la Galería de runbooks.

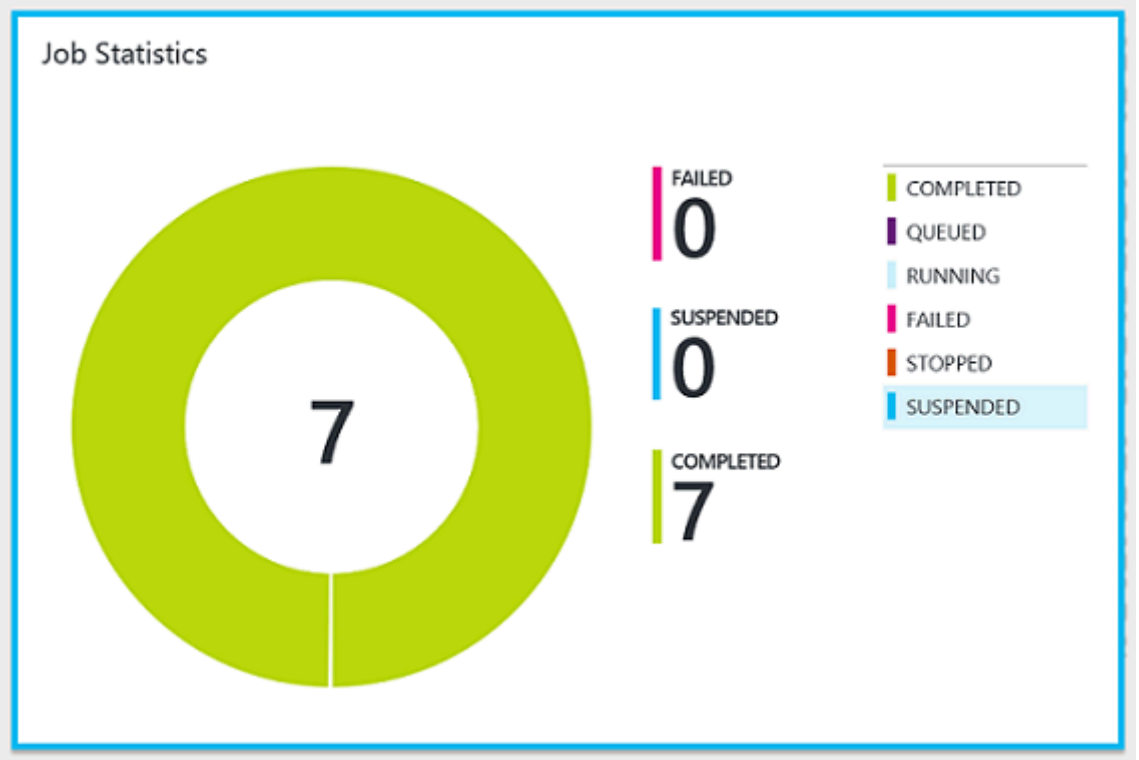
Proporcione el nombre del runbook que se va a restaurar y ejecútelo como un trabajo en Azure Automation para restaurar los runbooks eliminados.

- Descargue el script desde GitHub o importe el script de PowerShell llamado *List Deleted Automation Runbook* desde la Galería de runbooks para identificar los nombres de los runbooks eliminados en los últimos 29 días.

Obtención de estados del trabajo

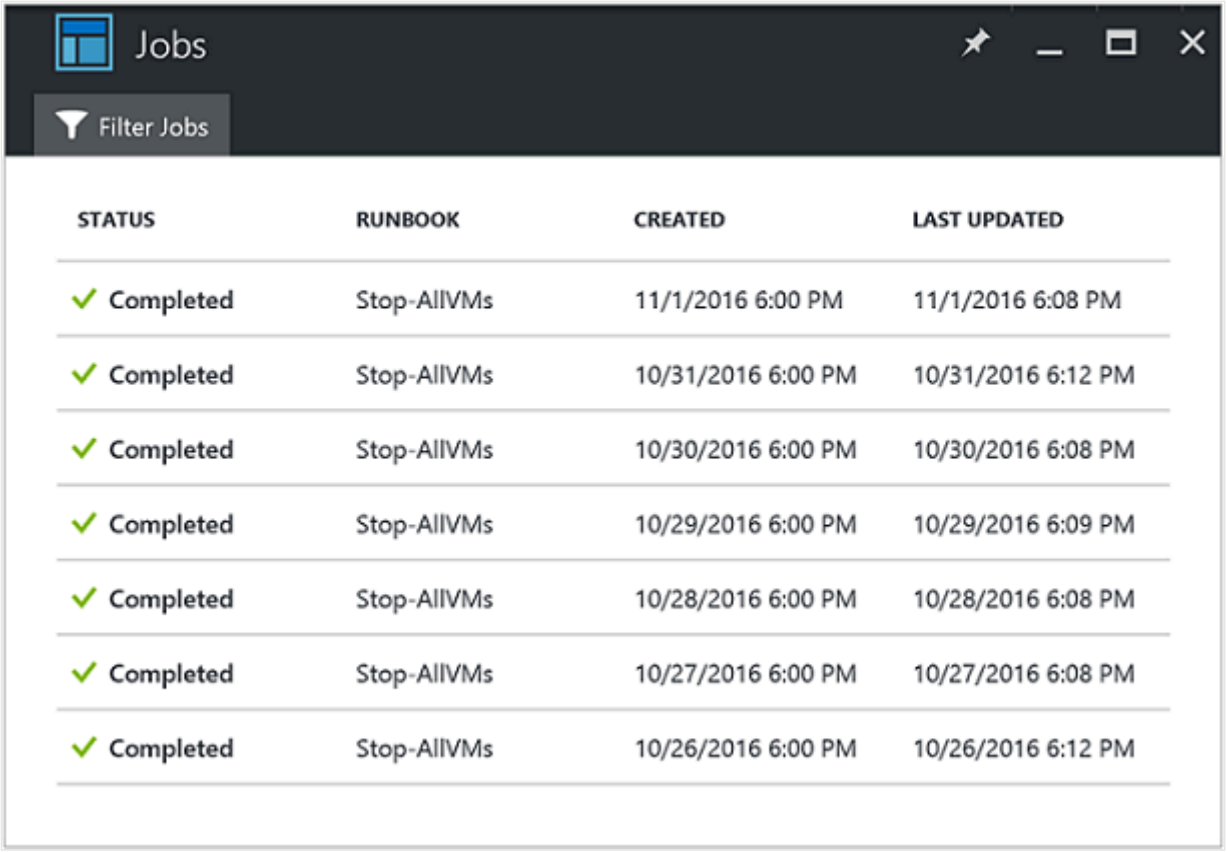
Visualización de los estados en Azure Portal

En Azure Automation, los detalles del control de trabajos se proporcionan en Trabajos. Cuando esté listo para ver los trabajos de runbook, use Azure Portal y acceda a su cuenta de Automation. A la derecha, se puede ver un resumen de todos los trabajos de runbook en **Estadísticas de trabajo**.

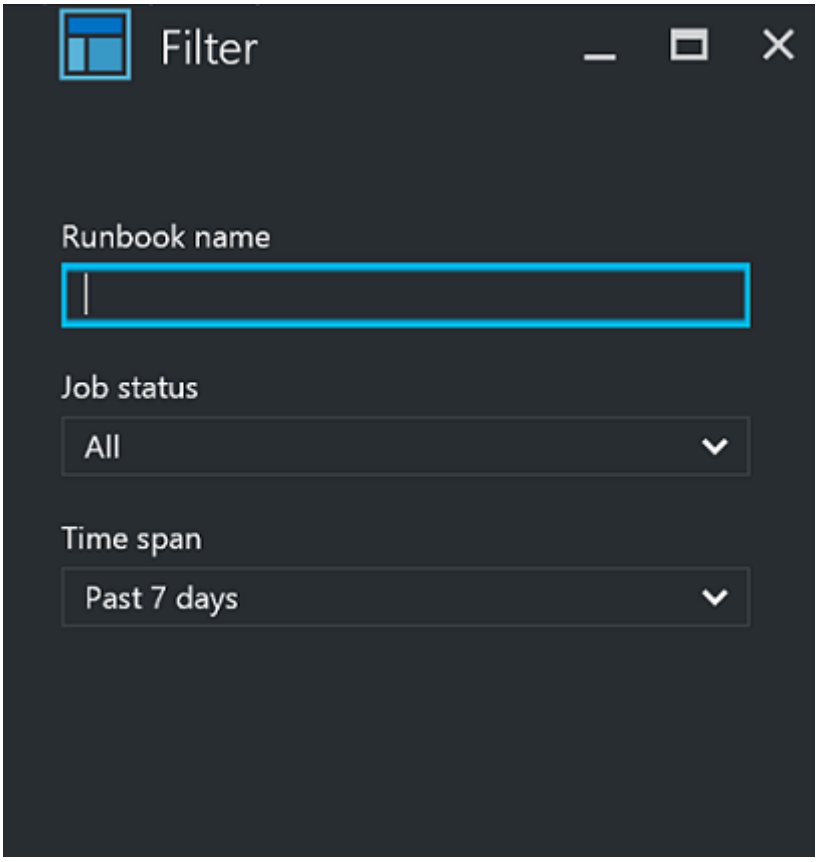


Este resumen muestra un contador y una representación gráfica del estado de trabajo de cada trabajo ejecutado.

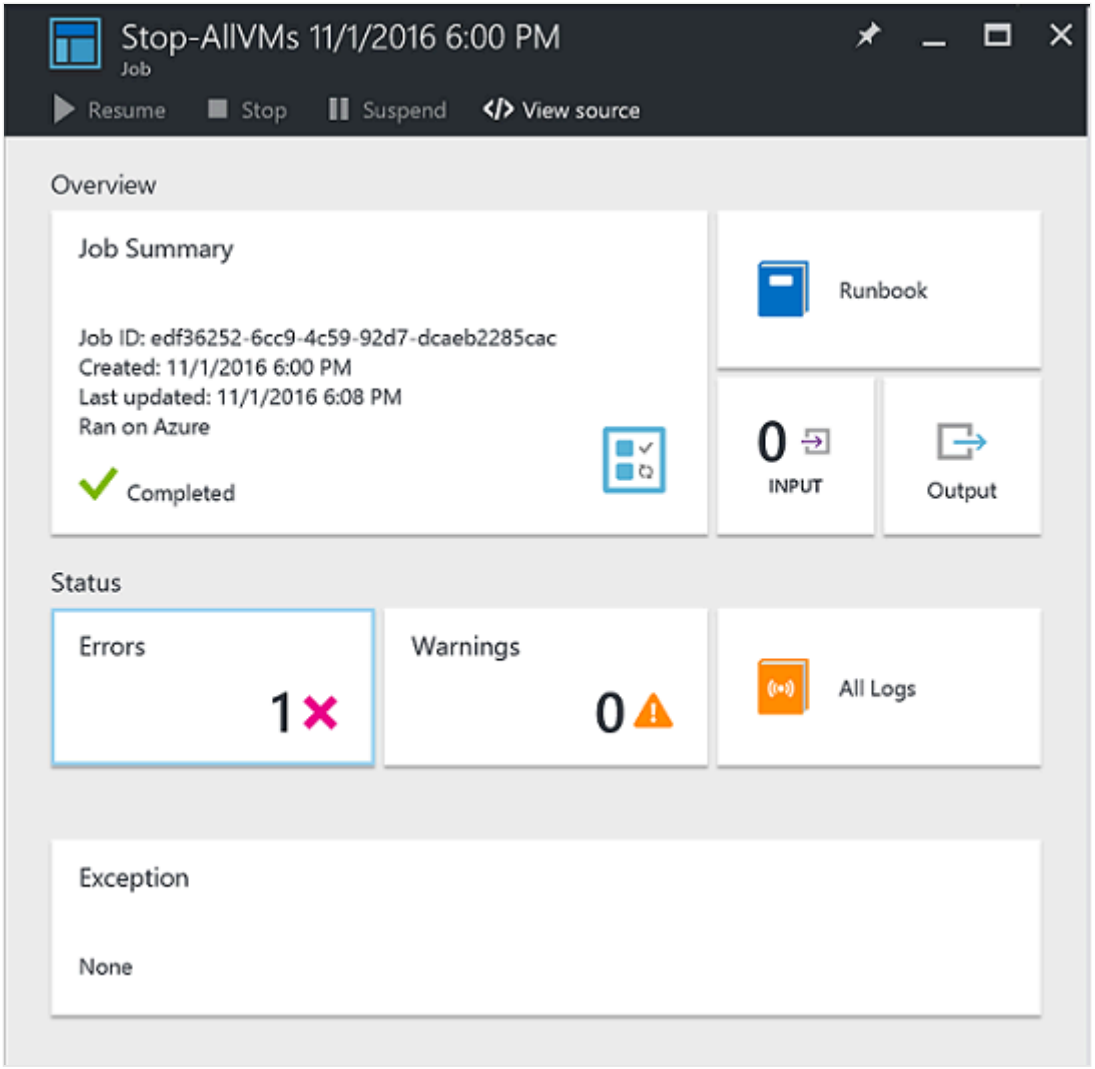
Al hacer clic en el icono aparece la página Trabajos, que incluye una lista resumida de todos los trabajos ejecutados. En esta página se muestra el estado, el nombre del runbook, la hora de inicio y la hora de finalización de cada trabajo.



Para filtrar la lista de trabajos, puede seleccionar **Filtrar trabajos**. Filtre por un runbook específico, por el estado del trabajo o por una opción de la lista desplegable, y proporcione el intervalo de tiempo para la búsqueda.



Como alternativa, puede ver los detalles resumidos de los trabajos de un runbook concreto; para ello, selecciónelo en la página Runbooks de la cuenta de Automation y, después, seleccione **Trabajos**. Esta acción muestra la página Trabajos. Desde aquí, puede hacer clic en un registro del trabajo para ver sus detalles y resultados.



Recuperación de los estados del trabajo con PowerShell

Puede utilizar el cmdlet Get-AzAutomationJob para recuperar los trabajos creados para un runbook y los detalles de un trabajo determinado. Si inicia un runbook mediante `Start-AzAutomationRunbook`, se devuelve el trabajo resultante. Utilice Get-AzAutomationJobOutput para recuperar la salida de un trabajo.

En el ejemplo siguiente se recupera el último trabajo para un runbook de ejemplo y se muestra su estado, los valores proporcionados para los parámetros del runbook y la salida del trabajo.

Azure PowerShell

```
$getJobParams = @{
    AutomationAccountName = 'MyAutomationAccount'
    ResourceGroupName     = 'MyResourceGroup'
```

```
    Runbookname           = 'Test-Runbook'
}
$job = (Get-AzAutomationJob @getJobParams | Sort-Object LastModifiedDate -Desc)[0]
$job | Select-Object JobId, Status, JobParameters

$getOutputParams = @{
    AutomationAccountName = 'MyAutomationAccount'
    ResourceGroupName     = 'MyResourceGroup'
    Id                    = $job.JobId
    Stream                = 'Output'
}
Get-AzAutomationJobOutput @getOutputParams
```

En el ejemplo siguiente se recupera la salida de un trabajo específico y se devuelve cada registro. Si hay una excepción para uno de los registros, el script escribe la excepción en lugar del valor. Este comportamiento resulta útil, ya que las excepciones pueden proporcionar información adicional que podría no registrarse normalmente durante la salida.

Azure PowerShell

```
$params = @{
    AutomationAccountName = 'MyAutomationAccount'
    ResourceGroupName     = 'MyResourceGroup'
    Stream                = 'Any'
}
$output = Get-AzAutomationJobOutput @params

foreach ($item in $output) {
    $jobOutParams = @{
        AutomationAccountName = 'MyAutomationAccount'
        ResourceGroupName     = 'MyResourceGroup'
        Id                    = $item.StreamRecordId
    }
    $fullRecord = Get-AzAutomationJobOutputRecord @jobOutParams

    if ($fullRecord.Type -eq 'Error') {
        $fullRecord.Value.Exception
    } else {
        $fullRecord.Value
    }
}
```

Pasos siguientes

- Para ver consultas de ejemplo, consulte Consultas de ejemplo para registros de trabajos y flujos de trabajo
- Para obtener información detallada sobre la administración de un runbook, vea Ejecución de un runbook en Azure Automation.

- Para obtener información sobre la compatibilidad con las versiones de tiempo de ejecución y su retirada al administrar runbooks, consulte Política de compatibilidad y retirada de entornos de ejecución de lenguajes para Azure Automation.
- Para preparar un runbook de PowerShell, vea Edición de runbooks de texto en Azure Automation.
- Para solucionar problemas con la ejecución de un runbook, consulte Solucionar problemas del runbook.

Last updated on 29/06/2026

Editar runbooks de texto en Azure Automation

Puede utilizar el editor de texto de Azure Automation para editar runbooks de PowerShell y runbooks del flujo de trabajo de PowerShell. Este editor tiene las características típicas de otros editores de código, como IntelliSense. También utiliza codificación de colores con características especiales adicionales que le ayudarán a acceder a los recursos comunes para los runbooks.

El editor de texto incluye una característica para insertar código para cmdlets, los recursos y los runbooks secundarios en un runbook. En lugar de escribir en el código personalmente, puede seleccionar entre una lista de recursos disponibles y el editor inserta el código adecuado en el runbook.

Cada runbook de Azure Automation tiene dos versiones: una de borrador y otra publicada. Edite la versión de borrador del runbook y después publíquela para que pueda ejecutarse. No se puede editar la versión publicada. Para obtener más información, vea Publicar un runbook.

En este artículo, se proporcionan pasos detallados para realizar diferentes funciones con este editor. No son aplicables a los runbooks gráficos. Para trabajar con estos runbooks, consulte Creación gráfica en Azure Automation.

> **ⓘ Importante**
>
> No incluya la palabra clave "AzureRm" en ningún script diseñado para ejecutarse con el módulo Az. La inclusión de la palabra clave, incluso en un comentario, puede hacer que AzureRm se cargue y, a continuación, entre en conflicto con el módulo Az.

Edición de un runbook con Azure Portal

1. En Azure Portal, seleccione su cuenta de Automation.
2. En **AUTOMATIZACIÓN DE PROCESOS**, seleccione **Runbooks** para abrir la lista de runbooks.
3. Elija el runbook que desea editar y, a continuación, haga clic en **Editar**.
4. Edite el runbook.
5. Haga clic en **Guardar** cuando termine las modificaciones.
6. Haga clic en **Publicar** si desea publicar la última versión de borrador del runbook.

Inserción de un cmdlet en un runbook

1. En el lienzo del editor de texto, coloque el cursor donde desee colocar el cmdlet.
2. Expanda el nodo **Cmdlets** en el control Biblioteca.
3. Expanda el módulo que contiene el cmdlet que desea usar.
4. Haga clic con el botón derecho en el nombre del cmdlet para insertarlo y seleccione **Agregar al lienzo**. Si el cmdlet tiene más de un conjunto de parámetros, el editor agrega el conjunto predeterminado. También puede expandir el cmdlet para seleccionar un conjunto de parámetros diferente.
5. Observe que el código del cmdlet se inserta con toda su lista de parámetros.
6. Proporcione un valor adecuado en lugar del valor entre corchetes (<>) para los parámetros necesarios. Quite todos los parámetros que no sean necesarios.

Inserción del código de un runbook secundario en un runbook

1. En el lienzo del editor de texto, coloque el cursor donde desee colocar el código del runbook secundario.
2. Expanda el nodo **Runbooks** en el control Biblioteca.
3. Haga clic con el botón derecho en el runbook para insertarlo y seleccione **Agregar al lienzo**.
4. Se inserta el código para el runbook secundario con los marcadores de posición de los parámetros de runbook.
5. Reemplace los marcadores de posición por valores adecuados para cada parámetro.

Inserción de un recurso en un runbook

1. En el control Lienzo del editor de texto, coloque el cursor donde desee colocar el código del runbook secundario.
2. Expanda el nodo **Recursos** en el control Biblioteca.
3. Expanda el nodo del tipo de recurso deseado.
4. Haga clic con el botón derecho en el nombre del recurso para insertarlo y seleccione **Agregar al lienzo**. Para recursos variables, seleccione **Agregar "Obtener variable" a lienzo** o **Agregar "Establecer variable" a lienzo** en función de si desea obtener o establecer la variable.
5. Observe que el código del recurso se inserta en el runbook.

Edición de un runbook de Azure Automation mediante Windows PowerShell

Para editar un runbook con Windows PowerShell, use el editor de su elección y guarde el runbook en un archivo **.ps1**. Puede usar el cmdlet Export-AzAutomationRunbook para recuperar el contenido del runbook. Puede usar el cmdlet Import-AzAutomationRunbook para reemplazar el runbook de borrador existente por el modificado.

Recuperación del contenido de un runbook mediante Windows PowerShell

En los siguientes comandos de ejemplo, se muestra cómo se recupera el script para un runbook y se guarda en un archivo de script. En este ejemplo, se recupera la versión de borrador. También es posible recuperar la versión publicada del runbook, aunque esta no se puede cambiar.

PowerShell

```
$resourceGroupName = "MyResourceGroup"
$automationAccountName = "MyAutomatonAccount"
$runbookName = "Hello-World"
$scriptFolder = "c:\runbooks"

Export-AzAutomationRunbook -Name $runbookName -AutomationAccountName
$automationAccountName -ResourceGroupName $resourceGroupName -OutputFolder
$scriptFolder -Slot Draft
```

Modificación del contenido de un runbook mediante Windows PowerShell

En los siguientes comandos de ejemplo, se muestra cómo reemplazar el contenido existente de un runbook por el de un archivo de script.

PowerShell

```
$resourceGroupName = "MyResourceGroup"
$automationAccountName = "MyAutomatonAccount"
$runbookName = "Hello-World"
$scriptFolder = "c:\runbooks"

Import-AzAutomationRunbook -Path "$scriptfolder\Hello-World.ps1" -Name $runbookName
-Type PowerShell -AutomationAccountName $automationAccountName -ResourceGroupName
$resourceGroupName -Force
Publish-AzAutomationRunbook -Name $runbookName -AutomationAccountName
$automationAccountName -ResourceGroupName $resourceGroupName
```

Pasos siguientes

- Aprenda a administrar runbooks en Azure Automation.
- Para empezar a trabajar con runbooks de flujo de trabajo de PowerShell, vea Información sobre el flujo de trabajo de PowerShell para Azure Automation.
- Certificados.
- Conexiones.
- Credenciales.
- Programaciones.
- Variables.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.

Last updated on 17/11/2025

Control de errores en runbooks gráficos de Azure Automation

> ⓘ **Importante**
>
> Las cuentas de ejecución de Azure Automation, incluidas las cuentas de ejecución clásicas, se retiraron el **30 de septiembre de 2023** y se reemplazaron por identidades administradas. Ya no se podrán crear o renovar cuentas de ejecución a través de Azure Portal. Para obtener más información, consulta migrar de una cuenta de ejecución existente a una identidad administrada.

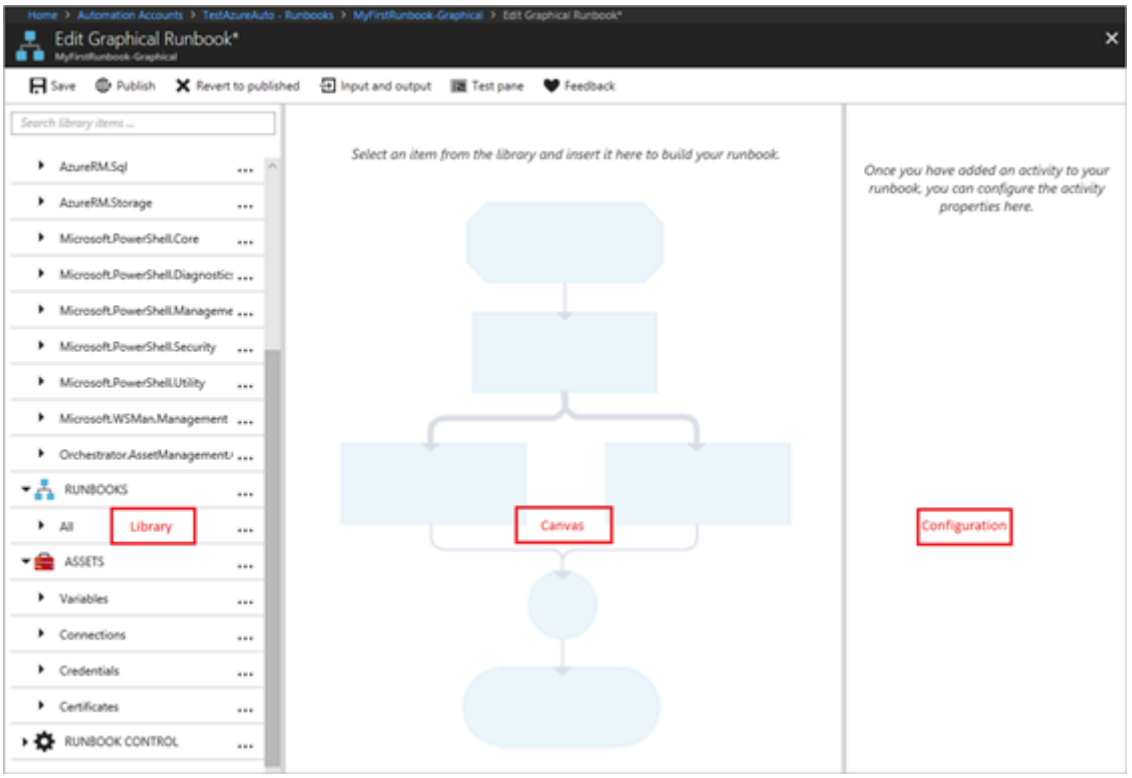
Todos los runbooks de Azure Automation son flujos de trabajo de Windows PowerShell. Los runbooks gráficos y los runbooks gráficos de flujo de trabajo de PowerShell generan código de PowerShell que se ejecuta en los trabajos de Automation, pero que no se puede ver ni modificar. Puede convertir un runbook gráfico en un runbook gráfico de flujo de trabajo de PowerShell y viceversa. Sin embargo, no puede convertir estos runbooks en un runbook textual. Además, el editor gráfico de Automation no puede importar un runbook textual.

La creación gráfica permite crear runbooks para Azure Automation sin las complejidades del código subyacente de Windows PowerShell o del flujo de trabajo de esta herramienta. Puede agregar actividades al lienzo desde una biblioteca de cmdlets y runbooks, vincularlas y configurarlas para conformar un flujo de trabajo. Si alguna vez ha utilizado System Center Orchestrator o Service Management Automation (SMA), la creación gráfica le debería resultar familiar. Este artículo proporciona una introducción a los conceptos que necesita para comenzar a crear un runbook gráfico.

> ⓘ **Nota:**
>
> No se puede agregar una firma digital a un runbook gráfico. Esta característica no se admite en Azure Automation.

Información general del editor de gráficos

Puede abrir el editor de gráficos en el Portal de Azure a través de la creación o la edición de un Runbook gráfico.



Las secciones siguientes describen los controles del editor de gráficos.

Control Lienzo

El control Lienzo permite diseñar el runbook. Puede agregar actividades desde los nodos del control Biblioteca al runbook y conectarlas con vínculos para definir la lógica del runbook. En la parte inferior del lienzo hay controles que permiten acercar y alejar.

Control de Biblioteca

El control Biblioteca permite seleccionar las actividades que se van a agregar al runbook. Se agregan al lienzo, donde puede conectarlas a otras actividades. El control Biblioteca incluye las secciones definidas en la tabla siguiente.

Expandir tabla

Sección	Descripción
Cmdlets	Todos los cmdlets que se pueden usar en el runbook. Los cmdlets se organizan por módulo. Están disponibles todos los módulos que ha instalado en la cuenta de Automation.
Runbooks	Los runbooks de la cuenta de Automation. Estos runbooks se pueden agregar al lienzo para usarse como runbooks secundarios. Solo se muestran los runbooks del mismo tipo básico que el runbook que se está editando. En el caso de los runbooks gráficos, solo se muestran runbooks basados en PowerShell. En los runbooks gráficos de flujo de trabajo de PowerShell, solo se muestran los runbooks basados en el flujo de trabajo de PowerShell.

Sección	Descripción
Activos	Los recursos de automatización de la cuenta de Automation que se pueden usar en el runbook. Cuando se agrega un recurso a un runbook, se agrega una actividad de flujo de trabajo que obtiene el recurso seleccionado. En el caso de los recursos de variables, puede seleccionar si agregar una actividad para obtener o seleccionar la variable.
Control de Runbook	Las actividades de control que se pueden usar en el runbook actual. Una actividad de unión toma varias entradas y espera hasta que todas se completan antes de continuar con el flujo de trabajo. Una actividad de código ejecuta una o varias líneas de código de PowerShell o de flujo de trabajo de PowerShell, en función del tipo de runbook gráfico. Puede usar esta actividad para ejecutar código personalizado o utilizar funcionalidades que sean difíciles de obtener con otras actividades.

Control de configuración

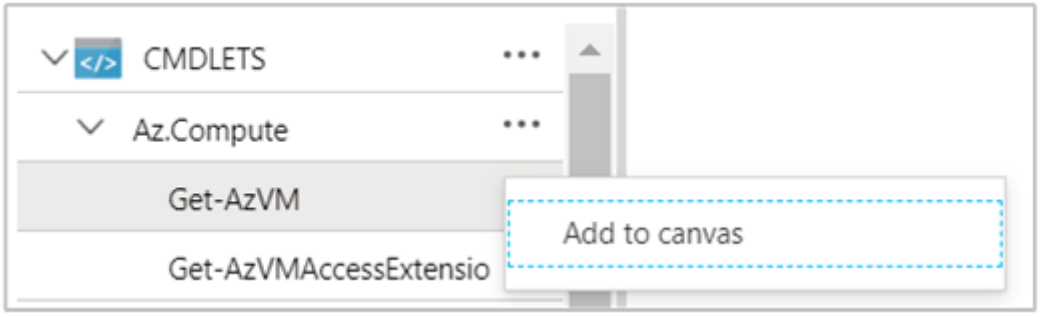
El control Configuración permite proporcionar detalles de un objeto seleccionado en el lienzo. Las propiedades disponibles en este control dependen del tipo de objeto seleccionado. Cuando selecciona una opción en el control Configuración, se abren hojas adicionales para ofrecer más información.

Control de Prueba

El control de Prueba no aparece cuando se inicia por primera vez el editor de gráficos. Se abre cuando se prueba un runbook gráfico de manera interactiva.

Uso de actividades

Las actividades son los bloques de creación de un runbook. Una actividad puede ser un cmdlet de PowerShell, un runbook secundario o un flujo de trabajo. Puede agregar una actividad al runbook; para ello, haga clic en ella con el botón derecho en el control Biblioteca y seleccione **Agregar a lienzo**. Luego puede hacer clic en la actividad y arrastrarla para colocarla en cualquier lugar del lienzo que desee. La ubicación de la actividad en el lienzo no afecta de ningún modo a la operación del runbook. Puede distribuir el runbook de la forma que considere más adecuada para visualizar su funcionamiento.

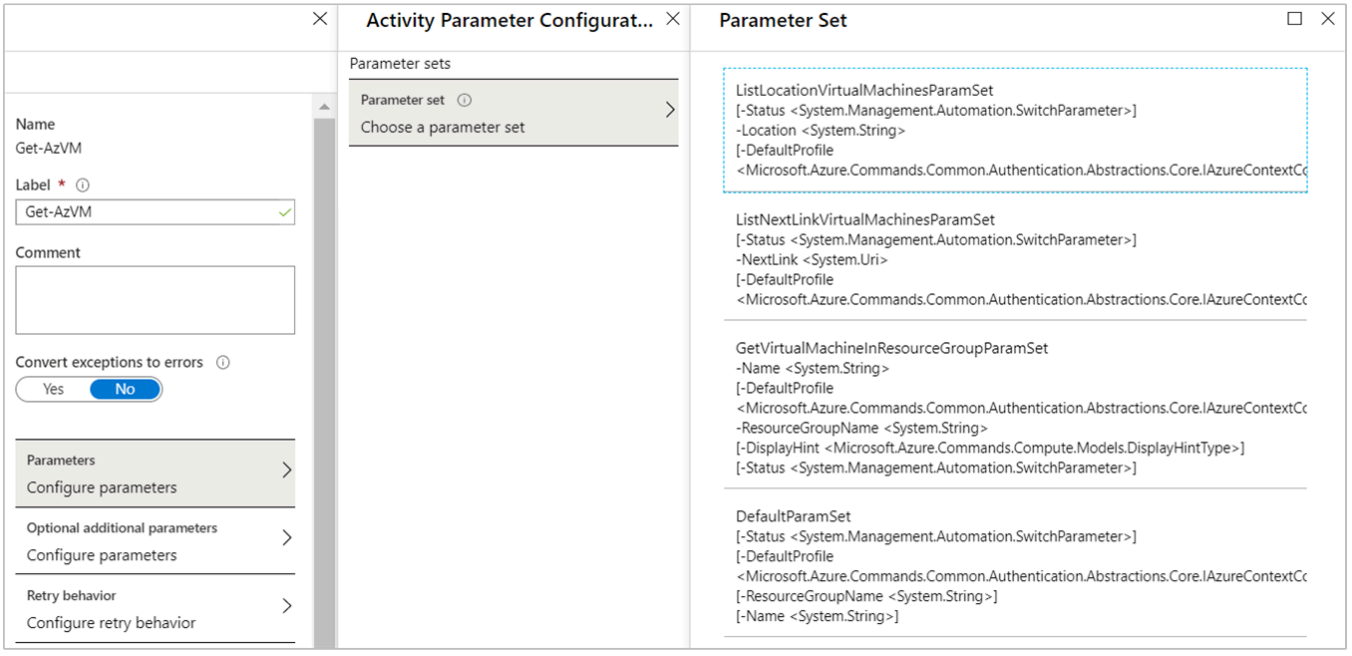


Seleccione una actividad en el lienzo para configurar sus propiedades y parámetros en la hoja Configuración. Puede cambiar la etiqueta de la actividad a un nombre que sea descriptivo. El runbook sigue ejecutando el cmdlet original. Simplemente se cambia el nombre para mostrar, que usa el editor gráfico. Tenga en cuenta que la etiqueta debe ser única dentro del runbook.

Conjuntos de parámetros

Un conjunto de parámetros define los parámetros obligatorios y opcionales que aceptan los valores de un cmdlet determinado. Todos los cmdlets tienen, al menos, un conjunto de parámetros y algunos tienen varios. Si un cmdlet tiene varios conjuntos de parámetros, debe seleccionar cuál va a usar para poder configurar los parámetros. Puede cambiar el conjunto de parámetros que una actividad usa mediante la selección de **Conjunto de parámetros** y luego de otro conjunto. En este caso, se pierden todos los valores de parámetro que ya había configurado.

En el ejemplo siguiente, el cmdlet Get-AzVM tiene tres conjuntos de parámetros. En el ejemplo se usa un conjunto denominado **ListLocationVirtualMachinesParamSet**, con un único parámetro opcional, para devolver la ubicación de las máquinas virtuales que se van a enumerar. En el ejemplo también se usa el conjunto de parámetros **GetVirtualMachineInResourceGroupParamSet** para especificar la máquina virtual que se va a devolver. Este conjunto tiene dos parámetros obligatorios y un parámetro opcional.



Valores de parámetros

Cuando se especifica un valor para un parámetro, se selecciona un origen de datos para determinar cómo se especifica el valor. Los orígenes de datos disponibles para un parámetro

concreto dependen de los valores válidos para dicho parámetro. Por ejemplo, Null no es una opción disponible para un parámetro que no permite valores nulos.

Expandir tabla

Data Source (Origen de datos)	Descripción
Valor constante	Escriba un valor para el parámetro. Este origen de datos solo se encuentra disponible para los siguientes tipos de datos: Int32, Int64, String, Boolean, DateTime, Switch.
Salida de la actividad	Utilice la salida de una actividad que precede a la actividad actual en el flujo de trabajo. Se enumeran todas las actividades válidas. Como valor del parámetro, use solo la actividad que genera el resultado. Si la actividad genera un objeto con varias propiedades, puede escribir el nombre de una propiedad específica después de seleccionar la actividad.
Entrada de Runbook	Seleccione una entrada de runbook como entrada para el parámetro de actividad.
Activo de variable	Seleccione una variable de Automation como entrada.
Activo de credencial	Seleccione una credencial de Automation como entrada.
Activo de certificado	Seleccione un certificado de Automation como entrada.
Activo de conexión	Seleccione una conexión de Automation como entrada.
PowerShell Expression	Especifique una expresión de PowerShell simple. La expresión se evalúa antes que la actividad y el resultado se usa como valor del parámetro. Puede usar variables para consultar la salida de una actividad o un parámetro de entrada de runbook.
No configurado	Borre cualquier valor configurado anteriormente.

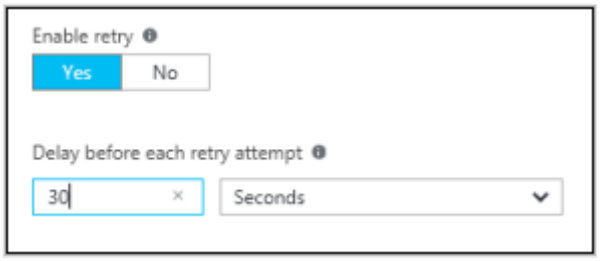
Parámetros adicionales opcionales

Todos los cmdlets tienen la opción de proporcionar parámetros adicionales. Se trata de parámetros comunes de PowerShell u otros parámetros personalizados. El editor gráfico muestra un cuadro de texto en el que puede incluir parámetros con la sintaxis de PowerShell. Por ejemplo, para usar el parámetro común `Verbose`, debe especificar `-Verbose:$True`.

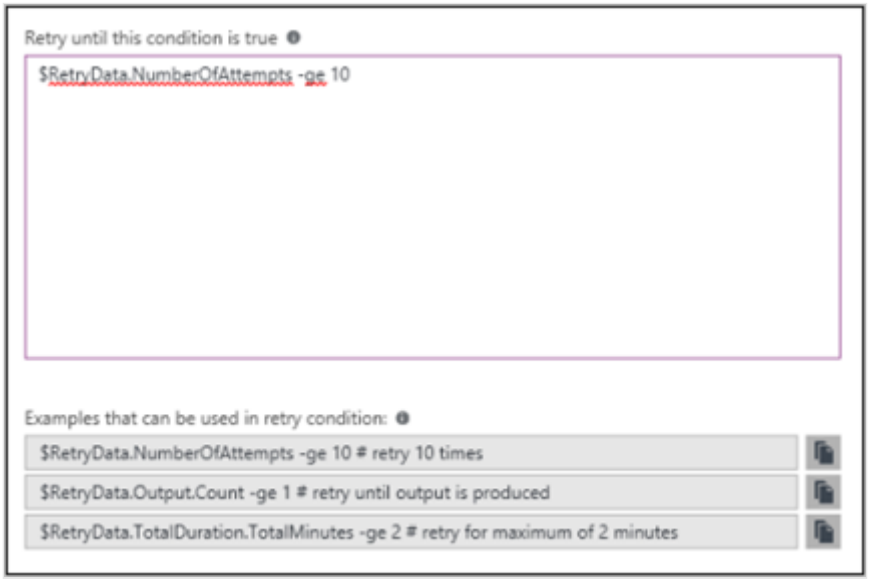
Vuelva a intentar la actividad

La funcionalidad de reintento permite que una actividad se ejecute varias veces hasta que se cumpla una condición determinada, de forma muy similar a un bucle. Esta característica se puede utilizar en las actividades que deben ejecutarse varias veces, que son propensas a errores y que pueden necesitar más de un intento para realizarse correctamente, o bien para probar que los datos de la información de salida son válidos.

Cuando se habilita el reintento de una actividad, puede establecer un retraso y una condición. El retraso es el tiempo (medido en segundos o minutos) que el runbook espera antes de volver a ejecutar la actividad. Si no se especifica un retraso, la actividad se ejecuta de nuevo inmediatamente después de completarse.



La condición de reintento es una expresión de PowerShell que se evalúa después de cada vez que se ejecuta la actividad. Si la expresión se resuelve en True, la actividad se vuelve a ejecutar. Si la expresión se resuelve en False, la actividad no se vuelve a ejecutar y el runbook pasa a la actividad siguiente.



La condición de reintento puede utilizar una variable denominada `RetryData` que proporciona acceso a información sobre los reintentos de actividad. Esta variable tiene las propiedades de la tabla siguiente:

Expandir tabla

Propiedad	Descripción
`NumberOfAttempts`	Número de veces que se ha ejecutado la actividad.
`Output`	Salida de la última ejecución de la actividad.
`TotalDuration`	Tiempo transcurrido desde la primera vez que se inició la actividad.
`StartedAt`	Hora (en formato UTC) a la que se inició la actividad por primera vez.

A continuación se muestran ejemplos de las condiciones de reintento de actividades.

PowerShell

```
# Run the activity exactly 10 times.
$RetryData.NumberOfAttempts -ge 10
```

PowerShell

```
# Run the activity repeatedly until it produces any output.
$RetryData.Output.Count -ge 1
```

PowerShell

```
# Run the activity repeatedly until 2 minutes has elapsed.
$RetryData.TotalDuration.TotalMinutes -ge 2
```

Después de configurar una condición de reintento en una actividad, esta incluirá dos indicaciones visuales que sirven de recordatorio. Una se muestra en la actividad y la otra cuando se revisa la configuración de la actividad.

Control de Script de flujo de trabajo

Un control de Script de flujo de trabajo es una actividad especial que acepta scripts de PowerShell o de flujo de trabajo de PowerShell, en función del tipo de runbook gráfico que se crea. Proporciona una funcionalidad que, de otro modo, podría no estar disponible. No puede aceptar parámetros, pero puede usar variables para los parámetros de entrada de runbook y de salida de actividad. Cualquier salida de la actividad se agrega al bus de datos. Las

excepciones se generan sin vínculo saliente, en cuyo caso la salida se agrega a la salida del runbook.

Por ejemplo, el siguiente código realiza cálculos de fecha mediante una variable de entrada del runbook denominada `NumberOfDays`. Luego envía un valor DateTime calculado como salida para su uso por parte de las actividades subsiguientes del runbook.

PowerShell

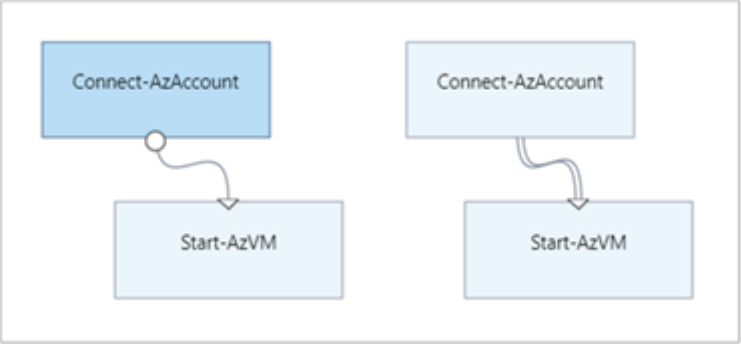
```
$DateTimeNow = (Get-Date).ToUniversalTime()
$DateTimeStart = ($DateTimeNow).AddDays(-$NumberOfDays)}
$DateTimeStart
```

Usar vínculos para el flujo de trabajo

Un vínculo en un runbook gráfico conecta dos actividades. Se muestra en el lienzo como una flecha que apunta desde la actividad de origen hasta la actividad de destino. Las actividades se ejecutan en la dirección de la flecha y la actividad de destino se inicia una vez que se completa la actividad de origen.

Crear un vínculo

Puede crear un vínculo entre dos actividades; para ello, seleccione la actividad de origen y haga clic en el círculo que aparece en la parte inferior de la forma. Arrastre la flecha a la actividad de destino y suéltela.



Seleccione el vínculo para configurar sus propiedades en la hoja Configuración. Las propiedades incluyen el tipo de vínculo, que se describe en la tabla siguiente.

Expandir tabla

Tipo de vínculo	Descripción
Canalización	La actividad de destino se ejecuta una vez para cada objeto generado desde la actividad de origen. La actividad de destino no se ejecuta si la actividad de origen no genera salida. La salida de la actividad de origen está disponible como objeto.
Secuencia	La actividad de destino se ejecuta una sola vez cuando recibe salida desde la actividad de origen. La salida de la actividad de origen está disponible como una matriz de objetos.

Actividad de iniciar runbook

Un runbook gráfico se inicia con cualquier actividad que no tenga un vínculo entrante. A menudo solo hay una actividad que actúa como actividad inicial del runbook. Si varias actividades no tienen un vínculo entrante, el runbook se inicia ejecutándolas en paralelo. Sigue los vínculos para ejecutar otras actividades a medida que finaliza cada una de ellas.

Especificar condiciones de vínculo

Cuando especifica una condición en un vínculo, la actividad de destino solo se ejecuta si la condición se resuelve en True. Normalmente usa una variable `ActivityOutput` en una condición para recuperar la salida desde la actividad de origen.

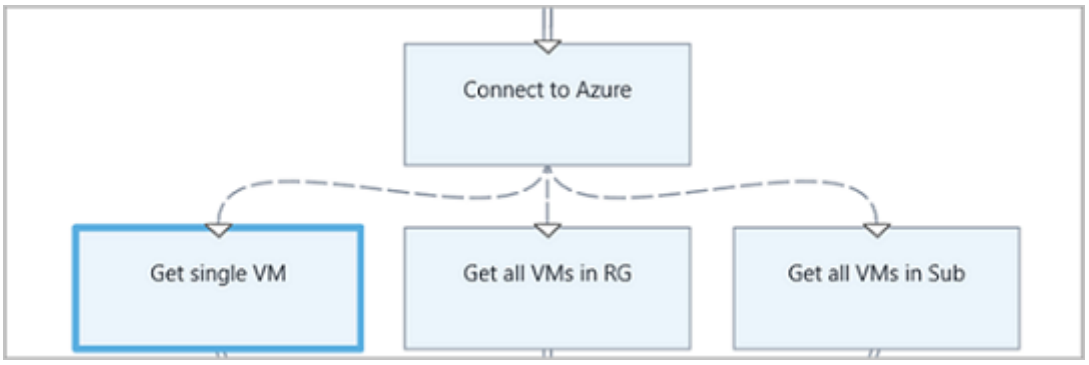
En un vínculo de canalización, debe especificar una condición para un único objeto. El runbook evalúa la condición en cada objeto generado por la actividad de origen. A continuación, ejecuta la actividad de destino en cada objeto que cumple con la condición. Por ejemplo, con una actividad de origen de `Get-AzVM`, puede usar la siguiente sintaxis para que un vínculo de canalización condicional recupere solo las máquinas virtuales del grupo de recursos denominado Group1.

PowerShell

```
$ActivityOutput['Get Azure VMs'].Name -match "Group1"
```

En el caso de un vínculo de secuencia, el runbook solo evalúa la condición una vez, debido a que se devuelve una única matriz que incluye todos los objetos de la actividad de origen. Por este motivo, el runbook no puede usar un vínculo de secuencia para el filtrado, como sucede con un vínculo de canalización. El vínculo de secuencia simplemente puede determinar si se ejecuta o no la siguiente actividad.

Por ejemplo, tomemos el siguiente conjunto de actividades de nuestro runbook **Iniciar VM**:



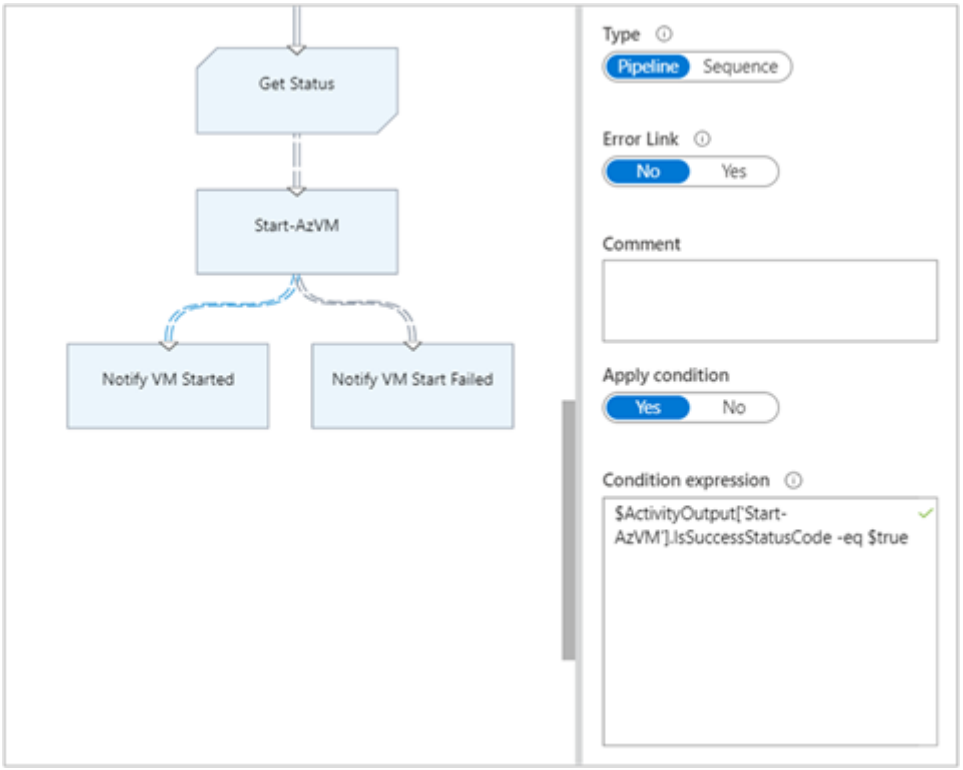
El runbook usa tres vínculos de secuencia diferentes que comprueban los valores de los parámetros de entrada `VMName` y `ResourceGroupName`, para determinar la acción adecuada que se debe llevar a cabo. Las posibles acciones consisten en iniciar una sola máquina virtual, iniciar todas las máquinas virtuales del grupo de recursos o iniciar todas las máquinas virtuales de una suscripción. En el vínculo de secuencia entre `Connect to Azure` y `Get single VM`, esta es la lógica de la condición:

PowerShell

```
<#
Both VMName and ResourceGroupName runbook input parameters have values
#>
(
(($VMName -ne $null) -and ($VMName.Length -gt 0))
) -and (
(($ResourceGroupName -ne $null) -and ($ResourceGroupName.Length -gt 0))
)
```

Cuando usa un vínculo condicional, los datos disponibles desde la actividad de origen a otras actividades de dicha rama se filtran por la condición. Si una actividad es el origen de varios vínculos, los datos disponibles para las actividades de cada rama dependen de la condición del vínculo que se conecta a dicha rama.

Por ejemplo, la actividad `Start-AzVM` del siguiente runbook inicia todas las máquinas virtuales. Además, tiene dos vínculos condicionales. El primer vínculo condicional usa la expresión `$ActivityOutput['Start-AzVM'].IsSuccessStatusCode -eq $true` para filtrar si la actividad `Start-AzVM` se completa correctamente. El segundo vínculo condicional usa la expresión `$ActivityOutput['Start-AzVM'].IsSuccessStatusCode -ne $true` para filtrar si la actividad `Start-AzVm` no puede iniciar la máquina virtual.



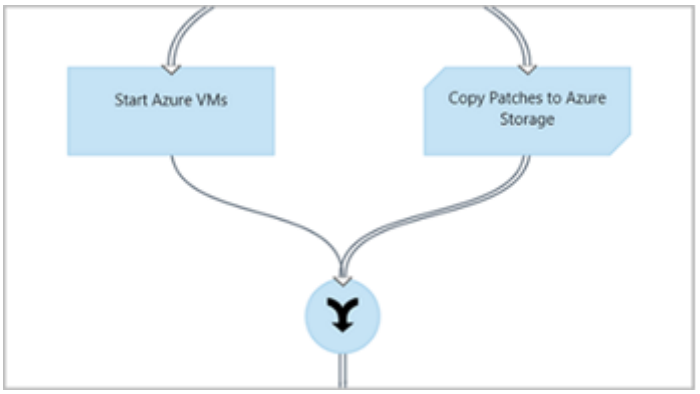
Una actividad que sigue el primer vínculo y usa la salida de la actividad de `Get-AzureVM` solo recupera las máquinas virtuales que estaban iniciadas en el momento en que se ejecutó `Get-AzureVM`. Todas las actividades que siguen el segundo vínculo solo obtienen las máquinas virtuales que estaban detenidas en el momento en que se ejecutó `Get-AzureVM`. Toda actividad que siga el tercer vínculo obtiene todas las máquinas virtuales, independientemente de cuál sea su estado de ejecución.

Usar uniones

Una unión es una actividad especial que espera hasta que se hayan completado todas las ramas entrantes. Esto permite que el runbook ejecute varias actividades en paralelo y garantiza que todas se hayan completado antes de continuar.

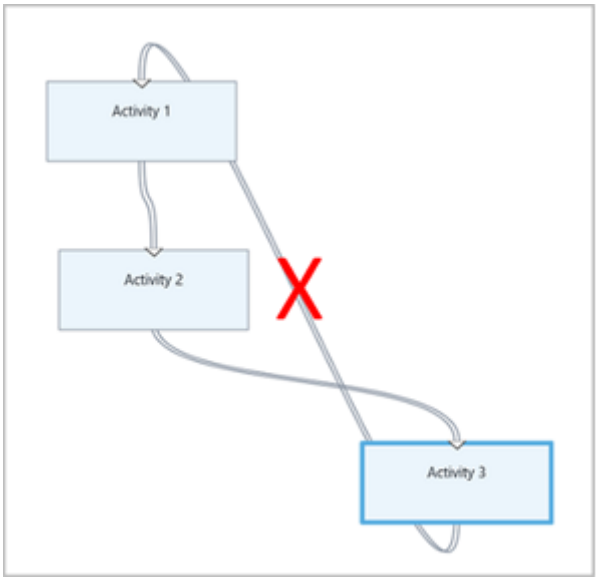
Si bien una unión puede tener una cantidad ilimitada de vínculos entrantes, solo uno de ellos puede ser una canalización. No se limita la cantidad de vínculos entrantes de secuencia. Puede crear la unión con varios vínculos entrantes de canalización y guardar el runbook, pero se producirá un error cuando se ejecute.

El ejemplo siguiente forma parte de un runbook que inicia un conjunto de máquinas virtuales, a la vez que descarga revisiones para aplicarlas a esas máquinas. Utiliza una unión para garantizar que ambos procesos se han completado antes de que el runbook continúe.



Trabajar con ciclos

Un ciclo se forma cuando una actividad de destino se vincula de vuelta a su actividad de origen o a otra actividad que, al final, se vincula a su origen. Actualmente, la creación gráfica no admite los ciclos. Si el runbook tiene un ciclo, lo guarda como corresponde, pero recibe un error cuando se ejecuta.



Uso compartido de datos entre actividades

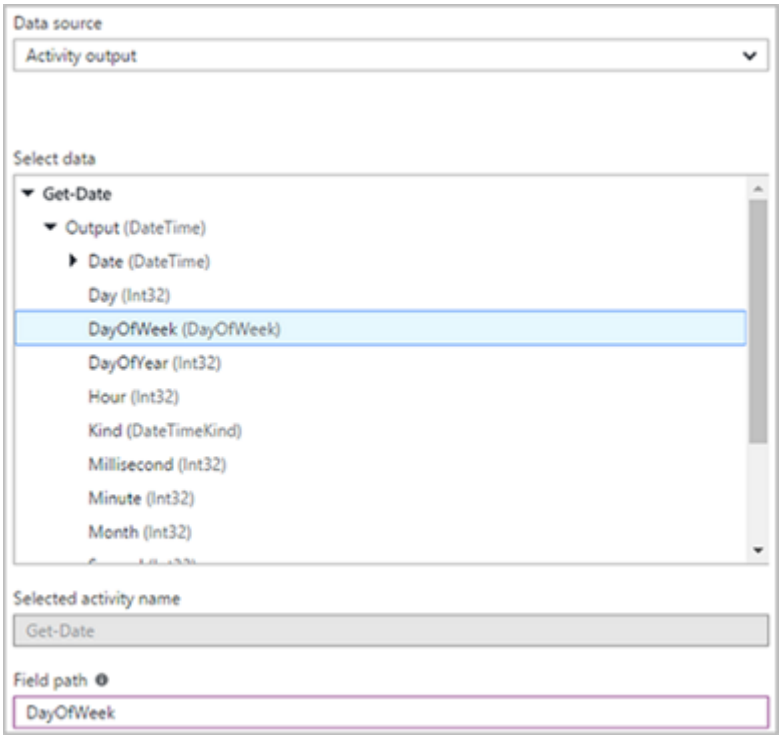
Todo dato generado por una actividad con un vínculo saliente se escribe en el bus de datos del runbook. Toda actividad del runbook puede usar datos del bus de datos para rellenar los valores de parámetros o para incluirlos en el código de script. Una actividad puede tener acceso a la salida de cualquier actividad anterior en el flujo de trabajo.

La manera en que los datos se escriben en el bus de datos depende del tipo de vínculo en la actividad. En el caso de un vínculo de canalización, los datos se generan como varios objetos. En un vínculo de secuencia, los datos se generan como una matriz. Si solo hay un valor, la salida es una matriz de un solo elemento.

El runbook tiene dos maneras de acceder a los datos del bus de datos:

- Usar un origen de datos de salida de actividad.
- Usar un origen de datos de expresión de PowerShell.

El primer método usa un origen de datos de salida de actividad para rellenar un parámetro de otra actividad. Si la salida es un objeto, el runbook puede especificar una única propiedad.



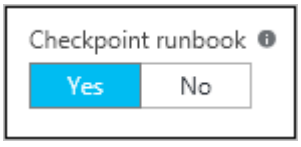
El segundo mecanismo de acceso a los datos recupera la salida de una actividad en un origen de datos de expresión de PowerShell o de una actividad de script de flujo de trabajo con una variable `ActivityOutput`, por medio de la sintaxis aquí mostrada. Si la salida es un objeto, el runbook puede especificar una única propiedad.

PowerShell

```
$ActivityOutput['Activity Label']
$ActivityOutput['Activity Label'].PropertyName
```

Usar puntos de comprobación

Puede establecer puntos de control en un runbook gráfico de flujo de trabajo de PowerShell; para ello, seleccione **Runbook de punto de control** en cualquier actividad. Esto permite establecer un punto de control después de que se ejecuta la actividad.



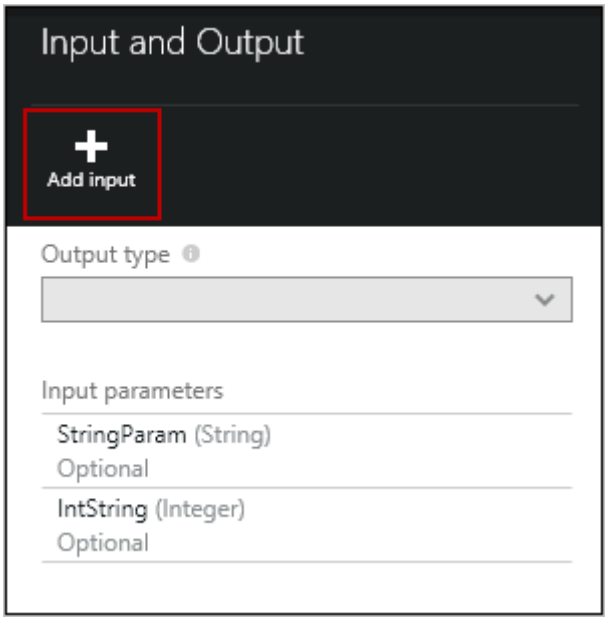
Los puntos de control solo se habilitan en los runbooks gráficos de flujo de trabajo de PowerShell; no están disponibles en los runbooks gráficos. Si el runbook usa cmdlets de Azure, debe incluir una actividad `Connect-AzAccount` después de cualquier actividad con punto de control. La operación de conexión se usa en caso de que el runbook se suspenda y deba reiniciarse a partir de este punto de control en un trabajo diferente.

Controlar entrada de runbook

Si el runbook actual se utiliza como secundario, requiere una entrada, ya sea de un usuario que inicia el runbook mediante Azure Portal o de otro runbook. Por ejemplo, si un runbook crea una máquina virtual, es posible que el usuario deba proporcionar información como el nombre de la máquina virtual y otras propiedades cada vez que se inicie el runbook.

El runbook acepta la entrada definiendo uno o más parámetros de entrada. El usuario proporciona valores para estos parámetros cada vez que se inicia el runbook. Cuando el usuario inicia el runbook mediante Azure Portal, se le solicita que proporcione valores para cada parámetro de entrada admitido por el runbook.

Al crear el runbook, puede acceder a sus parámetros de entrada haciendo clic en **Entrada y salida** en la barra de herramientas del runbook. Con esto se abre el control Entrada y salida, donde puede editar un parámetro de entrada existente o crear uno nuevo, mediante un clic en **Agregar entrada**.



Las propiedades de la siguiente tabla definen los parámetros de entrada:

Expandir tabla

Propiedad	Descripción
Nombre	Obligatorio. El nombre del parámetro. El nombre debe ser único en el runbook. Debe comenzar por una letra y solo puede contener letras, números y caracteres de subrayado. El nombre no puede contener ningún espacio.
Descripción	Opcional. Descripción del propósito del parámetro de entrada.
Tipo	Opcional. El tipo de datos que se espera para el valor del parámetro. Azure Portal proporciona un control adecuado para el tipo de datos de cada parámetro cuando se solicite una entrada. Los tipos de parámetro admitidos son String, Int32, Int64, Decimal, Boolean, DateTime y Object. Si no se selecciona ningún tipo de datos, el valor predeterminado es String.
Mandatory	Opcional. Esta propiedad especifica si se debe proporcionar un valor para el parámetro. Si elige `yes`, se debe proporcionar un valor cuando se inicia el runbook. Si elige `no`, no se necesita un valor cuando se inicia el runbook y se puede usar un valor predeterminado. No es posible iniciar el runbook si no se proporciona un valor para cada parámetro obligatorio que no tiene definido un valor predeterminado.
Valor predeterminado	Opcional. Valor utilizado para un parámetro si no se pasa uno cuando se inicia el runbook. Para establecer un valor predeterminado, elija `Custom`. Elija `None` si no desea proporcionar ningún valor predeterminado.

Controlar salida de runbook

La creación gráfica guarda los datos que crea cualquier actividad que no tiene un vínculo saliente en la salida del runbook. La salida se guarda con el trabajo del runbook y queda a disposición de un runbook principal cuando el runbook se usa como secundario.

Funcionamiento con expresiones de PowerShell

Una de las ventajas de la creación gráfica es que permite crear un runbook con un conocimiento mínimo de PowerShell. No obstante, actualmente es necesario conocer un poco PowerShell para rellenar determinados valores de parámetro y para configurar condiciones de vínculo. En esta sección se proporciona una introducción rápida a las expresiones de PowerShell. En Scripting con Windows PowerShellestán disponibles todos los detalles sobre PowerShell.

Usar una expresión de PowerShell como origen de datos

Puede usar una expresión de PowerShell como origen de datos para rellenar el valor de un parámetro de actividad con los resultados de código de PowerShell. La expresión puede ser una sola línea de código que realiza una función simple o varias líneas que desarrollan cierta

lógica compleja. La salida de un comando que no está asignado a una variable se envía al valor del parámetro.

Por ejemplo, el siguiente comando genera la fecha actual.

PowerShell

```
Get-Date
```

El fragmento de código siguiente crea una cadena a partir de la fecha actual y la asigna a una variable. El código envía el contenido de la variable a la salida.

PowerShell

```
$string = "The current date is " + (Get-Date)
$string
```

Los siguientes comandos evalúan la fecha actual y devuelven una cadena que indica si el día actual cae en fin de semana o es un día entre semana.

PowerShell

```
$date = Get-Date
if (($date.DayOfWeek = "Saturday") -or ($date.DayOfWeek = "Sunday")) { "Weekend" }
else { "Weekday" }
```

Usar salida de actividad

Para usar la salida de una actividad anterior en el runbook, use la variable `ActivityOutput` con la siguiente sintaxis.

PowerShell

```
$ActivityOutput['Activity Label'].PropertyName
```

Por ejemplo, puede tener una actividad con una propiedad que requiera el nombre de una máquina virtual. En este caso, el runbook puede usar la siguiente expresión.

PowerShell

```
$ActivityOutput['Get-AzureVM'].Name
```

Si la propiedad requiere el objeto de máquina virtual en lugar de simplemente un nombre, el runbook devuelve todo el objeto con la sintaxis siguiente.

PowerShell

```
$ActivityOutput['Get-AzureVM']
```

El runbook puede utilizar la salida de una actividad en una expresión más compleja, como la siguiente. Esta expresión concatena texto con el nombre de la máquina virtual.

PowerShell

```
"The computer name is " + $ActivityOutput['Get-AzureVM'].Name
```

Comparar valores

Utilice operadores de comparación para comparar valores o determinar si un valor coincide con un patrón especificado. Una comparación devuelve un valor de True o False.

Por ejemplo, la siguiente condición determina si la máquina virtual de una actividad denominada `Get-AzureVM` está actualmente detenida.

PowerShell

```
$ActivityOutput["Get-AzureVM"].PowerState -eq "Stopped"
```

La siguiente condición determina si la misma máquina virtual está en cualquier estado distinto de detenida.

PowerShell

```
$ActivityOutput["Get-AzureVM"].PowerState -ne "Stopped"
```

Puede combinar varias condiciones en el runbook usando un operador lógico, como `-and` u `-or`. Por ejemplo, la siguiente condición comprueba si la misma máquina virtual del ejemplo anterior se encuentra en un estado Detenido o Deteniendo.

PowerShell

```
($ActivityOutput["Get-AzureVM"].PowerState -eq "Stopped") -or ($ActivityOutput["Get-AzureVM"].PowerState -eq "Stopping")
```

Tablas hash

Las tablas hash son pares de nombre-valor útiles para devolver un conjunto de valores. También puede encontrar referencias a la tabla hash como diccionario. Las propiedades de

ciertas actividades esperan una tabla hash en lugar de un valor simple.

Una tabla hash se crea con la siguiente sintaxis. Puede contener cualquier número de entradas, pero cada una se define mediante un nombre y un valor.

PowerShell

```
@{ <name> = <value>; [<name> = <value> ] ...}
```

Por ejemplo, la siguiente expresión crea una tabla hash que se usará como origen de datos para un parámetro de actividad que espera una tabla hash de valores para una búsqueda en Internet.

PowerShell

```
$query = "Azure Automation"
$count = 10
$h = @{'q'=$query; 'lr'='lang_ja';  'count'=$Count}
$h
```

En el ejemplo siguiente se usa la salida de una actividad llamada `Get X Connection` para rellenar una tabla hash.

PowerShell

```
@{'ApiKey'=$ActivityOutput['Get X Connection'].ConsumerAPIKey;
    'ApiSecret'=$ActivityOutput['Get X Connection'].ConsumerAPISecret;
    'AccessToken'=$ActivityOutput['Get X Connection'].AccessToken;
    'AccessTokenSecret'=$ActivityOutput['Get X Connection'].AccessTokenSecret}
```

Autenticarse en los recursos de Azure

Los runbooks de Azure Automation que administran recursos de Azure tienen que autenticarse en Azure. Identidades administradas es el mecanismo predeterminado que usa un runbook de Automation para acceder a los recursos de Azure Resource Manager de la suscripción. Puede agregar esta funcionalidad a un runbook gráfico importando el siguiente runbook en la cuenta de automatización, que aprovecha la identidad administrada asignada por el sistema de la cuenta de automatización para autenticar y acceder a los recursos de Azure.

PowerShell

```
wget
https://raw.githubusercontent.com/azureautomation/runbooks/master/Utility/AzMI/Azure
AutomationTutorialWithIdentityGraphical.graphrunbook -outfile
AzureAutomationTutorialWithIdentityGraphical.graphrunbook
```

Exportar un runbook gráfico

Solo puede exportar la versión publicada de un runbook gráfico. Si el runbook todavía no se ha publicado, el botón **Exportar** está deshabilitado. Al hacer clic en el botón **Exportar**, el runbook se descarga en el equipo local. El nombre del archivo coincide con el nombre del runbook, seguido de una extensión **.graphrunbook**.

Importar un runbook gráfico

Puede importar un archivo de runbook gráfico o de runbook gráfico de flujo de trabajo de PowerShell seleccionando la opción **Importar** al agregar un runbook. Al seleccionar el archivo para importar, puede mantener el mismo nombre o indicar otro nuevo. El campo **Tipo de Runbook** muestra el tipo de runbook después de evaluar el archivo seleccionado. Si intenta seleccionar un tipo diferente que no sea correcto, el editor gráfico mostrará un mensaje para indicar que hay posibles conflictos y que puede haber errores de sintaxis durante la conversión.

Ejecutar un runbook gráfico

Cada runbook gráfico de Azure Automation tiene una versión de borrador y una versión publicada. Solo puede ejecutar la versión publicada y solo puede editar la versión de borrador. Los cambios realizados en la versión de borrador no afectan la versión publicada. Cuando la versión borrador está lista para usar, se publica, lo que sobrescribe la versión publicada actual.

Puede probar la versión de borrador de un runbook en Azure Portal, sin modificar la versión publicada. También puede probar un nuevo runbook antes de publicarlo, para comprobar que funciona correctamente antes de cualquier reemplazo de versión. La prueba de un runbook ejecuta la versión de borrador y garantiza que se completan todas las acciones que realiza. No se crea ningún historial de trabajos, pero el panel de salida de la prueba muestra la salida.

Para abrir el control Prueba del runbook gráfico, abra el runbook para editarlo y, a continuación, haga clic en el botón **Panel de prueba**. El control Prueba solicita los parámetros de entrada; puede iniciar el runbook haciendo clic en **Iniciar**.

Publicar un runbook gráfico

Para publicar un runbook gráfico, abra el runbook para editarlo y, a continuación, haga clic en **Publicar**. A continuación se indican los posibles estados del runbook:

- Nuevo: el runbook aún no se ha publicado.
- Publicado: el runbook se ha publicado.
- En edición: el runbook se ha editado después de haberlo publicado y la versión de borrador y la versión publicada son distintas.

NAME	AUTHORING STATUS	LAST MODIFIED	TAGS
Hello-World	Published	12/15/2017 4:08 PM	
MyFirstRunbook-Graphical	In edit	12/6/2017 8:02 AM	
MyFirstRunbook-PowerShell	Published	1/24/2018 12:04 PM	
MyFirstRunbook-Python	New	1/24/2018 12:32 PM	

Tiene la opción de revertir a la versión publicada de un runbook. Esta operación anula cualquier cambio realizado desde la última vez que se publicó el runbook. Reemplaza la versión de borrador del runbook con la versión publicada.

Pasos siguientes

- Para empezar a trabajar con runbooks gráficos, consulte Tutorial: Crear un runbook gráfico.
- Para más información sobre los tipos de runbook, sus ventajas y sus limitaciones, consulte Tipos de runbooks de Azure Automation.

- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.

Last updated on 17/11/2025

Creación de runbooks modulares en Automation

Un procedimiento recomendado en Azure Automation es escribir runbooks que sean reutilizables y modulares con una función independiente a la que puedan llamar otros runbooks. Con frecuencia, un runbook primario llama a uno o varios runbooks secundarios para realizar la funcionalidad necesaria.

Hay dos maneras de llamar a un runbook secundario: en modo insertado o a través de un cmdlet. En la tabla siguiente se resumen las diferencias que le ayudarán a decidir cuál es mejor para sus escenarios.

Expandir tabla

	Insertado	Cmdlet
Trabajo	Los runbooks secundarios se ejecutan en el mismo trabajo que el primario.	Se crea un trabajo independiente para el runbook secundario.
Ejecución	El runbook primario espera a que el runbook secundario se complete antes de continuar.	El runbook primario continúa inmediatamente después de que se inicie el runbook secundario, *o* el runbook primario espera a que finalice el trabajo secundario.
Salida	El runbook primario puede obtener la salida directamente del runbook secundario.	El runbook primario debe obtener la salida del trabajo de runbook secundario, *o* el runbook primario puede obtener la salida directamente del runbook secundario.
Parámetros	Los valores de los parámetros del runbook secundario se especifican por separado y pueden usar cualquier tipo de datos.	Los valores de los parámetros del runbook secundario deben combinarse en una sola tabla hash. Esta tabla hash solo puede incluir tipos de datos simples, de matriz y de objeto que usen la serialización JSON.
Cuenta de Automation	El runbook primario solo puede usar un runbook secundario de la misma cuenta de Automation.	Los runbooks primarios pueden usar un runbook secundario desde cualquier cuenta de Automation, desde la misma suscripción de Azure e incluso desde otra suscripción con la que se tenga conexión.
Publicación	Un runbook secundario debe publicarse antes de publicar el runbook primario.	Un runbook secundario se publica en cualquier momento antes de iniciar el runbook primario.

Invocación de un runbook secundario mediante la ejecución insertada

Para llamar a un runbook en modo insertado desde otro runbook, use el nombre del runbook y proporcione valores para sus parámetros, igual a como usaría una actividad o un cmdlet. De esta manera, todos los runbooks de la misma cuenta de Automation estarán disponibles para que todos los demás los usen. El runbook primario espera a que el runbook secundario se complete antes de pasar a la siguiente línea y cualquier salida se devuelve directamente al primario.

Al invocar un runbook en modo insertado, este se ejecuta en el mismo trabajo que el runbook primario. No hay ninguna indicación en el historial de trabajos del runbook secundario. Todas las excepciones y salidas de flujo pertenecientes al runbook secundario se asocian al primario. Este comportamiento da lugar a que se creen menos trabajos y facilita el seguimiento y la solución de problemas.

Cuando se publica un runbook, cualquier runbook secundario al que llame deberá estar ya estar publicado. Esto se debe a que Azure Automation crea una asociación con los runbooks secundarios cuando compila un runbook. Si aún no se han publicado los runbooks secundarios, el runbook primario parece publicarse correctamente, pero genera una excepción cuando se inicia.

Si obtiene una excepción, puede volver a publicar el runbook primario para que haga referencia correctamente a los runbooks secundarios. No es necesario que vuelva a publicar el runbook primario si cambia algún runbook secundario, porque ya se habrá creado la asociación.

Los parámetros de un runbook secundario al que se llama de modo insertado pueden ser de cualquier tipo de datos, incluidos objetos complejos. No hay ninguna serialización JSON, como sucede cuando se inicia el runbook mediante Azure Portal o con el cmdlet Start-AzAutomationRunbook.

Tipos de runbook

Actualmente, se admite PowerShell 5.1 y solo determinados tipos de runbooks pueden llamarse entre sí:

- Un runbook de PowerShell y un runbook gráfico pueden llamarse entre sí en modo insertado, ya que ambos se basan en PowerShell.
- Un runbook de flujo de trabajo de PowerShell puede llamar a otro runbook de flujo de trabajo de PowerShell insertado.

- Los tipos de PowerShell y los tipos de flujo de trabajo de PowerShell no pueden llamarse entre sí en modo insertado. Deben usar `Start-AzAutomationRunbook`.

> ⓘ **Importante**
>
> - La ejecución de scripts secundarios mediante `.\child-runbook.ps1` no se admite en PowerShell 7.1 y PowerShell 7.2. **Solución alternativa**: Use `Start-AutomationRunbook` (cmdlet interno) o `Start-AzAutomationRunbook` (desde el módulo Az.Automation) para iniciar otro runbook desde un runbook primario.
> - No se admite la ejecución de scripts secundarios mediante runbooks gráficos de flujo de trabajo de PowerShell. **Solución alternativa**: Utilizar `Start-AzAutomationRunbook` (desde el módulo Az.Automation) desde el runbook principal (a través de un bloque InlineScript) para iniciar otro runbook.

El orden de publicación de runbooks solo es importante para runbooks de flujo de trabajo de PowerShell.

Cuando el runbook llama a un runbook secundario de flujo de trabajo de PowerShell mediante la ejecución insertada, usa el nombre del runbook. El nombre debe comenzar por `.\\` para especificar que el script se encuentra en el directorio local.

Ejemplo

En el ejemplo siguiente se inicia un runbook secundario de prueba que acepta un objeto complejo, un valor entero y un valor booleano. Los resultados del runbook secundario se asignan a una variable. En este caso, el runbook secundario es un runbook de flujo de trabajo de PowerShell.

PowerShell

```
$vm = Get-AzVM -ResourceGroupName "LabRG" -Name "MyVM"
$output = PSWF-ChildRunbook -VM $vm -RepeatCount 2 -Restart $true
```

A continuación se muestra el mismo ejemplo con un runbook de PowerShell como secundario.

PowerShell

```
$vm = Get-AzVM -ResourceGroupName "LabRG" -Name "MyVM"
$output = .\PS-ChildRunbook.ps1 -VM $vm -RepeatCount 2 -Restart $true
```

Inicio de un runbook secundario mediante un cmdlet

> **ⓘ Importante**
>
> Si el runbook llama a un runbook secundario con el cmdlet `Start-AzAutomationRunbook` y el parámetro `Wait`, y el runbook secundario genera un resultado de objeto, es posible que se produzca un error en la operación. Para resolver el error, consulte el tema sobre Runbooks secundarios con salida de objeto. En este artículo se muestra cómo implementar la lógica para sondear los resultados mediante el cmdlet Get-AzAutomationJobOutputRecord.

Puede utilizar `Start-AzAutomationRunbook` para iniciar un runbook, tal como se describe en Inicio de un runbook con Windows PowerShell. Existen dos modos de usar este cmdlet:

- El cmdlet devuelve el identificador del trabajo cuando se crea el trabajo para el runbook secundario.
- El cmdlet espera a que finalice el trabajo secundario y devuelve la salida del runbook secundario. Para habilitar este modo, el script especifica el parámetro `Wait`.

El trabajo de un runbook secundario que se inició con un cmdlet se ejecuta de forma independiente al trabajo del runbook primario. Como resultado de este comportamiento, habrá más trabajos que cuando se inicia el runbook en modo insertado y será más difícil realizar un seguimiento de los trabajos. Asimismo, el primario puede iniciar varios runbooks secundarios de forma asincrónica sin tener que esperar a que se complete cada uno de ellos. Para esta ejecución en paralelo que llama a los runbooks secundarios en modo insertado, el runbook primario debe usar la palabra clave parallel.

La salida del runbook secundario no se devuelve al runbook primario de forma confiable debido al tiempo. Además, las variables `$VerbosePreference` y `$WarningPreference`, entre otras, podrían no propagarse a los runbooks secundarios. Para evitar estos problemas, puede iniciar los runbooks secundarios como trabajos de Automation independientes mediante el cmdlet `Start-AzAutomationRunbook` con el parámetro `Wait`. Esta técnica bloquea el runbook primario hasta que se completa el runbook secundario.

Si no quiere bloquear el runbook primario durante la espera, puede iniciar el runbook secundario mediante el cmdlet `Start-AzAutomationRunbook` sin el parámetro `Wait`. En este caso, el runbook debe usar Get-AzAutomationJob para esperar la finalización del trabajo. También debe usar Get-AzAutomationJobOutput y Get-AzAutomationJobOutputRecord para recuperar los resultados.

Los parámetros de un runbook secundario iniciado con un cmdlet se proporcionan como una tabla hash, como se describe en Parámetros de runbook. Solo puede usar tipos de datos simples. Si el runbook tiene un parámetro con un tipo de datos complejo, debe llamarse en modo insertado.

Es posible que se pierda el contexto de la suscripción cuando inicia runbooks secundarios como trabajos individuales. Para que el runbook secundario ejecute los cmdlets del módulo Az en una suscripción de Azure específica, el secundario debe autenticarse en esta suscripción de forma independiente al runbook primario.

Si los trabajos de una misma cuenta de Automation funcionan con varias suscripciones, la selección de una suscripción en uno de los trabajos puede cambiar el contexto de suscripción actualmente seleccionado para otros trabajos. Para evitar esta situación, use `Disable-AzContextAutosave -Scope Process` al principio de cada runbook. Esta acción solo guarda el contexto de esa ejecución de runbook.

Ejemplo

En el ejemplo siguiente se inicia un runbook secundario con parámetros y se espera a que finalice mediante el cmdlet `Start-AzAutomationRunbook` con el parámetro `Wait`. Una vez que finaliza el runbook secundario, el ejemplo recopila la salida del cmdlet del runbook secundario. Para usar `Start-AzAutomationRunbook`, el script debe autenticarse en su suscripción de Azure.

PowerShell

```
# Ensure that the runbook does not inherit an AzContext
Disable-AzContextAutosave -Scope Process

# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext

$params = @{"VMName"="MyVM";"RepeatCount"=2;"Restart"=$true}

Start-AzAutomationRunbook `
    -AutomationAccountName 'MyAutomationAccount' `
    -Name 'Test-ChildRunbook' `
    -ResourceGroupName 'LabRG' `
    -DefaultProfile $AzureContext `
    -Parameters $params -Wait
```

Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario,

haga lo siguiente:

1. En la línea 5, quite `$AzureContext = (Connect-AzAccount -Identity).context`.
2. Reemplace el valor por `$AzureContext = (Connect-AzAccount -Identity -AccountId <ClientId>).context`.
3. Escriba el id. de cliente.

Pasos siguientes

- Para ejecutar el runbook, consulte Inicio de un runbook en Azure Automation.
- Para supervisar la operación del runbook, consulte Salida y mensajes del runbook en Azure Automation.

Last updated on 17/11/2025

Configuración de los parámetros de entrada de un runbook en Automation

Los parámetros de entrada del runbook aumentan la flexibilidad del runbook porque permiten pasarle datos cuando se inicia. Estos parámetros permiten dirigir acciones del runbook a entornos y escenarios específicos. En este artículo se describe la configuración y el uso de los parámetros de entrada en los runbooks.

Los parámetros de entrada se pueden configurar para PowerShell, Flujo de trabajo de PowerShell, gráficos y runbooks de Python. Un runbook puede tener varios parámetros con tipos de datos diferentes o sin parámetros. Los parámetros de entrada pueden ser obligatorios u opcionales y puede usar valores predeterminados para los parámetros opcionales.

La asignación de valores a los parámetros de entrada para un runbook se realiza al iniciarlo. Puede iniciar un runbook desde Azure Portal, un servicio web o PowerShell. También puede iniciarlo como Runbook secundario al que se llama insertado en otro Runbook.

Tipos de entrada

Azure Automation admite varios valores de parámetro de entrada en los distintos tipos de runbook. Los tipos de entrada admitidos para cada tipo de runbook se enumeran en la tabla siguiente.

Expandir tabla

Tipo de runbook	Entradas de parámetros admitidas
PowerShell	- String - Security.SecureString - INT32 - Boolean - DateTime - Array - Collections.Hashtable - Management.Automation.SwitchParameter
Flujo de trabajo de PowerShell	- String - Security.SecureString - INT32 - Boolean - DateTime - Array - Collections.Hashtable - Management.Automation.SwitchParameter

Tipo de runbook	Entradas de parámetros admitidas
Graphical PowerShell	- String - INT32 - INT64 - Boolean - Decimal - DateTime - Object
Python	- String

Configuración de parámetros de entrada en runbooks de PowerShell

Los runbooks de PowerShell y de Flujo de trabajo de PowerShell en Azure Automation admiten parámetros de entrada que se definen mediante las siguientes propiedades.

Expandir tabla

Propiedad	Descripción
Tipo	Necesario. Tipo de datos previsto para el valor del parámetro. Es válido cualquier tipo .NET.
Nombre	Obligatorio. El nombre del parámetro. Este nombre debe ser único dentro del runbook, debe comenzar por una letra y puede contener solo letras, números o caracteres de subrayado.
Mandatory	Opcional. Valor booleano que especifica si el parámetro requiere un valor. Si se establece en True, es necesario proporcionar un valor cuando se inicia el runbook. Si se establece en False, el valor es opcional. Si no especifica un valor para la propiedad `Mandatory`, PowerShell considera el parámetro de entrada opcional de forma predeterminada.
Valor predeterminado	Opcional. Valor que se usa para el parámetro si no se pasa un valor de entrada al iniciar el runbook. El runbook puede establecer un valor predeterminado para cualquier parámetro.

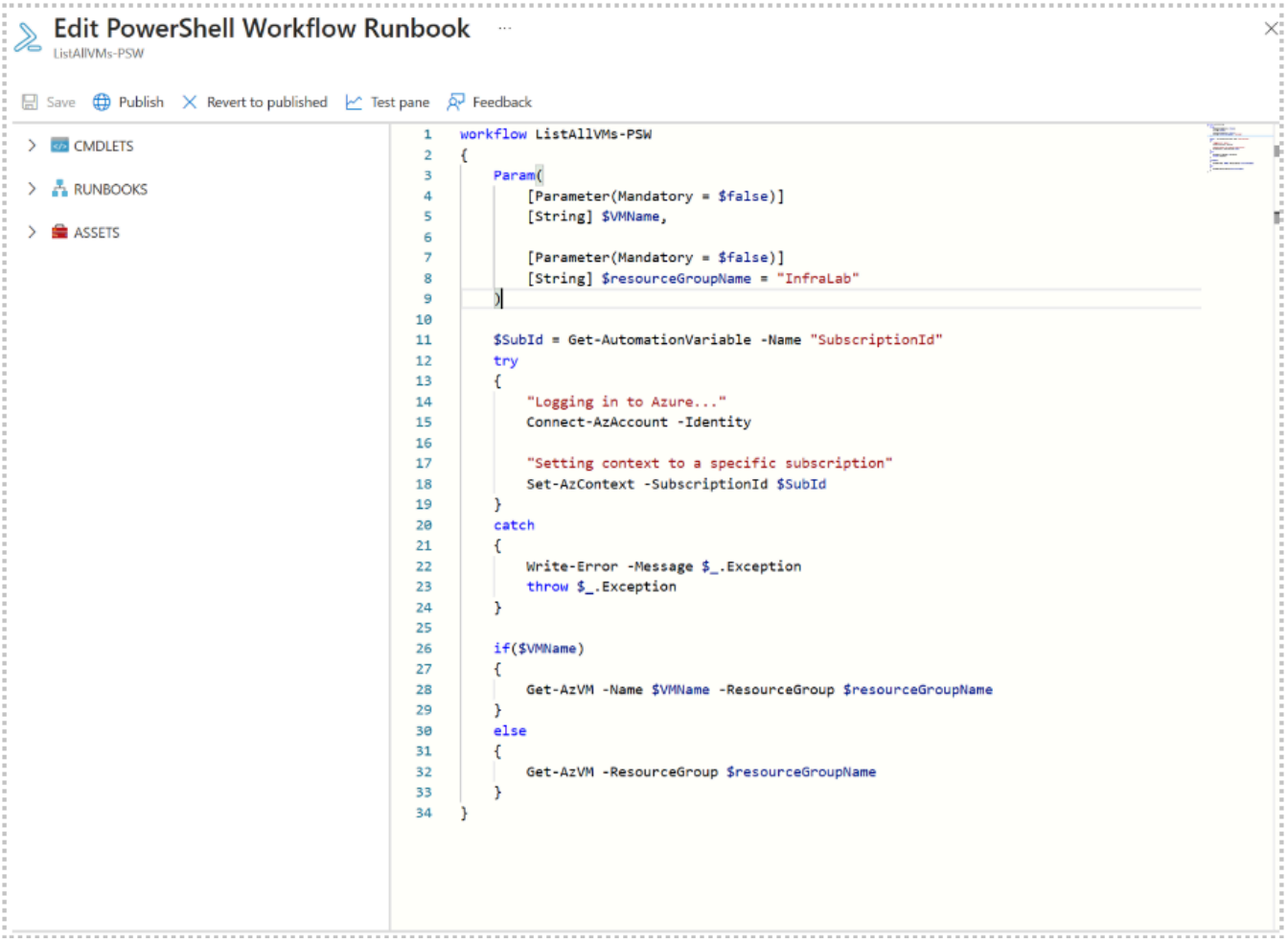
Windows PowerShell admite más atributos de parámetros de entrada de los enumerados anteriormente, como validación, alias y conjuntos de parámetros. Sin embargo, Azure Automation actualmente solo admite las propiedades de parámetros de entrada enumeradas.

Por ejemplo, echemos un vistazo a una definición de parámetro en un runbook de Flujo de trabajo de PowerShell. Esta definición tiene el siguiente formato general, donde los distintos parámetros están separados por comas.

PowerShell

```
Param
(
  [Parameter (Mandatory= $true/$false)]
  [Type] $Name1 = <Default value>,

  [Parameter (Mandatory= $true/$false)]
  [Type] $Name2 = <Default value>
)
```

Ahora vamos a configurar los parámetros de entrada para un runbook de Flujo de trabajo de PowerShell que proporciona detalles sobre las máquinas virtuales, una o varias, de un grupo de recursos. Este runbook tiene dos parámetros, como se muestra en la siguiente captura de pantalla: el nombre de la máquina virtual (`VMName`) y el nombre del grupo de recursos (`resourceGroupName`).



En esta definición de parámetros, los parámetros de entrada son parámetros simples de tipo cadena.

Tenga en cuenta que los runbooks de PowerShell y de Flujo de trabajo de PowerShell admiten todos los tipos simples y complejos, como `Object` o `PSCredential`, como parámetros de entrada. Si el runbook tiene un parámetro de entrada de tipo object, debe usar una tabla hash

de PowerShell con pares nombre-valor para pasar un valor. Supongamos que tiene el siguiente parámetro en un runbook.

PowerShell

```
[Parameter (Mandatory = $true)]
[object] $FullName
```

En este caso, puede pasar el siguiente valor al parámetro.

PowerShell

```
@{"FirstName"="Joe";"MiddleName"="Bob";"LastName"="Smith"}
```

Para runbooks de PowerShell 7.1, proporcione parámetros de entrada de matriz en el formato siguiente:

Expandir tabla

Nombre	Valor
TESTPARAMETER	does,this,even,work

> **Nota:**
>
> Cuando no se pasa un valor a un parámetro de cadena opcional con un valor predeterminado NULL, el valor del parámetro es una cadena vacía en lugar de NULL.

Configuración de parámetros de entrada en Runbooks gráficos

Para ilustrar la configuración de los parámetros de entrada para un runbook gráfico, vamos a crear un runbook que proporciona detalles sobre las máquinas virtuales, una o varias, de un grupo de recursos. Para más información, consulte Mi primer runbook gráfico.

Un runbook gráfico usa estas actividades de runbooks principales:

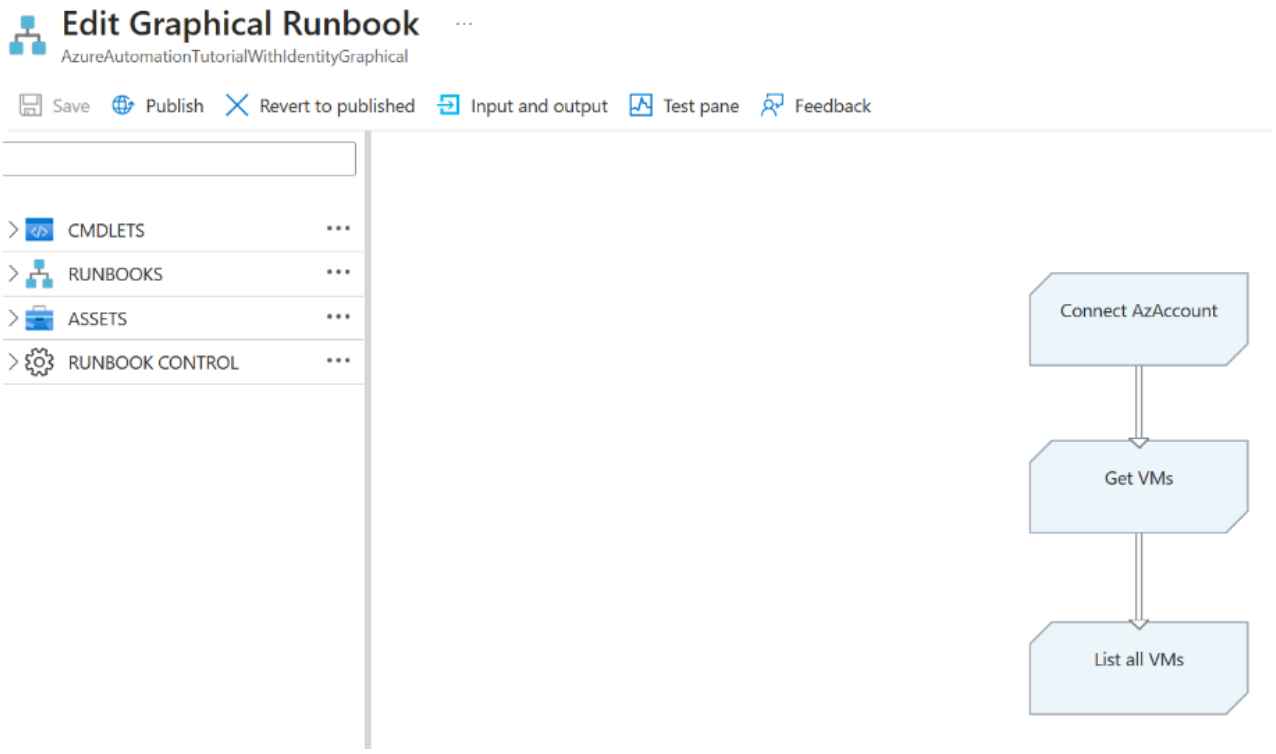
- Autentíquese con Azure mediante la identidad administrada configurada para la cuenta de Automation.
- Definición del cmdlet Get-AzVM para obtener las propiedades de la máquina virtual.
- Uso de la actividad Write-Output para escribir en la salida los nombres de máquina virtual.

La actividad `Get-AzVM` define dos entradas, el nombre de la VM y el nombre del grupo de recursos. Puesto que estos nombres pueden ser diferentes cada vez que se inicia el runbook, debe agregar parámetros de entrada al runbook para aceptar estas entradas. Consulte Creación gráfica en Azure Automation.

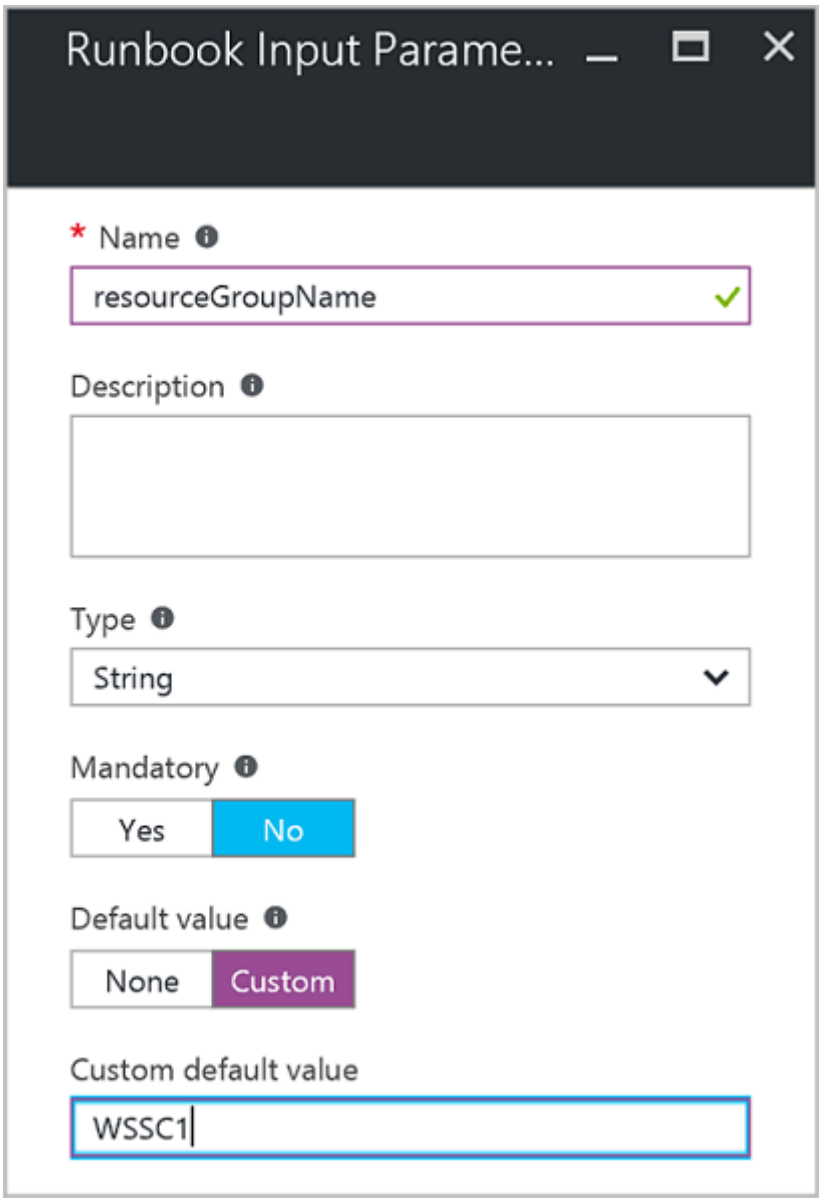
Siga estos pasos para configurar los parámetros de entrada.

1. Seleccione el runbook gráfico de la página Runbooks y, a continuación, haga clic en **Editar**.

2. En el editor gráfico, haga clic en el botón **Entrada y salida** y, a continuación, en **Agregar entrada** para abrir el panel Parámetro de entrada de runbook.



3. El control Entrada y salida muestra una lista de parámetros de entrada definidos para el runbook. Puede agregar un nuevo parámetro de entrada o editar la configuración de uno existente. Para agregar un nuevo parámetro al runbook, haga clic en **Agregar entrada** para abrir la hoja **Parámetro de entrada de runbook**, donde puede configurar parámetros mediante las propiedades definidas en Creación gráfica en Azure Automation.



4. Cree dos parámetros con las siguientes propiedades que usará la actividad `Get-AzVM` y, a continuación, haga clic en **Aceptar**.

   - Parámetro 1:
     - **Nombre** -- **VMName**
     - **Tipo**: cadena
     - **Obligatorio** -- **No**

   - Parámetro 2:
     - **Nombre** -- **resourceGroupName**
     - **Tipo**: cadena
     - **Obligatorio** -- **No**
     - **Valor predeterminado** -- **Custom**
     - Valor predeterminado personalizado: nombre del grupo de recursos que contiene las máquinas virtuales

5. Puede ver los parámetros en el control Entrada y salida.

6. Haga clic de nuevo en **Aceptar** y, a continuación, haga clic en **Guardar**.

7. Haga clic en **Publicar** para publicar el runbook.

Configuración de parámetros de entrada en runbooks de Python

A diferencia de los runbooks gráficos, de PowerShell y de Flujo de trabajo de PowerShell, los runbooks de Python no toman parámetros con nombre. El editor de runbooks analiza todos los parámetros de entrada como una matriz de valores de argumento. Para obtener acceso a la matriz, puede importar el módulo `sys` al script de Python y usar la matriz `sys.argv`. Es importante tener en cuenta que el primer elemento de la matriz, `sys.argv[0]`, es el nombre del script. Por lo tanto, el primer parámetro de entrada real es `sys.argv[1]`.

Para obtener un ejemplo de cómo usar los parámetros de entrada en un runbook de Python, vea My first Python runbook in Azure Automation (Mi primer runbook de Python en Azure Automation).

> ⓘ **Nota:**
>
> Los argumentos con espacios no se admiten actualmente. Como solución alternativa, podría usar \\t además de \\n.

Asignación de valores a parámetros de entrada en Runbooks

En esta sección se describen varias maneras de pasar valores a los parámetros de entrada en los runbooks. Puede asignar valores de parámetro en el momento de:

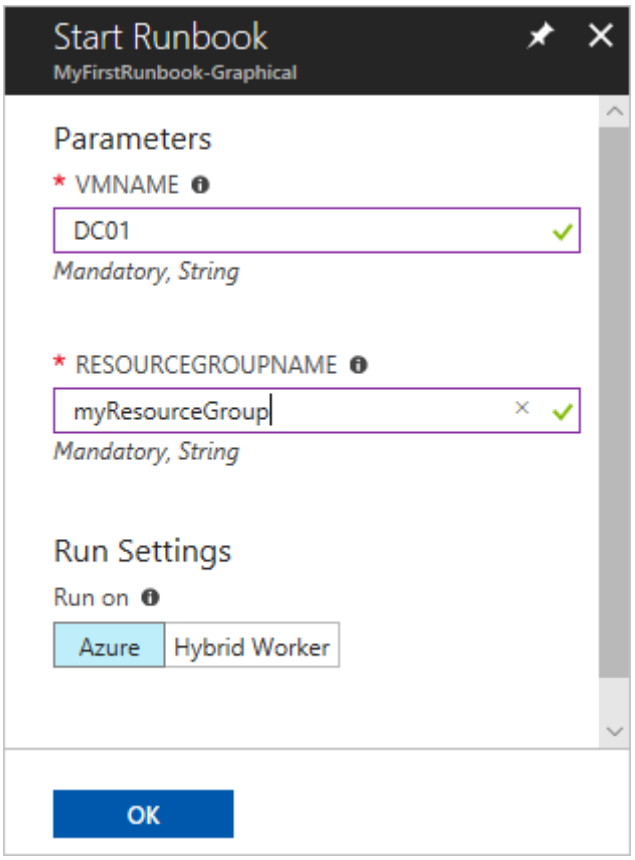
- Iniciar un runbook
- Probar un runbook
- Vinculación de una programación para el runbook
- Creación de un webhook para el runbook

Inicio de un Runbook y asignación de parámetros

Se puede iniciar un runbook de muchas maneras: mediante Azure Portal, con un webhook, con cmdlets de PowerShell, con la API REST o con el SDK.

Inicio de un runbook publicado mediante Azure Portal y asignación de parámetros

Al iniciar el runbook en Azure Portal, se abre la hoja **Iniciar runbook** y puede escribir los valores de los parámetros que ha creado.



En la etiqueta situada debajo del cuadro de entrada, puede ver las propiedades que se han establecido para definir los atributos del parámetro; por ejemplo, obligatorio u opcional, tipo o valor predeterminado. En el globo de ayuda junto al nombre del parámetro también se define la información importante que necesita para tomar decisiones acerca de los valores de entrada de los parámetros.

> ⓘ **Nota:**
>
> Los parámetros de cadena admiten valores vacíos de tipo cadena. Si escribe `[EmptyString]` en el cuadro del parámetro de entrada, pasará una cadena vacía al parámetro. Además, los parámetros de cadena no admiten valores NULL. Si no pasa ningún valor a un parámetro de cadena, PowerShell lo interpreta como NULL.

Inicio de un runbook publicado mediante cmdlets de PowerShell y asignación de parámetros

- **Cmdlets de Azure Resource Manager:** puede iniciar un runbook de Automation creado en un grupo de recursos mediante Start-AzureRmAutomationRunbook.

PowerShell

```
   $params = @{"VMName"="WSVMClassic";"resourceGroupeName"="WSVMClassicSG"}

   Start-AzAutomationRunbook -AutomationAccountName "TestAutomation" -Name
"Get-AzureVMGraphical" –ResourceGroupName $resourceGroupName -Parameters
$params
```

- **Cmdlets del modelo de implementación clásico de Azure:** puede iniciar un runbook de automatización creado en un grupo de recursos predeterminado mediante Start-AzureAutomationRunbook.

PowerShell

```
   $params = @{"VMName"="WSVMClassic"; "ServiceName"="WSVMClassicSG"}

   Start-AzureAutomationRunbook -AutomationAccountName "TestAutomation" -Name
"Get-AzureVMGraphical" -Parameters $params
```

> **Nota:**
>
> Cuando se inicia un runbook mediante cmdlets de PowerShell, se crea un parámetro predeterminado, `MicrosoftApplicationManagementStartedBy`, con el valor `PowerShell`. Puede ver este parámetro en el panel Detalles del trabajo.

Inicio de un runbook con un SDK y asignación de parámetros

- **Método Azure Resource Manager:** puede iniciar un runbook mediante el SDK de un lenguaje de programación. A continuación, se muestra un fragmento de código C# para iniciar un Runbook en su cuenta de Automation. Puede ver todo el código en nuestro repositorio de GitHub.

C#

```
public Job StartRunbook(string runbookName, IDictionary<string, string>
parameters = null)
{
  var response = AutomationClient.Jobs.Create(resourceGroupName,
automationAccount, new JobCreateParameters
  {
    Properties = new JobCreateProperties
    {
```

```
      Runbook = new RunbookAssociationProperty
      {
        Name = runbookName
      },
      Parameters = parameters
    }
  });
  return response.Job;
}
```

- **Método que usa el modelo de implementación clásico de Azure:** puede iniciar un runbook mediante el SDK de un lenguaje de programación. A continuación, se muestra un fragmento de código C# para iniciar un Runbook en su cuenta de Automation. Puede ver todo el código en nuestro repositorio de GitHub.

C#

```
public Job StartRunbook(string runbookName, IDictionary<string, string>
parameters = null)
{
  var response = AutomationClient.Jobs.Create(automationAccount, new
JobCreateParameters
  {
    Properties = new JobCreateProperties
    {
      Runbook = new RunbookAssociationProperty
      {
        Name = runbookName
      },
      Parameters = parameters
    }
  });
  return response.Job;
}
```

Para iniciar este método, cree un diccionario para almacenar los parámetros del runbook, `VMName` y `resourceGroupName`, así como sus valores. A continuación, inicie el Runbook. A continuación, se muestra el fragmento de código C# para llamar al método definido más arriba.

C#

```
IDictionary<string, string> RunbookParameters = new Dictionary<string, string>
();

// Add parameters to the dictionary.
RunbookParameters.Add("VMName", "WSVMClassic");
RunbookParameters.Add("resourceGroupName", "WSSC1");
```

```
//Call the StartRunbook method with parameters
StartRunbook("Get-AzureVMGraphical", RunbookParameters);
```

Inicio de un runbook con la API REST y asignación de parámetros

Puede crear e iniciar un trabajo de runbook con la API de REST de Azure Automation mediante el método `PUT` con el siguiente URI de solicitud:

`https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Automation/automationAccounts/{automationAccountName}/jobs/{jobName}?api-version=2017-05-15-preview`

En el identificador URI de solicitud, reemplace los siguientes parámetros:

- `subscriptionId`: el identificador de la suscripción a Azure.
- `resourceGroupName`: nombre del grupo de recursos de la cuenta de Automation.
- `automationAccountName`: nombre de la cuenta de Automation hospedada en el servicio en la nube especificado.
- `jobName`: GUID del trabajo. Los GUID de PowerShell se pueden crear mediante `[GUID]::NewGuid().ToString()*`.

Para pasar parámetros al trabajo de runbook, use el cuerpo de la solicitud. Admite la siguiente información, proporcionada en formato JSON:

- Nombre del Runbook: obligatorio. El nombre del Runbook para que se inicie el trabajo.
- Parámetros del Runbook: opcionales. Diccionario de la lista de parámetros en formato (nombre, valor), donde el nombre es de tipo cadena y el valor puede ser cualquier valor JSON válido.

Si desea iniciar el runbook **Get-AzureVMTextual** creado antes con `VMName` y `resourceGroupName` como parámetros, use el siguiente formato JSON para el cuerpo de la solicitud.

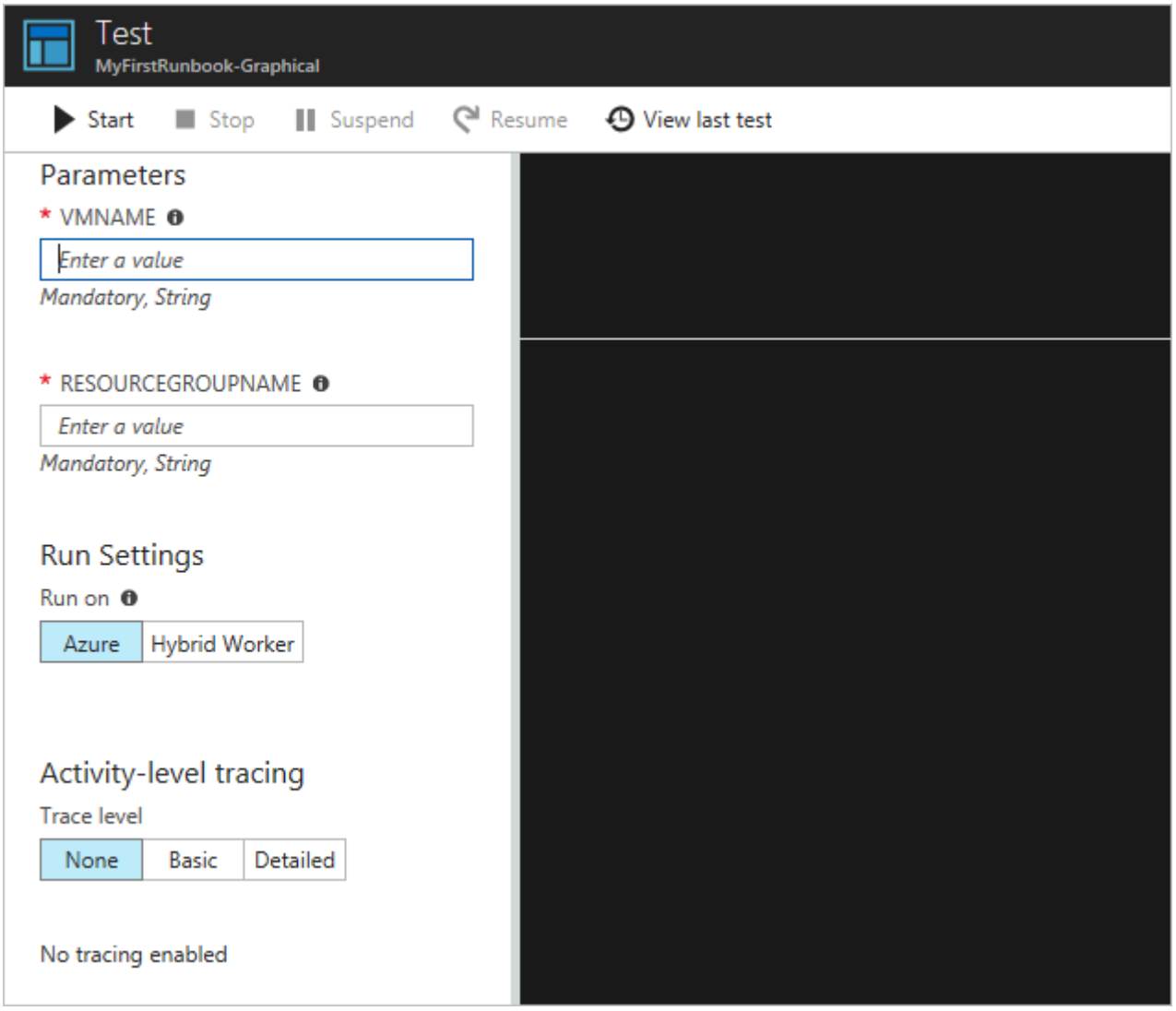
JSON

```
{
  "properties":{
    "runbook":{
    "name":"Get-AzureVMTextual"},
  "parameters":{
     "VMName":"WindowsVM",
     "resourceGroupName":"ContosoSales"}
    }
}
```

Si el trabajo se crea correctamente, se devuelve un código de estado HTTP 201. Para más información sobre los encabezados y el cuerpo de la respuesta, consulte Creación de un trabajo de runbook mediante la API REST.

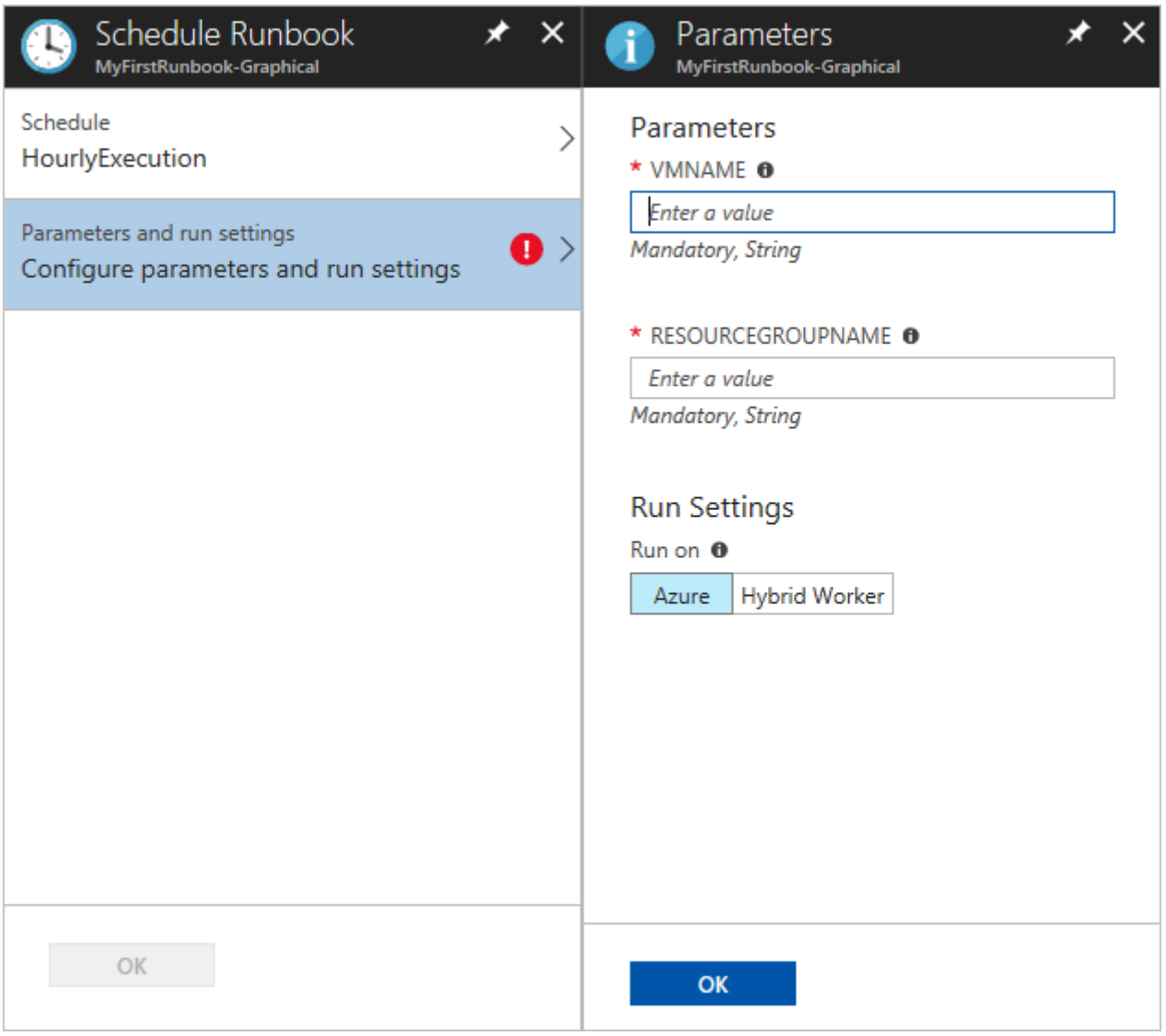
Prueba de un Runbook y asignación de parámetros

Al probar la versión de borrador del runbook mediante la opción de prueba, se abre la página Prueba. Use esta página para configurar los valores de los parámetros que ha creado.



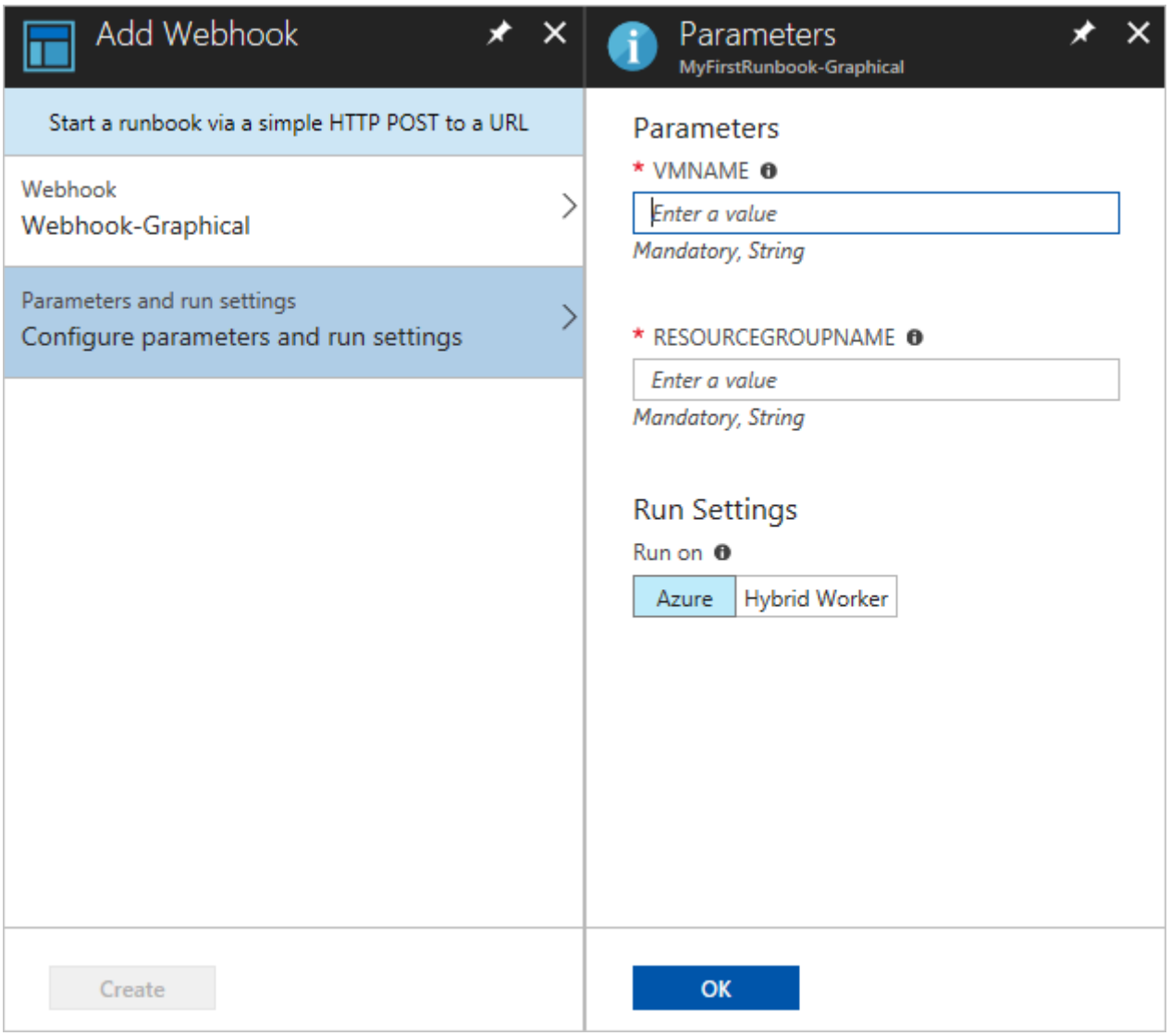
Vinculación de una programación a un Runbook y asignación de parámetros

También puede vincular una programación a su Runbook para que este se inicie en un momento determinado. Asigne los parámetros de entrada al crear la programación y el runbook los usará cuando se inicie mediante la programación. No se puede guardar la programación hasta que se proporcionan todos los valores de los parámetros obligatorios.

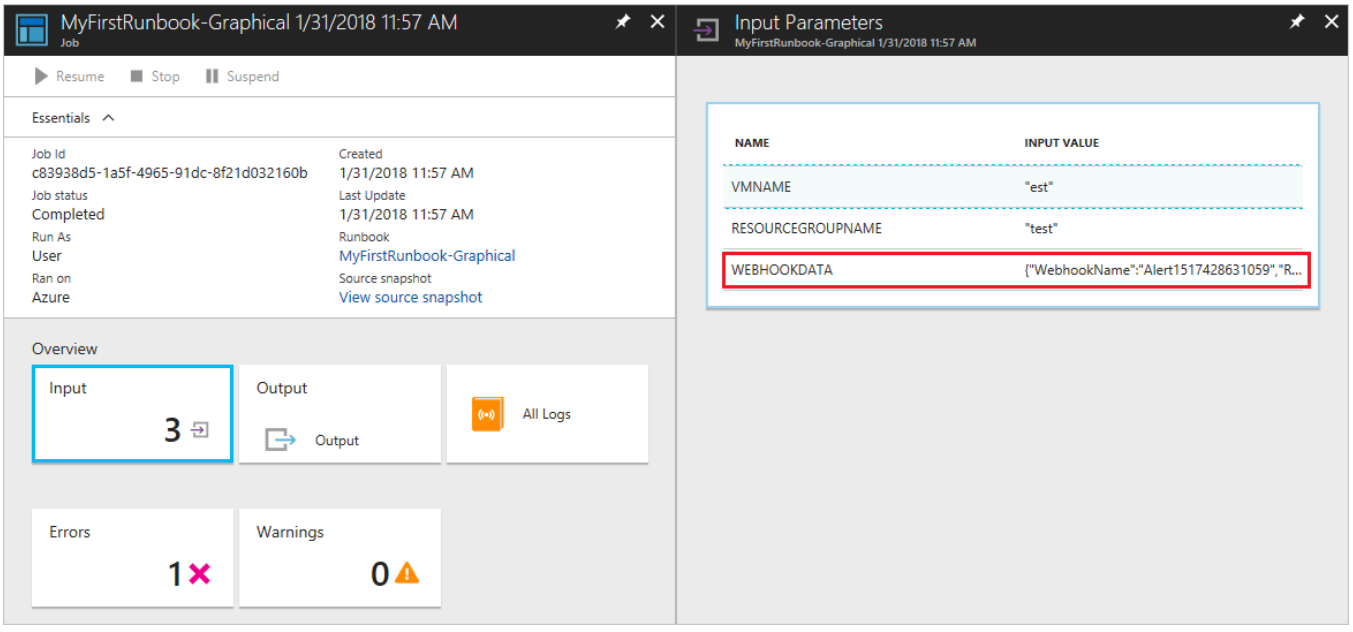


Creación de un Webhook para un Runbook y asignación de parámetros

Puede crear un Webhook para el Runbook y configurar los parámetros de entrada del Runbook. No se puede guardar el webhook hasta que se proporcionen todos los valores de los parámetros obligatorios.



Cuando se ejecuta un runbook mediante un webhook, se envía un parámetro de entrada predefinido, `[WebhookData](automation-webhooks.md)`, junto con los parámetros de entrada que defina.



Paso de un objeto JSON a un runbook

Puede ser útil almacenar los datos que desea pasar a un runbook en un archivo JSON. Por ejemplo, podría crear un archivo JSON que contiene todos los parámetros que desea pasar a un runbook. Para hacerlo, debe convertir el código JSON en una cadena y luego convertir esta cadena en un objeto de PowerShell antes de pasárselo al runbook.

Esta sección utiliza un ejemplo en el que un script de PowerShell llama a Start-AzAutomationRunbook para iniciar un runbook de PowerShell, pasando el contenido del archivo JSON al runbook. El runbook de PowerShell inicia una máquina virtual de Azure y recupera los parámetros de la máquina virtual desde el objeto JSON.

Creación del archivo JSON

Escriba el código siguiente en un archivo de texto y guárdelo como **test.json** en algún lugar del equipo local.

JSON

```
{
   "VmName" : "TestVM",
   "ResourceGroup" : "AzureAutomationTest"
}
```

Creación del runbook

Cree un nuevo runbook de PowerShell llamado **Test-Json** en Azure Automation.

Para aceptar los datos de JSON, el runbook debe tomar un objeto como parámetro de entrada. De ese modo, el runbook puede usar las propiedades definidas en el archivo JSON.

PowerShell

```
Param(
     [parameter(Mandatory=$true)]
     [object]$json
)

# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process

# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
```

```
DefaultProfile $AzureContext

# Convert object to actual JSON
$json = $json | ConvertFrom-Json

# Use the values from the JSON object as the parameters for your command
Start-AzVM -Name $json.VMName -ResourceGroupName $json.ResourceGroup -DefaultProfile
$AzureContext
```

Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:

1. En la línea 10, quite `$AzureContext = (Connect-AzAccount -Identity).context`.
2. Reemplace el valor por `$AzureContext = (Connect-AzAccount -Identity -AccountId <ClientId>).context`.
3. Escriba el id. de cliente.

Guarde y publique este runbook en su cuenta de Automation.

Llamada al runbook desde PowerShell

Ahora puede llamar al runbook desde la máquina local mediante Azure PowerShell.

1. Inicie sesión en Azure como se muestra. Después, se le pedirá que escriba las credenciales de Azure.

   PowerShell
   ```
   Connect-AzAccount
   ```

   > ⓘ **Nota:**
   >
   > En el caso de los runbooks de PowerShell, `Add-AzAccount` y `Add-AzureRMAccount` son alias de `Connect-AzAccount`. Tenga en cuenta que estos alias no están disponibles para los runbooks gráficos. Un runbook gráfico solo puede usar `Connect-AzAccount`.

2. Obtenga el contenido del archivo JSON guardado y conviértalo en una cadena. `JsonPath` indica la ruta de acceso donde se guardó el archivo JSON.

   PowerShell
   ```
   $json =  (Get-content -path 'JsonPath\test.json' -Raw) | Out-string
   ```

3. Convierta el contenido de la cadena `$json` en un objeto de PowerShell.

PowerShell

```
$JsonParams = @{"json"=$json}
```

4. Cree una tabla hash para los parámetros de `Start-AzAutomationRunbook`.

PowerShell

```
$RBParams = @{
     AutomationAccountName = 'AATest'
     ResourceGroupName = 'RGTest'
     Name = 'Test-Json'
     Parameters = $JsonParams
}
```

Tenga en cuenta que está estableciendo el valor de `Parameters` en el objeto de PowerShell que contiene los valores del archivo JSON.

5. Inicie el runbook.

PowerShell

```
$job = Start-AzAutomationRunbook @RBParams
```

Pasos siguientes

- Para preparar un runbook de texto, consulte Edición de runbooks de texto en Azure Automation.
- Para preparar un runbook gráfico, consulte Creación de runbooks gráficos en Azure Automation.

Last updated on 17/11/2025

Administración de runbooks en Azure Automation

Para agregar un runbook a Azure Automation, puede crear uno o importar uno existente desde un archivo o desde la Galería de runbooks. En este artículo se proporciona información sobre la administración de un runbook, así como sobre patrones recomendados y procedimientos recomendados para el diseño de runbooks. Puede encontrar todos los detalles sobre el acceso a los módulos y los runbooks de la comunidad en Galerías de runbooks y módulos para Azure Automation.

Crear un manual de procedimientos

Cree un runbook nuevo en Azure Automation mediante Azure Portal o PowerShell. Una vez creado el runbook, se puede editar siguiendo la información de las secciones siguientes:

- Editar un runbook textual en Azure Automation
- Aprendizaje de los conceptos clave del flujo de trabajo de PowerShell para los runbooks de Automation
- Administración de paquetes de Python 2 en Azure Automation
- Administración de paquetes de Python 3 (versión preliminar) en Azure Automation

Crear un runbook en el portal de Azure

1. Inicie sesión en Azure Portal.
2. Busque y seleccione **Cuentas de automatización**.
3. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
4. Desde la cuenta de Automation, seleccione **Runbooks** en **Automatización de procesos** para abrir la lista de runbooks.
5. Haga clic en **Crear un libro de procedimientos**.
   a. Asigne un nombre al runbook.
   b. Desde el menú desplegable **Tipo de runbook**. Seleccione su tipo. El nombre del runbook debe empezar por una letra y puede tener letras, números, guiones bajos y guiones.
   c. Seleccione la **versión del entorno de ejecución**.
   d. Escriba la **descripción** correspondiente.
6. Haga clic en **Crear** para crear el runbook.

Creación de un runbook con PowerShell

Use el cmdlet New-AzAutomationRunbook para crear un runbook vacío. Use el parámetro `Type` para especificar uno de los tipos de runbook definidos para `New-AzAutomationRunbook`.

En el ejemplo siguiente se muestra cómo crear un nuevo runbook vacío.

Azure PowerShell

```
$params = @{
    AutomationAccountName = 'MyAutomationAccount'
    Name                  = 'NewRunbook'
    ResourceGroupName     = 'MyResourceGroup'
    Type                  = 'PowerShell'
}
New-AzAutomationRunbook @params
```

Importación de un runbook

Puede importar un script de PowerShell o de flujo de trabajo de PowerShell (**.ps1**), un runbook gráfico (**.graphrunbook**) o un script de Python 2 o Python 3 (**.py**) para crear su propio runbook. Especifique el tipo de runbook que se creará durante la importación teniendo en cuenta las consideraciones siguientes.

- Puede importar un archivo **.ps1** que no contenga un flujo de trabajo, ya sea en un runbook de PowerShell o en un runbook de flujo de trabajo de PowerShell. Si lo importa en un runbook de flujo de trabajo de PowerShell, se convertirá en un flujo de trabajo. En este caso, se incluyen comentarios en el runbook para describir los cambios realizados.
- Solo puede importar un archivo **.ps1** que contenga un PowerShell Workflow en un runbook de PowerShell Workflow. Si el archivo contiene varios flujos de trabajo de PowerShell, se produce un error en la importación. Debe guardar cada flujo de trabajo en su propio archivo e importar cada uno por separado.
- No debe importar un archivo **.ps1** que contiene un flujo de trabajo de PowerShell en un runbook de PowerShell, ya que el motor de scripts de PowerShell no lo reconoce.
- Importe solo un archivo **.graphrunbook** en un runbook gráfico nuevo.

Importación de un runbook desde Azure Portal

Puede usar el siguiente procedimiento para importar un archivo de script en Azure Automation.

> ⓘ **Nota:**
>
> Solo puede importar un archivo **.ps1** en un runbook de flujo de trabajo de PowerShell mediante el portal.

1. En el portal de Azure, busque y seleccione **Cuentas de automatización**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. Desde la cuenta de Automation, seleccione **Runbooks** en **Automatización de procesos** para abrir la lista de runbooks.
4. Haga clic en **Importar un runbook**. Puede seleccionar cualquiera de las siguientes opciones:
   a. **Buscar un archivo**: esta opción selecciona un archivo de la máquina local.
   b. **Examinar desde la galería**: puede examinar y seleccionar un runbook existente de la galería.
5. Seleccione el archivo.
6. Si el campo **Nombre** está habilitado, tiene la opción de cambiar el nombre del runbook. El nombre debe empezar por una letra y puede tener letras, números, guiones bajos y guiones.
7. El **tipo de libro de procedimientos** se completa automáticamente, pero puede cambiar el tipo después de tener en cuenta las restricciones aplicables.
8. La **versión del entorno de ejecución** se rellena automáticamente o puede elegir la versión de la lista desplegable.
9. Haga clic en **Import**. El nuevo runbook aparece en la lista de runbooks de la cuenta de Automation.
10. Tiene que publicar el runbook antes de poder ejecutarlo.

> ⓘ **Nota:**
>
> Después de importar un runbook gráfico, puede convertirlo en otro tipo, pero no puede convertir un runbook gráfico en un runbook de texto.

Importar un runbook con PowerShell

Use el cmdlet Import-AzAutomationRunbook para importar un archivo de script como un runbook de borrador. Si ya existe el runbook, se producirá un error de importación, a menos que utilice el parámetro `Force` con el cmdlet.

En los ejemplos siguientes se muestra cómo importar un archivo de script en un runbook.

Azure PowerShell

```
$params = @{
    AutomationAccountName = 'MyAutomationAccount'
    Name                  = 'Sample_TestRunbook'
    ResourceGroupName     = 'MyResourceGroup'
    Type                  = 'PowerShell'
    Path                  = 'C:\Runbooks\Sample_TestRunbook.ps1'
}
Import-AzAutomationRunbook @params
```

Gestionar recursos

Si el runbook crea un recurso, el script debe comprobar que el recurso ya exista antes de intentar crearlo. Este es un ejemplo básico.

PowerShell

```
$vmName = 'WindowsVM1'
$rgName = 'MyResourceGroup'
$myCred = Get-AutomationPSCredential 'MyCredential'

$vmExists = Get-AzResource -Name $vmName -ResourceGroupName $rgName
if (-not $vmExists) {
    Write-Output "VM $vmName does not exist, creating"
    New-AzVM -Name $vmName -ResourceGroupName $rgName -Credential $myCred
} else {
    Write-Output "VM $vmName already exists, skipping"
}
```

Recuperación de detalles del registro de actividad

Puede recuperar detalles del runbook, como la persona o la cuenta que ha iniciado un runbook, del registro de actividad para la cuenta de Automation. En el siguiente ejemplo de PowerShell se proporciona el último usuario que ha ejecutado el runbook especificado.

PowerShell

```
$rgName = 'MyResourceGroup'
$accountName = 'MyAutomationAccount'
$runbookName = 'MyRunbook'
$startTime = (Get-Date).AddDays(-1)

$params = @{
    ResourceGroupName = $rgName
    StartTime         = $startTime
}
$JobActivityLogs = (Get-AzLog @params).Where( { $_.Authorization.Action -eq
```

```
'Microsoft.Automation/automationAccounts/jobs/write' })

$JobInfo = @{}
foreach ($log in $JobActivityLogs) {
    # Get job resource
    $JobResource = Get-AzResource -ResourceId $log.ResourceId

    if ($null -eq $JobInfo[$log.SubmissionTimestamp] -and
$JobResource.Properties.Runbook.Name -eq $runbookName) {
        # Get runbook
        $jobParams = @{
            ResourceGroupName     = $rgName
            AutomationAccountName = $accountName
            Id                    = $JobResource.Properties.JobId
        }
        $Runbook = Get-AzAutomationJob @jobParams | Where-Object RunbookName -EQ
$runbookName

        # Add job information to hashtable
        $JobInfo.Add($log.SubmissionTimestamp, @($Runbook.RunbookName, $Log.Caller,
$JobResource.Properties.jobId))
    }
}
$JobInfo.GetEnumerator() | Sort-Object Key -Descending | Select-Object -First 1
```

Seguimiento del progreso

Se recomienda crear los runbooks para que tengan una naturaleza modular, con una lógica que se pueda reutilizar y reiniciar fácilmente. El seguimiento del progreso de un runbook ayuda a garantizar que la lógica del runbook se ejecute correctamente si surgen incidencias.

Puede realizar un seguimiento del progreso de un runbook mediante un origen externo, como una cuenta de almacenamiento, una base de datos o archivos compartidos. Cree lógica en el runbook para comprobar primero el estado de la última acción que se ha realizado. Luego, en función de los resultados de la comprobación, la lógica puede omitir o continuar tareas específicas del runbook.

Evitación de trabajos simultáneos

Algunos runbooks se comportan de forma extraña si se ejecutan en varios trabajos al mismo tiempo. En este caso, es importante que un runbook implemente la lógica para establecer si ya hay un trabajo en ejecución. Este es un ejemplo básico.

PowerShell

```
# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process
```

```
# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext

# Check for already running or new runbooks
$runbookName = "runbookName"
$resourceGroupName = "resourceGroupName"
$automationAccountName = "automationAccountName"

$jobs = Get-AzAutomationJob -ResourceGroupName $resourceGroupName -
AutomationAccountName $automationAccountName -RunbookName $runbookName -DefaultProfile
$AzureContext

# Ranking all the active jobs
$activeJobs = $jobs | where {$_.status -eq 'Running' -or $_.status -eq 'Queued' -or
$_.status -eq 'New' -or $_.status -eq 'Activating' -or $_.status -eq 'Resuming'} |
Sort-Object -Property CreationTime
$jobRanking = @()
$rank = 0
ForEach($activeJob in $activeJobs)
{
    $rank = $rank + 1
    $activeJob | Add-Member -MemberType NoteProperty -Name jobRanking -Value $rank -
Force
    $jobRanking += $activeJob
}

$AutomationJobId = $PSPrivateMetadata.JobId.Guid
$currentJob = $activeJobs | where {$_.JobId -eq $AutomationJobId}
$currentJobRank = $currentJob.jobRanking

# Only allow the Job with Rank = 1 to start processing.
If($currentJobRank -ne "1")
{
    Write-Output "$(Get-Date -Format yyyy-MM-dd-hh-mm-ss.ffff) Concurrency check
failed as Current Job Ranking is not 1 but $($currentJobRank) therefore exiting..."
    Exit
} Else
{
    Write-Output "$(Get-Date -Format yyyy-MM-dd-hh-mm-ss.ffff) Concurrency check
passed. Start processing.."
}
```

Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:

1. En la línea 5, elimine `$AzureContext = (Connect-AzAccount -Identity).context`,

2. Sustitúyalo por `$AzureContext = (Connect-AzAccount -Identity -AccountId <ClientId>).context`, y
3. Escriba el id. de cliente.

> **Nota:**
>
> En el caso de los trabajos híbridos de PowerShell 7.2, realice cambios en la línea 28. Reemplace `$PSPrivateMetadata.JobId.Guid` por `$env:PSPrivateMetaData`.

Control de errores transitorios en un script dependiente del tiempo

Los runbooks deben ser sólidos y capaces de controlar los errores, incluidos los transitorios, que pueden provocar que se reinicien o fallen. Si se produce un error en un runbook, Azure Automation vuelve a intentarlo.

Si su runbook normalmente se ejecuta dentro de un límite de tiempo, haga que el script incluya lógica para controlar el tiempo de ejecución. Esta comprobación garantiza la ejecución de operaciones, como el inicio, el apagado o el escalado horizontal, solo durante horas específicas.

> **Nota:**
>
> La hora local en el proceso de espacio aislado de Azure se establece en UTC. Al realizar cálculos de fecha y hora en sus runbooks, debe tenerse en cuenta este hecho.

Lógica de reintento en el manual de procedimientos para evitar fallos transitorios

Los runbooks suelen realizar llamadas a sistemas remotos, como Azure a través de ARM, Azure Resource Graph, servicios SQL y otros servicios web. Cuando el sistema al que llaman los runbooks está ocupado, no está disponible temporalmente o aplica limitación bajo carga, las llamadas pueden producir errores en tiempo de ejecución. Para crear resistencia en los runbooks, debe implementar la lógica de reintento al realizar las llamadas para que los runbooks puedan controlar un problema transitorio sin errores.

Para obtener más información, consulte patrón de reintento y REST general y directrices de reintento.

Ejemplo 1: Si el runbook realiza solo una o dos llamadas

PowerShell

```
$searchServiceURL = "https://$searchServiceName.search.windows.net"
$resource = Get-AzureRmResource -ResourceType "Microsoft.Search/searchServices" -
ResourceGroupName $searchResourceGroupName -ResourceName  $searchServiceName -
ApiVersion 2015-08-19
$searchAPIKey = (Invoke-AzureRmResourceAction -Action listAdminKeys -ResourceId
$resource.ResourceId -ApiVersion 2015-08-19 -Force).PrimaryKey
```

Al llamar a `Invoke-AzureRmResourceAction`, puede observar errores transitorios. En este escenario, se recomienda implementar el siguiente patrón básico en torno a la llamada al cmdlet.

PowerShell

```
$searchServiceURL = "https://$searchServiceName.search.windows.net"
$resource = Get-AzureRmResource -ResourceType "Microsoft.Search/searchServices" -
ResourceGroupName $searchResourceGroupName -ResourceName  $searchServiceName -
ApiVersion 2015-08-19

    # Adding in a retry
    $Stoploop = $false
    $Retrycount = 0

    do {
        try   {
               $searchAPIKey = (Invoke-AzureRmResourceAction -Action listAdminKeys -
ResourceId $resource.ResourceId -ApiVersion 2015-08-19 -Force).PrimaryKey
               write-verbose "Invoke-AzureRmResourceAction on $resource.ResourceId
completed"
               $Stoploop = $true
              }
        catch {
               if ($Retrycount -gt 3)
                 {
                  Write-verbose "Could not Invoke-AzureRmResourceAction on
$resource.ResourceId after 3 retrys."
                  $Stoploop = $true
                 }
               else
                 {
                  Write-verbose "Could not Invoke-AzureRmResourceAction on
$resource.ResourceId retrying in 30 seconds..."
                  Start-Sleep -Seconds 30
                  $Retrycount = $Retrycount + 1
                 }
               }
        }
    While ($Stoploop -eq $false)
```

> ⓘ **Nota:**
>
> El reintento de la llamada es de hasta tres veces, suspendiendo durante 30 segundos cada vez.

Ejemplo 2: si el runbook realiza llamadas remotas frecuentes

Si el runbook realiza llamadas remotas frecuentes, podría experimentar problemas transitorios de tiempo de ejecución. Cree una función que implemente la lógica de reintento para cada llamada que se realiza y pase la llamada como un bloque de script que se va a ejecutar.

PowerShell

```
Function ResilientRemoteCall {

         param(
               $scriptblock
               )

         $Stoploop = $false
         $Retrycount = 0

         do {
             try   {
                    Invoke-Command -scriptblock $scriptblock
                    write-verbose "Invoked $scriptblock completed"
                    $Stoploop = $true
                   }
             catch {
                    if ($Retrycount -gt 3)
                      {
                       Write-verbose "Invoked $scriptblock failed 3 times and we will
not try again."
                       $Stoploop = $true
                      }
                    else
                      {
                       Write-verbose "Invoked $scriptblock failed  retrying in 30
seconds..."
                       Start-Sleep -Seconds 30
                       $Retrycount = $Retrycount + 1
                      }
                    }
             }
         While ($Stoploop -eq $false)
}
```

A continuación, puede pasar cada llamada remota a la función como

`ResilientRemoteCall { Get-AzVm }` o

`ResilientRemoteCall { $searchAPIKey = (Invoke-AzureRmResourceAction -Action listAdminKeys -ResourceId $resource.ResourceId -ApiVersion 2015-08-19 -Force).PrimaryKey}`

Trabajo con varias suscripciones

El runbook debe poder trabajar con suscripciones. Por ejemplo, para controlar varias suscripciones, el runbook debe usar el cmdlet Disable-AzContextAutosave. Este cmdlet garantiza que no se recupere el contexto de autenticación desde otro runbook que se ejecute en el mismo espacio aislado.

PowerShell

```
# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process

# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription `
    -DefaultProfile $AzureContext

$childRunbookName = 'childRunbookDemo'
$resourceGroupName = "resourceGroupName"
$automationAccountName = "automationAccountName"

$startParams = @{
    ResourceGroupName     = $resourceGroupName
    AutomationAccountName = $automationAccountName
    Name                  = $childRunbookName
    DefaultProfile        = $AzureContext
}
Start-AzAutomationRunbook @startParams
```

Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:

1. En la línea 5, quite `$AzureContext = (Connect-AzAccount -Identity).context`,
2. Sustitúyalo por `$AzureContext = (Connect-AzAccount -Identity -AccountId <ClientId>).context`, y
3. Escriba el id. de cliente.

Trabajo con un script personalizado

> ⓘ **Nota:**
>
> Normalmente no se pueden ejecutar scripts y runbooks personalizados en el host con un agente de Log Analytics instalado.

Para utilizar un script personalizado, haga lo siguiente:

1. Crear una cuenta de Automation.
2. Implemente el rol Hybrid Runbook Worker.

Prueba de un runbook

Al probar un runbook, se ejecuta la versión de borrador y se completará cualquier acción que este realice. No se crea ningún historial de trabajos, pero los flujos salida y advertencia y error se muestran en el panel de **salida de la prueba**. Los mensajes del flujo detallado solo se muestran en el panel de salida si la variable VerbosePreference está establecida en `Continue`.

Aunque se esté ejecutando la versión de borrador, el runbook sigue ejecutándose con normalidad y realiza cualquier acción sobre los recursos del entorno. Por este motivo, solo debe probar runbooks en recursos no pertenecientes a entornos de producción.

> ⓘ **Nota:**
>
> Todas las acciones de ejecución de runbooks se registran en el **registro de actividad** de la cuenta de Automation con el nombre de la operación **Crear un trabajo de Azure Automation**. Sin embargo, la ejecución de runbooks en un panel de prueba donde se ejecuta la versión de borrador del runbook se registraría en los registros de actividad con el nombre de la operación **Escribir un borrador de runbook de Azure Automation**. Seleccione las pestañas **Operación** y **JSON** para ver el ámbito que termina en .. */runbooks/(nombre de runbook)/draft/testjob*.

El procedimiento para probar cada tipo de runbook es el mismo. No hay diferencias entre realizar las pruebas en el editor de texto o en el editor gráfico de Azure Portal.

1. Abra la versión de borrador del runbook en el editor de texto o el editor gráfico.
2. Haga clic en **Prueba** para abrir la página de **prueba**.

3. Si el runbook tiene parámetros, se enumeran en el panel izquierdo, donde puede proporcionar los valores que se usarán para la prueba.
4. Si quiere ejecutar la prueba en una instancia de Hybrid Runbook Worker, cambie los **Parámetros de ejecución** a **Hybrid Worker** y seleccione el nombre del grupo de destino. De lo contrario, mantenga el valor predeterminado **Azure** para ejecutar la prueba en la nube.
5. Haga clic en **Iniciar** para iniciar la prueba.
6. Puede usar los botones en el panel de **salida** para detener o suspender un runbook de flujo de trabajo de PowerShell o gráfico mientras se está probando. Cuando se suspende el runbook, este completa la actividad que está realizando en ese momento antes de suspenderse. Una vez suspendido, puede detener o reiniciar el runbook.
7. Inspeccione la salida del runbook en el panel **Output**.

Publicar un manual de procedimientos

Cuando crea o importa un nuevo runbook, debe publicarlo antes de poder ejecutarlo. Cada runbook de Azure Automation tiene una versión de borrador y una versión publicada. Solo es posible ejecutar la versión publicada y solo es posible editar la versión de borrador. Los cambios realizados en la versión de borrador no afectan la versión publicada. Cuando la versión en borrador deba ponerse a disposición, la publica, sobrescribiendo la versión publicada actual con la versión en borrador.

Publicar un runbook en el portal de Azure

1. En el portal de Azure, busque y seleccione **Cuentas de automatización**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. Abra el runbook en la cuenta de Automation.
4. Haga clic en **Editar**.
5. Haga clic en el botón **Publicar** y, a continuación, seleccione **Sí** en el mensaje de comprobación.

Publicación de un runbook mediante PowerShell

Utilice el cmdlet Publish-AzAutomationRunbook para publicar su runbook.

Azure PowerShell

```
$accountName = "MyAutomationAccount"
$runbookName = "Sample_TestRunbook"
$rgName = "MyResourceGroup"
```

```
$publishParams = @{
    AutomationAccountName = $accountName
    ResourceGroupName     = $rgName
    Name                  = $runbookName
}
Publish-AzAutomationRunbook @publishParams
```

Programación de un runbook en Azure Portal

Una vez publicado el runbook, puede programarlo para que funcione:

1. En el portal de Azure, busque y seleccione **Cuentas de automatización**.
2. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
3. Seleccione el runbook de la lista de runbooks.
4. Seleccione **Programas** en **Recursos**.
5. Seleccione **Agregar una programación**.
6. En el panel **Programar runbook**, seleccione **Vincular una programación al runbook**.
7. En el panel **Programación**, seleccione **Crear una nueva programación**.
8. Escriba un nombre, una descripción y otros parámetros en el panel **Nueva programación**.
9. Una vez creada la programación, resáltela y haga clic en **Aceptar**. Ahora debería estar vinculada a su runbook.
10. Busque un correo electrónico en el buzón que le informa sobre el estado del runbook.

Restauración del runbook eliminado

Puede recuperar un runbook eliminado mediante scripts de PowerShell. Para recuperar un runbook, asegúrese de que se cumplan las condiciones siguientes:

- Los runbooks que se van a restaurar se eliminaron en los últimos 29 días.
- La cuenta de Automation para ese runbook existe.
- El permiso de rol *colaborador de Automation* se concede a la identidad administrada asignada por el sistema de la cuenta de Automation.

Script de PowerShell

- Ejecute el script de PowerShell como un trabajo en la cuenta de Automation para restaurar los runbooks eliminados.
- Descargue el script de PowerShell desde GitHub. Como alternativa, puede importar el script de PowerShell llamado *Restore Automation runbook* desde la Galería de runbooks.

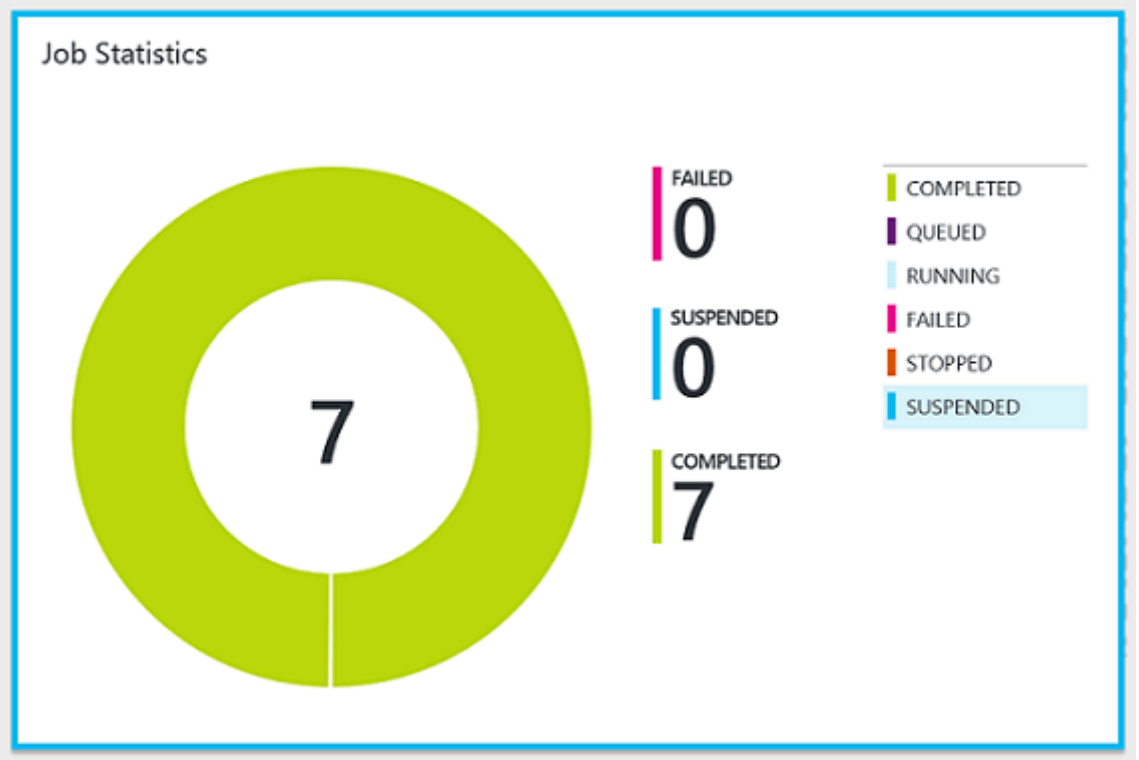
Proporcione el nombre del runbook que se va a restaurar y ejecútelo como un trabajo en Azure Automation para restaurar los runbooks eliminados.

- Descargue el script desde GitHub o importe el script de PowerShell llamado *List Deleted Automation Runbook* desde la Galería de runbooks para identificar los nombres de los runbooks eliminados en los últimos 29 días.

Obtención de estados del trabajo

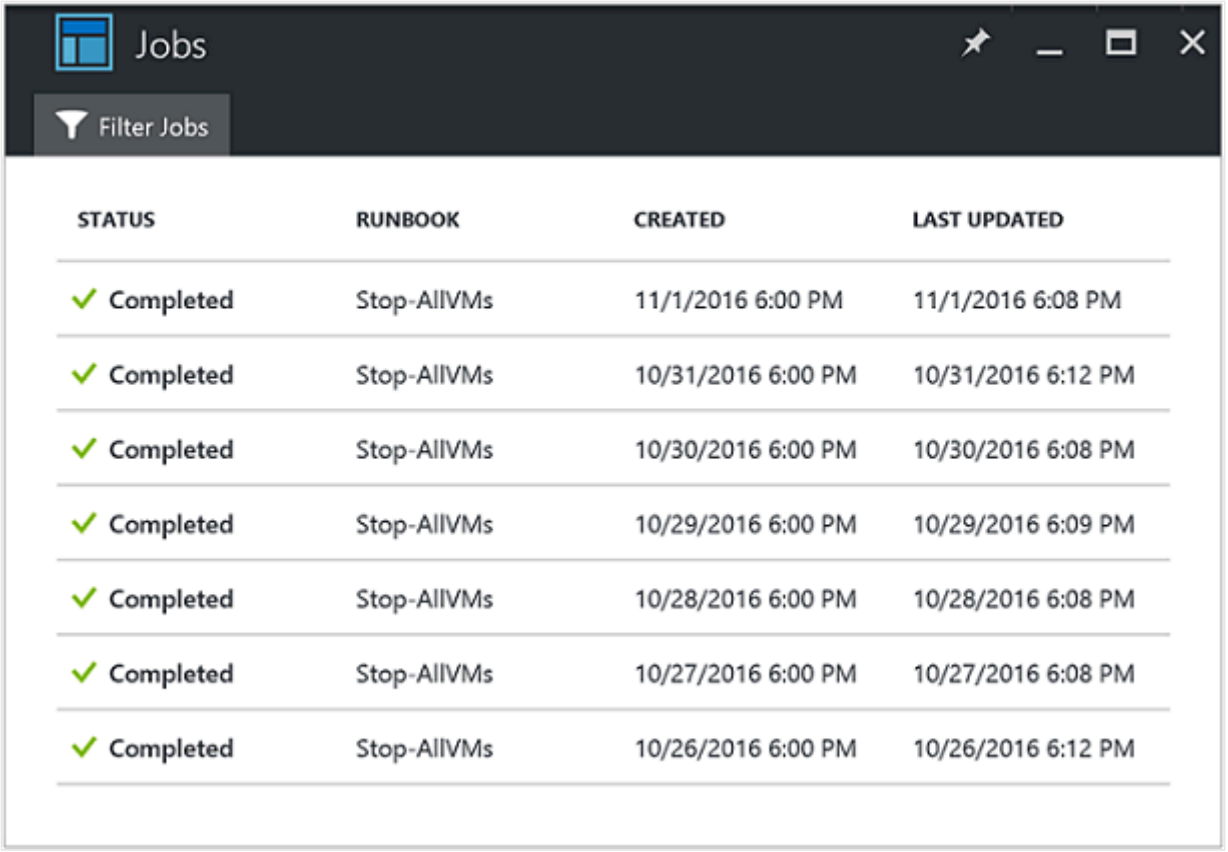
Visualización de los estados en Azure Portal

En Azure Automation, los detalles del control de trabajos se proporcionan en Trabajos. Cuando esté listo para ver los trabajos de runbook, use Azure Portal y acceda a su cuenta de Automation. A la derecha, se puede ver un resumen de todos los trabajos de runbook en **Estadísticas de trabajo**.

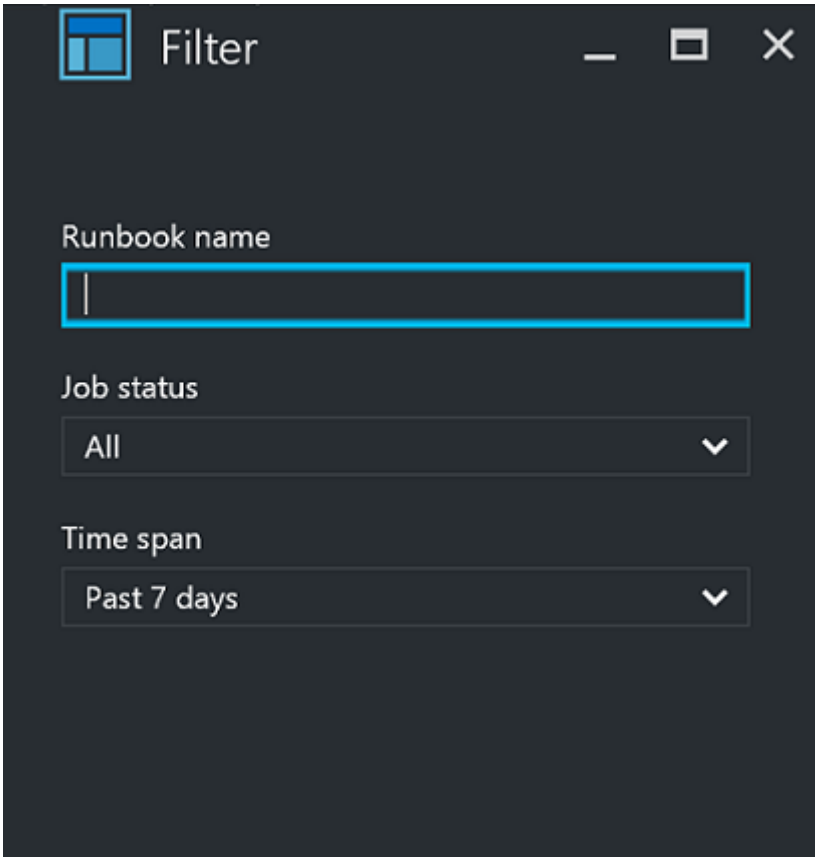


Este resumen muestra un contador y una representación gráfica del estado de trabajo de cada trabajo ejecutado.

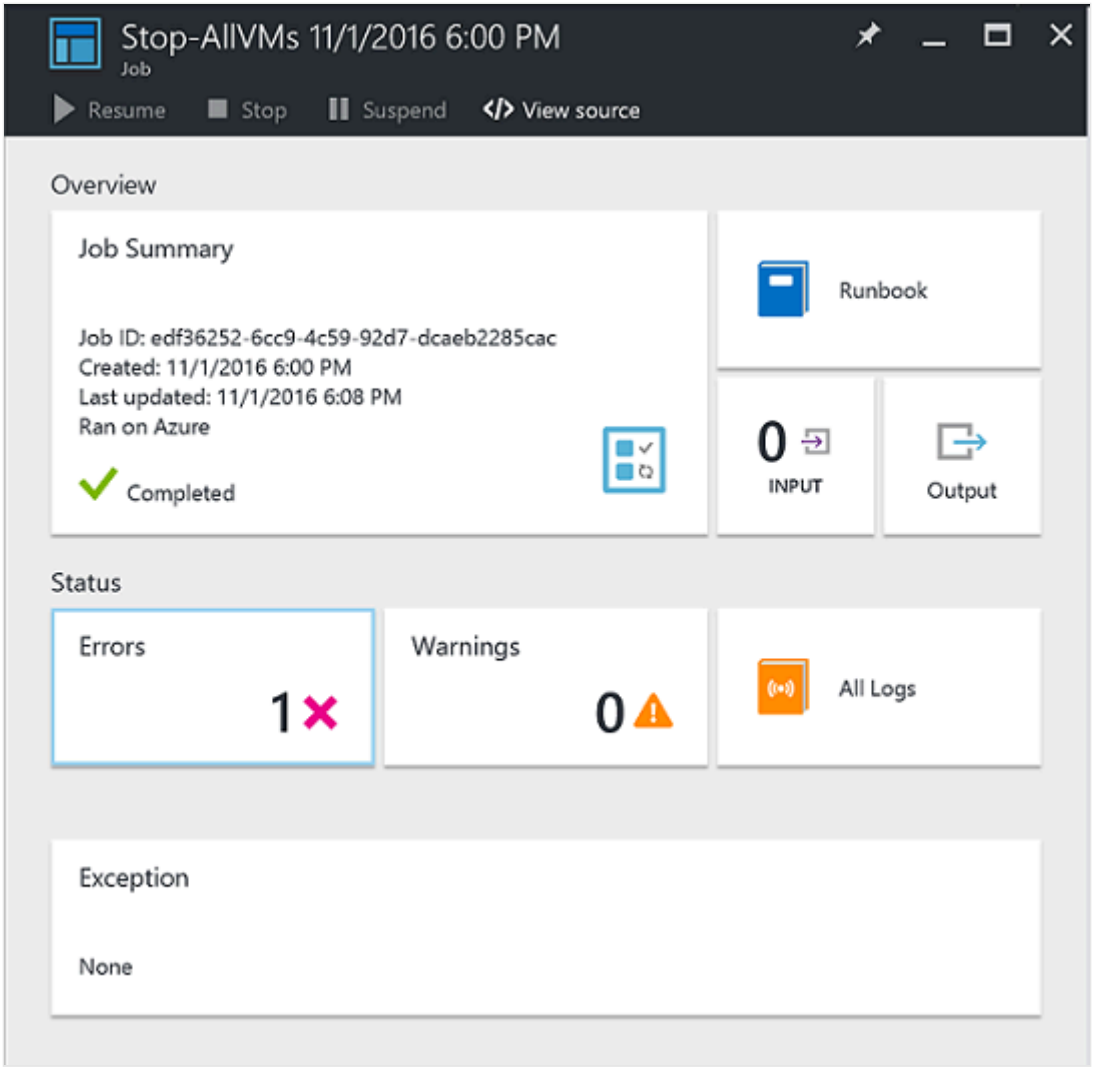
Al hacer clic en el icono aparece la página Trabajos, que incluye una lista resumida de todos los trabajos ejecutados. En esta página se muestra el estado, el nombre del runbook, la hora de inicio y la hora de finalización de cada trabajo.



Para filtrar la lista de trabajos, puede seleccionar **Filtrar trabajos**. Filtre por un runbook específico, por el estado del trabajo o por una opción de la lista desplegable, y proporcione el intervalo de tiempo para la búsqueda.



Como alternativa, puede ver los detalles resumidos de los trabajos de un runbook concreto; para ello, selecciónelo en la página Runbooks de la cuenta de Automation y, después, seleccione **Trabajos**. Esta acción muestra la página Trabajos. Desde aquí, puede hacer clic en un registro del trabajo para ver sus detalles y resultados.



Recuperación de los estados del trabajo con PowerShell

Puede utilizar el cmdlet Get-AzAutomationJob para recuperar los trabajos creados para un runbook y los detalles de un trabajo determinado. Si inicia un runbook mediante `Start-AzAutomationRunbook`, se devuelve el trabajo resultante. Utilice Get-AzAutomationJobOutput para recuperar la salida de un trabajo.

En el ejemplo siguiente se recupera el último trabajo para un runbook de ejemplo y se muestra su estado, los valores proporcionados para los parámetros del runbook y la salida del trabajo.

Azure PowerShell

```
$getJobParams = @{
    AutomationAccountName = 'MyAutomationAccount'
    ResourceGroupName     = 'MyResourceGroup'
```

```
    Runbookname           = 'Test-Runbook'
}
$job = (Get-AzAutomationJob @getJobParams | Sort-Object LastModifiedDate -Desc)[0]
$job | Select-Object JobId, Status, JobParameters

$getOutputParams = @{
    AutomationAccountName = 'MyAutomationAccount'
    ResourceGroupName     = 'MyResourceGroup'
    Id                    = $job.JobId
    Stream                = 'Output'
}
Get-AzAutomationJobOutput @getOutputParams
```

En el ejemplo siguiente se recupera la salida de un trabajo específico y se devuelve cada registro. Si hay una excepción para uno de los registros, el script escribe la excepción en lugar del valor. Este comportamiento resulta útil, ya que las excepciones pueden proporcionar información adicional que podría no registrarse normalmente durante la salida.

Azure PowerShell

```
$params = @{
    AutomationAccountName = 'MyAutomationAccount'
    ResourceGroupName     = 'MyResourceGroup'
    Stream                = 'Any'
}
$output = Get-AzAutomationJobOutput @params

foreach ($item in $output) {
    $jobOutParams = @{
        AutomationAccountName = 'MyAutomationAccount'
        ResourceGroupName     = 'MyResourceGroup'
        Id                    = $item.StreamRecordId
    }
    $fullRecord = Get-AzAutomationJobOutputRecord @jobOutParams

    if ($fullRecord.Type -eq 'Error') {
        $fullRecord.Value.Exception
    } else {
        $fullRecord.Value
    }
}
```

Pasos siguientes

- Para ver consultas de ejemplo, consulte Consultas de ejemplo para registros de trabajos y flujos de trabajo
- Para obtener información detallada sobre la administración de un runbook, vea Ejecución de un runbook en Azure Automation.

- Para obtener información sobre la compatibilidad con las versiones de tiempo de ejecución y su retirada al administrar runbooks, consulte Política de compatibilidad y retirada de entornos de ejecución de lenguajes para Azure Automation.
- Para preparar un runbook de PowerShell, vea Edición de runbooks de texto en Azure Automation.
- Para solucionar problemas con la ejecución de un runbook, consulte Solucionar problemas del runbook.

Last updated on 29/06/2026

Inicio de un runbook en Azure Automation

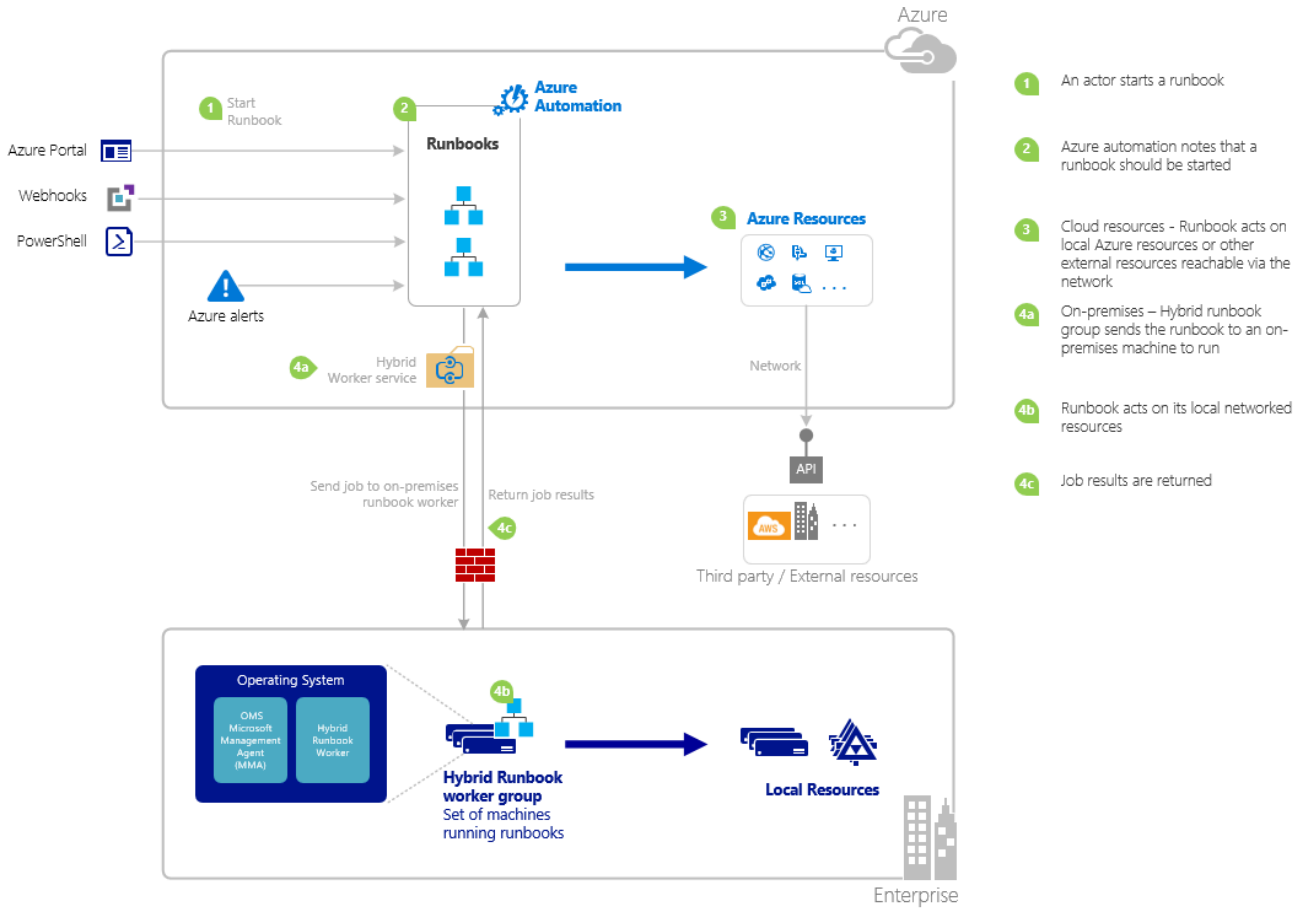
La tabla siguiente le ayuda a determinar el método para iniciar un runbook en Azure Automation que sea más adecuado para su escenario concreto. Este artículo incluye detalles acerca de cómo iniciar un runbook con Azure Portal y Windows PowerShell. En otra documentación a la que puede acceder desde los siguientes vínculos se proporcionan detalles sobre los otros métodos.

Expandir tabla

Método	Características
Azure Portal	• Método más sencillo con la interfaz de usuario interactiva. • Forma para proporcionar valores de parámetro simples. • Seguimiento sencillo del estado del trabajo. • Acceso autenticado con inicio de sesión de Azure.
Windows PowerShell	• Permite llamar desde la línea de comandos con los cmdlets de Windows PowerShell. • Puede incluirse en una característica automatizada con varios pasos. • La solicitud se autentica con el certificado o el usuario de OAuth principal/servicio principal. • Permite proporcionar valores de parámetro simples y complejos. • Realizar el seguimiento del estado del trabajo. • Responder a la necesidad del cliente para admitir los cmdlets de PowerShell.
API de Azure Automation	• El método más flexible, pero también más complejo. • Permite llamar desde cualquier código personalizado que pueda realizar solicitudes HTTP. • Solicite autenticación con certificado, entidad de servicio o entidad de seguridad de usuario de OAuth. • Permite proporcionar valores de parámetro simples y complejos. *Si va a llamar a un runbook de Python mediante la API, la carga de JSON se tiene que serializar.* • Realizar el seguimiento del estado del trabajo.
Webhooks	• Permite iniciar el runbook desde una única solicitud HTTP. • Autenticado con el token de seguridad de la dirección URL. • El cliente no puede reemplazar los valores de parámetro especificados al crear webhook. El runbook puede definir un único parámetro que se rellena con los detalles de la solicitud HTTP. • No tiene capacidad de realizar un seguimiento del estado del trabajo a través de la dirección URL de webhook.
Respuesta a una alerta de Azure	• Permite iniciar un runbook en respuesta a una alerta de Azure. • Permite configurar webhook para el runbook y vincular a la alerta. • Autenticado con el token de seguridad de la dirección URL.
Programación	• Permite iniciar automáticamente el runbook en una programación según la hora, diaria, semanal o mensual.

Método	Características
	• Permite manipular la programación a través de Azure Portal, los cmdlets de PowerShell o la API de Azure. • Permite proporcionar valores de parámetro que se usarán con la programación.
Desde otro Runbook	• Permite usar un runbook como una actividad en otro runbook. • Es útil para la funcionalidad utilizada por varios runbooks. • Permite proporcionar valores de parámetro a un runbook secundario y usar la salida de un runbook principal.

La siguiente imagen ilustra el proceso paso a paso detallado en el ciclo de vida de un runbook. Incluye las distintas formas de iniciar un runbook en Azure Automation, los componentes requeridos para que Hybrid Runbook Worker ejecute runbooks de Azure Automation y las interacciones entre los distintos componentes. Para obtener información sobre cómo ejecutar runbooks de Automation en su centro de datos, consulte Trabajos híbridos de runbook



Trabajar con parámetros del runbook

Cuando se inicia un runbook desde Azure Portal o Windows PowerShell, la instrucción se envía a través del servicio web Azure Automation. Este servicio no admite parámetros con tipos de datos complejos. Si necesita proporcionar un valor para un parámetro complejo, se debe

llamar en línea desde otro runbook, tal como se describe en Runbooks secundarios en Azure Automation.

El servicio web de Azure Automation proporciona una funcionalidad especial para parámetros que usan ciertos tipos de datos, como se describe en las secciones siguientes.

Valores con nombre

Si el parámetro tiene el tipo de datos [object], puede usar el siguiente formato JSON para enviar una lista de valores con nombre: *{"Nombre1":Valor1, "Nombre2":Valor2, "Nombre3":Valor3}*. Estos valores deben ser tipos simples. El runbook recibe el parámetro como un elemento PSCustomObject con propiedades que corresponden a cada valor con nombre.

Considere el siguiente runbook de prueba que acepta un parámetro denominado user.

PowerShell

```
Workflow Test-Parameters
{
   param (
      [Parameter(Mandatory=$true)][object]$user
   )
    $userObject = $user | ConvertFrom-JSON
    if ($userObject.Show) {
        foreach ($i in 1..$userObject.RepeatCount) {
            $userObject.FirstName
            $userObject.LastName
        }
    }
}
```

El texto siguiente podría utilizarse para el parámetro de usuario.

JSON

```
{FirstName:'Joe',LastName:'Smith',RepeatCount:'2',Show:'True'}
```

Esta acción devuelve la siguiente salida:

Resultados

```
Joe
Smith
Joe
Smith
```

Matrices

Si el parámetro es una matriz como [array] o [string], puede usar entonces el siguiente formato JSON para enviarle una lista de valores: *[Value1,Value2,Value3]*. Estos valores deben ser tipos simples.

Considere el siguiente runbook de prueba que acepta un parámetro denominado *user*.

PowerShell

```
Workflow Test-Parameters
{
   param (
      [Parameter(Mandatory=$true)][array]$user
   )
    if ($user[3]) {
        foreach ($i in 1..$user[2]) {
            $ user[0]
            $ user[1]
        }
    }
}
```

El texto siguiente podría utilizarse para el parámetro de usuario.

input

```
["Joe","Smith",2,true]
```

Esta acción devuelve la siguiente salida:

Resultados

```
Joe
Smith
Joe
Smith
```

Credenciales

Si el parámetro es de tipo de datos `PSCredential`, puede proporcionar el nombre de un recurso de credenciales de Azure Automation. El runbook recupera la credencial con el nombre que especifique. El runbook de prueba siguiente acepta un parámetro denominado `credential`.

PowerShell

```
Workflow Test-Parameters
{
   param (
      [Parameter(Mandatory=$true)][PSCredential]$credential
   )
   $credential.UserName
}
```

El texto siguiente se puede utilizar para el parámetro user suponiendo la existencia previa de un recurso de credenciales denominado `My Credential`.

input

```
My Credential
```

Suponiendo que el nombre de usuario de la credencial es `jsmith`, se muestra la salida siguiente.

Resultados

```
jsmith
```

Inicio de un runbook con Azure Portal

1. En Azure Portal, seleccione **Automation** y después elija el nombre de una cuenta de Automation.
2. En el panel izquierdo, seleccione **Runbooks**.
3. En la página **Runbooks**, seleccione un runbook y haga clic en **Iniciar**.
4. Si el runbook tiene parámetros, se le pide que proporcione los valores de cada parámetro con un cuadro de texto. Para más información sobre los parámetros, consulte Parámetros de runbook.
5. En el panel **Trabajo**, puede ver el estado del trabajo de runbook.

Inicio de un runbook con PowerShell

Puede utilizar Start-AzureRmAutomationRunbook para iniciar un runbook con Windows PowerShell. El código de ejemplo siguiente inicia un runbook denominado **Test-Runbook**.

Azure PowerShell

```
Start-AzAutomationRunbook -AutomationAccountName "MyAutomationAccount" -Name "Test-Runbook" -ResourceGroupName "ResourceGroup01"
```

`Start-AzAutomationRunbook` devuelve un objeto de trabajo que puede usar para realizar un seguimiento del estado una vez que se inicia el runbook. A continuación, puede usar este objeto de trabajo con Get-AzureAutomationJob para determinar el estado del trabajo y Get-AzureAutomationJobOutput para obtener su salida. El ejemplo siguiente inicia un runbook denominado **Test-Runbook**, espera hasta que se ha completado y después muestra la salida.

Azure PowerShell

```
$runbookName = "Test-Runbook"
$ResourceGroup = "ResourceGroup01"
$AutomationAcct = "MyAutomationAccount"

$job = Start-AzAutomationRunbook -AutomationAccountName $AutomationAcct -Name
$runbookName -ResourceGroupName $ResourceGroup

$doLoop = $true
While ($doLoop) {
   $job = Get-AzAutomationJob -AutomationAccountName $AutomationAcct -Id $job.JobId
-ResourceGroupName $ResourceGroup
   $status = $job.Status
   $doLoop = (($status -ne "Completed") -and ($status -ne "Failed") -and ($status -
ne "Suspended") -and ($status -ne "Stopped"))
}

Get-AzAutomationJobOutput -AutomationAccountName $AutomationAcct -Id $job.JobId -
ResourceGroupName $ResourceGroup -Stream Output
```

Si el runbook requiere parámetros, tiene que proporcionarlos como una tabla hash. La clave de la tabla hash tiene que coincidir con el nombre del parámetro y el valor es el valor del parámetro. En el ejemplo siguiente se muestra cómo iniciar un runbook con dos parámetros de cadena denominados FirstName y LastName, un número entero denominado RepeatCount y un parámetro booleano denominado Show. Para más información sobre los parámetros, consulte Parámetros de runbook.

Azure PowerShell

```
$params = @{"FirstName"="Joe";"LastName"="Smith";"RepeatCount"=2;"Show"=$true}
Start-AzAutomationRunbook -AutomationAccountName "MyAutomationAccount" -Name "Test-
Runbook" -ResourceGroupName "ResourceGroup01" -Parameters $params
```

Pasos siguientes

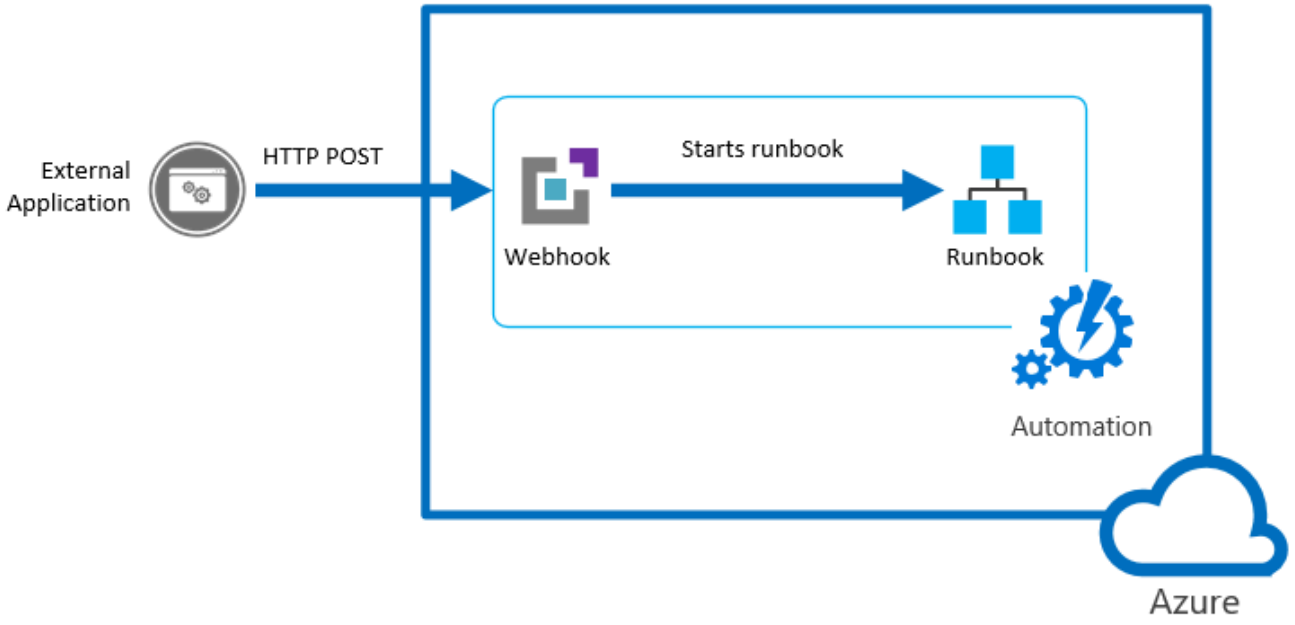
- Para obtener información detallada sobre la administración de runbooks, consulte Administración de runbooks en Azure Automation.
- Para obtener información sobre PowerShell, vea Documentación de PowerShell.

- Para solucionar problemas con la ejecución de un runbook, vea Solución de problemas relativos a errores con runbooks.

Last updated on 17/11/2025

Inicio de un runbook desde un webhook

Un webhook permite a un servicio externo iniciar un runbook determinado en Azure Automation a través de una única solicitud HTTP. Los servicios externos incluyen Azure DevOps Services, GitHub, registros de Azure Monitor y aplicaciones personalizadas. Este servicio puede usar un webhook para iniciar un runbook sin implementar la API de Azure Automation al completo. Puede comparar webhooks con otros métodos para iniciar un runbook en Iniciar un runbook en Azure Automation.



Para comprender los requisitos de cliente de TLS 1.2 o posteriores con webhooks, consulte TLS para Azure Automation.

Propiedades de un webhook

En la tabla siguiente se describen las propiedades que debe configurar para un webhook.

Expandir tabla

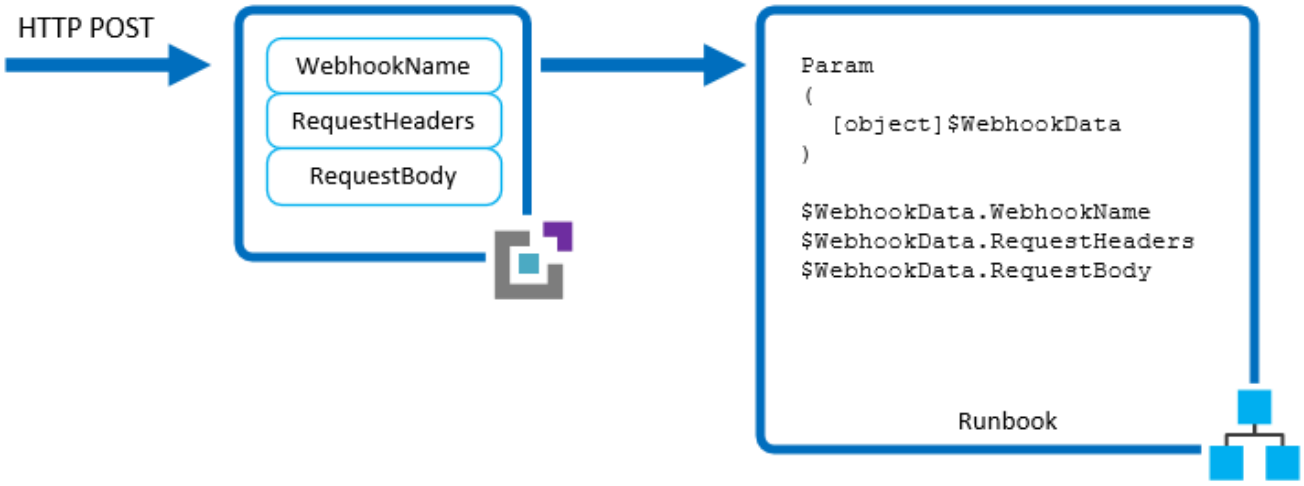
Propiedad	Descripción
Nombre	Nombre del webhook. Puede indicar cualquier nombre que desee, ya que no se expone al cliente. Solo se utiliza para que usted identifique el runbook en Azure Automation. Como práctica recomendada, debe proporcionar al webhook un nombre relacionado con el cliente que lo usa.

Propiedad	Descripción
URL	Dirección URL del webhook. Es la dirección única a la que llama un cliente con una solicitud HTTP POST para iniciar el runbook vinculado al webhook. Se genera automáticamente al crear el webhook. No se puede especificar una dirección URL personalizada. La dirección URL contiene un token de seguridad que permite que un sistema de terceros invoque el runbook sin autenticación adicional, por lo que debe tratarla como una contraseña. Por motivos de seguridad, solo puede ver la dirección URL en el portal de Azure al crear el webhook. Anote la dirección URL en una ubicación segura para su uso futuro.
Fecha de expiración	Fecha de expiración del webhook, tras la cual ya no se puede usar. Esta fecha se puede modificar una vez creado el webhook, siempre y cuando no haya expirado.
habilitado	Este valor indica si el webhook se habilita de forma predeterminada cuando se crea. Si establece esta propiedad en Deshabilitado, ningún cliente puede usar el webhook. Puede establecer esta propiedad al crear el webhook o en cualquier otro momento después de su creación.

Parámetros usados cuando el webhook inicia un runbook

Un webhook puede definir valores para parámetros de runbook que se usan cuando se inicia el runbook. El webhook debe incluir los valores de los parámetros obligatorios del runbook y puede incluir valores para los parámetros opcionales. Los valores de parámetro configurados en un webhook se pueden modificar incluso después de crear el webhook. Varios webhooks vinculados a un único runbook puede utilizar distintos valores de parámetros de runbook. Cuando un cliente inicia un runbook mediante un webhook, no puede reemplazar los valores de los parámetros definidos en el webhook.

Para recibir datos del cliente, el runbook admite un único parámetro llamado `WebhookData`. Este parámetro define un objeto que contiene datos que el cliente incluye en una solicitud POST.



El parámetro `WebhookData` tiene las siguientes propiedades:

Expandir tabla

Propiedad	Descripción
WebhookName	Nombre del webhook.
RequestHeader	PSCustomObject que contiene los encabezados de la solicitud POST entrante.
RequestBody	Cuerpo de la solicitud POST entrante. Este cuerpo conserva el formato de los datos, como cadena, JSON, XML o datos codificados por formulario. El runbook debe escribirse para que trabaje con el formato de datos que se espera.

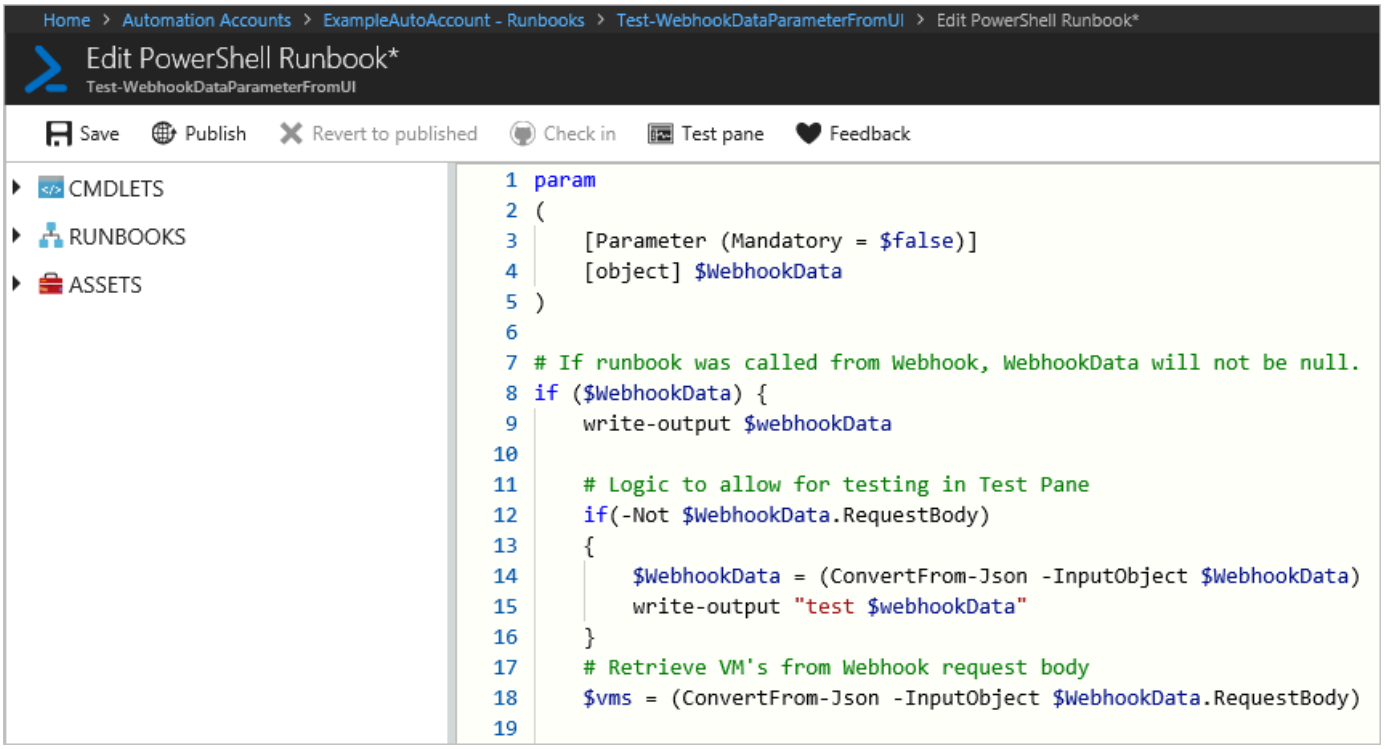
No hay ninguna configuración del webhook obligatoria para admitir el parámetro `WebhookData`, y el runbook no tiene que aceptarlo. Si el runbook no define el parámetro, se ignoran los detalles de la solicitud enviada desde el cliente.

> **Nota:**
>
> Al llamar a un webhook, el cliente siempre debe almacenar los valores de los parámetros, por si se produce un error en la llamada. Si se produce un problema de conexión o de interrupción de la red, la aplicación no podrá recuperar las llamadas de webhook con errores.

Si especifica un valor para `WebhookData` durante la creación del webhook, se invalidará cuando el webhook inicie el runbook con los datos de la solicitud POST del cliente. Esto sucederá incluso si la aplicación no contiene ningún dato en el cuerpo de la solicitud.

Si inicia un runbook que define `WebhookData` mediante un mecanismo que no sea un webhook, puede proporcionar un valor para `WebhookData` que el runbook reconozca. Este valor debe ser un objeto con las mismas propiedades que el parámetro `WebhookData`, para que el runbook pueda funcionar correctamente con él, del mismo modo que si trabajara con objetos `WebhookData` reales pasados por un webhook.

Por ejemplo, si va a iniciar el siguiente libro de ejecución desde el portal de Azure y desea pasar datos de ejemplo de un webhook para probar, debe pasar los datos en formato JSON en la interfaz de usuario.



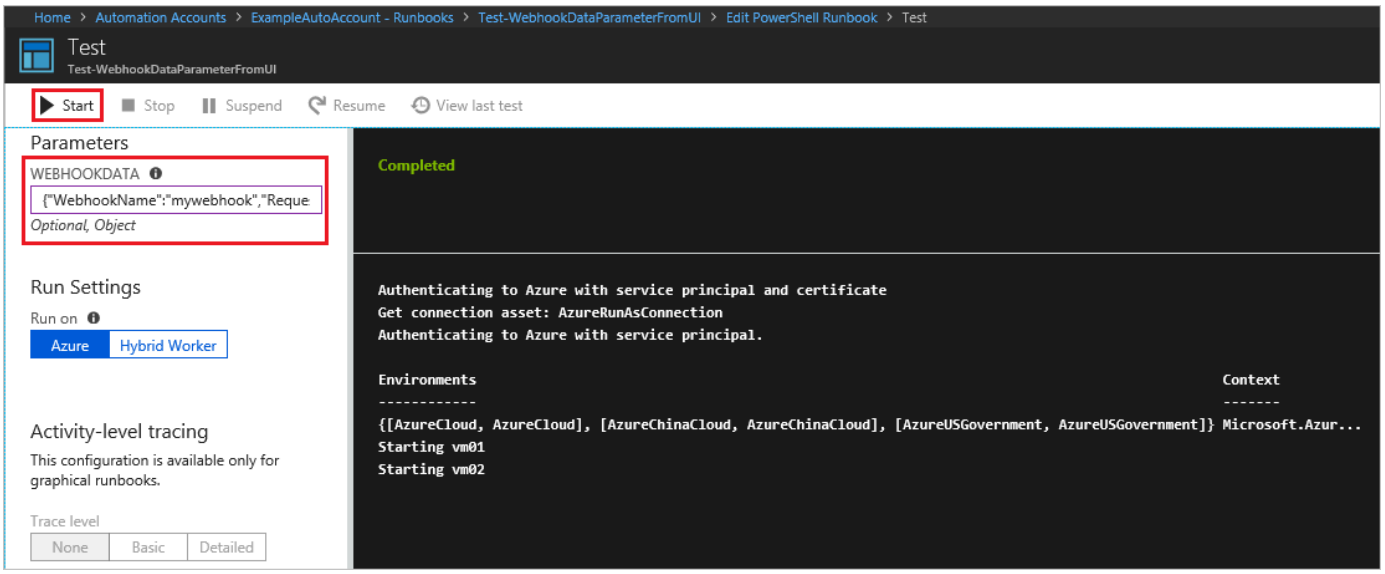
En el siguiente ejemplo de runbook, vamos a definir las siguientes propiedades para `WebhookData`:

- **WebhookName**: MyWebhook
- **RequestBody**: `*[{'ResourceGroup': 'myResourceGroup','Name': 'vm01'},{'ResourceGroup': 'myResourceGroup','Name': 'vm02'}]*`

Ahora pasamos el siguiente objeto JSON en la interfaz de usuario para el parámetro `WebhookData`. Este ejemplo, con los retornos de carro y los caracteres de nueva línea, coincide con el formato que se pasa desde un webhook.

JSON

```
{"WebhookName":"mywebhook","RequestBody":"[\r\n {\r\n \"ResourceGroup\": \"vm01\",\r\n \"Name\": \"vm01\"\r\n },\r\n {\r\n \"ResourceGroup\": \"vm02\",\r\n \"Name\": \"vm02\"\r\n }\r\n]"}
```



> **Nota:**
>
> Azure Automation registra los valores de todos los parámetros de entrada junto con el trabajo del runbook. Por tanto, se registra cualquier entrada que proporcione el cliente en la solicitud de webhook, y esta entrada estará disponible para cualquiera con acceso al trabajo de automatización. Por este motivo, debe tener cuidado en cómo incluir información confidencial en las llamadas de webhook.

Seguridad del webhook

La seguridad de un webhook se basa en la privacidad de su dirección URL, que contiene un token de seguridad que permite que se invoque el webhook. Azure Automation no realiza ninguna autenticación en una solicitud siempre que se realice en la dirección URL correcta. Por esta razón, los clientes no deben utilizar webhooks en runbooks que realicen operaciones altamente confidenciales sin utilizar medios alternativos para validar la solicitud.

Considere las estrategias siguientes:

- Puede incluir lógica dentro de un runbook para determinar si se le llama desde un webhook. Establezca que el runbook compruebe la propiedad `WebhookName` del parámetro `WebhookData`. El runbook puede realizar más validaciones buscando información determinada en las propiedades `RequestHeader` y `RequestBody`.
- Haga que el runbook realice alguna validación de una condición externa cuando recibe una solicitud de webhook. Por ejemplo, considere un runbook al que llama GitHub cada vez que haya un nuevo commit en un repositorio de GitHub. El runbook podría conectarse a GitHub para validar que ha ocurrido un nuevo commit antes de continuar.

- Azure Automation admite etiquetas de servicio de red virtual Azure, específicamente GuestAndHybridManagement. Puede usar etiquetas de servicio para definir controles de acceso de red en grupos de seguridad de red o Azure Firewall y desencadenar webhooks desde dentro de la red virtual. Las etiquetas de servicio se pueden usar en lugar de direcciones IP específicas al crear reglas de seguridad. Al especificar el nombre de la etiqueta de servicio **GuestAndHybridManagement** en el campo de origen o destino apropiado de una regla, puede permitir o denegar el tráfico del servicio Automation. Esta etiqueta de servicio no admite un control más pormenorizado mediante la restricción de los intervalos de IP a una región específica.

Creación de un webhook

> ⓘ **Nota:**
>
> - Al iniciar el runbook de PowerShell 7 mediante el webhook, convierte automáticamente el parámetro de entrada del webhook en un JSON no válido. Para obtener más información, consulte Problemas conocidos - PowerShell7.1 (vista preliminar). Se recomienda usar el webhook con el runbook de PowerShell 5.
> - No se admite el uso de un webhook para iniciar un runbook de Python.

Para crear un webhook, siga estos pasos:

1. Cree un runbook de PowerShell con el código siguiente:

PowerShell

```
param
(
    [Parameter(Mandatory=$false)]
    [object] $WebhookData
)

write-output "start"
write-output ("object type: {0}" -f $WebhookData.gettype())
write-output $WebhookData
write-output "`n`n"
write-output $WebhookData.WebhookName
write-output $WebhookData.RequestBody
write-output $WebhookData.RequestHeader
write-output "end"

if ($WebhookData.RequestBody) {
    $names = (ConvertFrom-Json -InputObject $WebhookData.RequestBody)
```

```
        foreach ($x in $names)
        {
            $name = $x.Name
            Write-Output "Hello $name"
        }
}
else {
    Write-Output "Hello World!"
}
```

2. Cree un webhook mediante el portal de Azure o PowerShell o la API REST. Un webhook requiere un runbook publicado. En este recorrido se emplea una versión modificada del runbook creado a partir de Crear un runbook de Azure Automation.

Azure Portal

Para crear un webhook mediante el portal de Azure, haga lo siguiente:

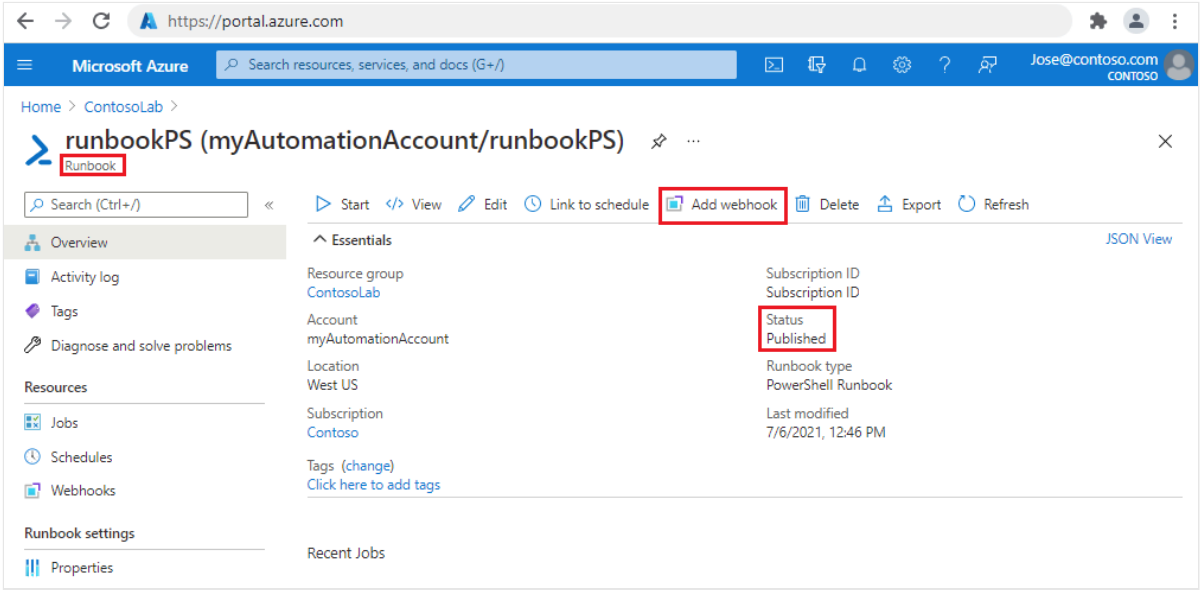
a. Inicie sesión en el portal Azure.

b. En el portal de Azure, vaya a la cuenta de Automation.

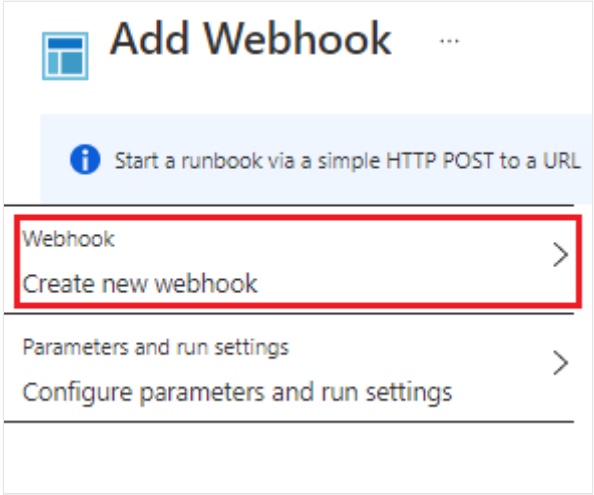
c. En **Automatización de procesos**, seleccione **Runbooks** para abrir la página de **runbooks**.

d. Seleccione el runbook en la lista para abrir la página **Información general** del runbook.

e. Seleccione **Agregar Webhook** para abrir la página **Agregar Webhook**.

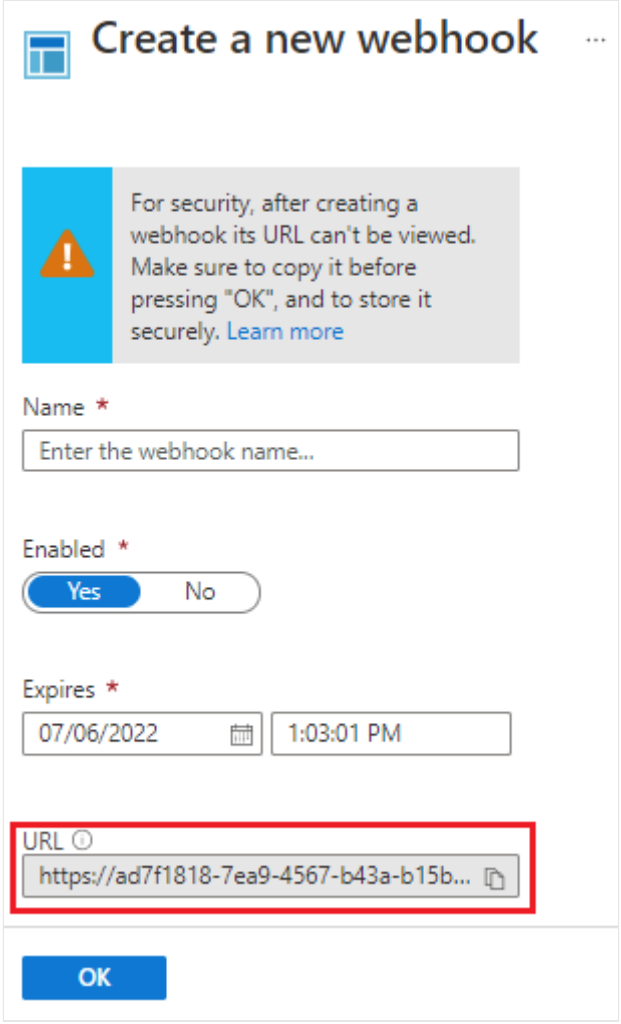


f. En la página **Agregar Webhook**, seleccione **Crear nuevo Webhook**.



g. Escriba el **Nombre** del webhook. La fecha de expiración del campo **Expira** tiene como valor predeterminado un año a partir de la fecha actual.

h. Seleccione el icono de copia o presione `Ctrl + C` copie la dirección URL del webhook. A continuación, guarde la dirección URL en una ubicación segura.

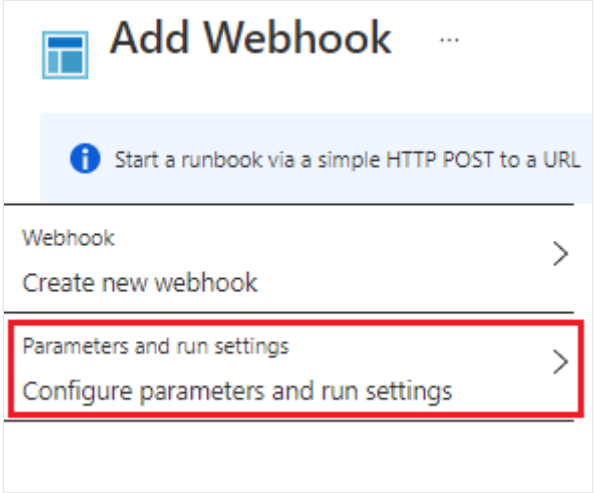


> ⓘ **Importante**

> Una vez creado el webhook, ya no puede volver a recuperar la dirección URL. Asegúrese de copiarlo y grabarlo como se indicó anteriormente.

i. Seleccione **Aceptar** para volver a la página **Agregar Webhook**.

j. En la página **Agregar Webhook**, seleccione **Configurar parámetros y ejecutar configuraciones** para abrir la página **Parámetros**.



k. Revise la página **Parámetros**. Para el runbook de ejemplo que se usa en este artículo, no se necesita ningún cambio. Seleccione **Aceptar** para volver a la página **Agregar Webhook**.

l. En la página **Agregar Webhook**, seleccione **Crear**. Se crea el webhook y se le devuelve a la página **Información general** del runbook.

Uso de webhooks

En este ejemplo se usa el cmdlet de PowerShell Invoke-WebRequest para enviar una solicitud PUT al nuevo webhook.

Para usar un webhook, siga estos pasos:

1. Prepare los valores para pasar al runbook como cuerpo de la llamada de webhook. En el caso de los valores relativamente simples, puede incluir los valores en un script, como se muestra a continuación:

```
PowerShell
```

```
$Names  = @(
            @{ Name="Hawaii"},
            @{ Name="Seattle"},
            @{ Name="Florida"}
        )

$body = ConvertTo-Json -InputObject $Names
```

2. En el caso de conjuntos más grandes, puede ser aconsejable usar un archivo. Cree un archivo denominado `names.json` y pegue el código siguiente:

JSON

```
[
    { "Name": "Hawaii" },
    { "Name": "Florida" },
    { "Name": "Seattle" }
]
```

Cambie el valor de la variable `$file` por la ruta de acceso real al archivo JSON antes de ejecutar los siguientes comandos de PowerShell.

PowerShell

```
# Revise file path with actual path
$file = "path\names.json"
$bodyFile = Get-Content -Path $file
```

3. Ejecute los siguientes comandos de PowerShell para llamar al webhook mediante la API de REST.

PowerShell

```
$response = Invoke-WebRequest -Method Post -Uri $webhookURI -Body $body -
UseBasicParsing
$response

$responseFile = Invoke-WebRequest -Method Post -Uri $webhookURI -Body $bodyFile -
UseBasicParsing
$responseFile
```

Con fines ilustrativos, se realizaron dos llamadas para los dos métodos diferentes de generar el cuerpo. Para producción, use solo un método. La salida debe tener un aspecto similar al siguiente (solo se muestra una salida):

```
StatusCode        : 202
StatusDescription : Accepted
Content           : {"JobIds":["707920e8-96c7-44a3-9976-afb9ce0db982"]}
RawContent        : HTTP/1.1 202 Accepted
                    Pragma: no-cache
                    Strict-Transport-Security: max-age=31536000; includeSubDomains
                    Content-Length: 51
                    Cache-Control: no-cache
                    Content-Type: application/json; charset=utf-8
                    Dat...
Forms             :
Headers           : {[Pragma, no-cache], [Strict-Transport-Security, max-age=31536000; includeSubDomains], [Content-Length, 51], [Cache-Control, no-cache]...}
Images            : {}
InputFields       : {}
Links             : {}
ParsedHtml        :
RawContentLength  : 51
```

El cliente recibe uno de los siguientes códigos de retorno de la solicitud `POST`.

Expandir tabla

Código	Texto	Descripción
202	Accepted	La solicitud se aceptó y el runbook se puso en cola correctamente.
400	Solicitud incorrecta	No se aceptó la solicitud por uno de los siguientes motivos: • El webhook ha expirado. • El webhook está deshabilitado. • El token de la dirección URL no es válido.
404	No encontrado	No se aceptó la solicitud por uno de los siguientes motivos: • No se encontró el webhook. • No se encontró el runbook. • No se encontró la cuenta.
500	Error interno del servidor	La dirección URL es válida, pero se produjo un error. Vuelva a enviar la solicitud.

Si la solicitud es correcta, la respuesta del webhook contiene el identificador de trabajo en formato JSON, como se muestra a continuación. Contiene un solo identificador de trabajo, pero el formato JSON permite realizar potenciales mejoras en el futuro.

JSON

```
{"JobIds":["<JobId>"]}
```

4. El cmdlet de PowerShell Get-AzAutomationJobOutput se usará para obtener la salida. También se puede usar la API Azure Automation.

PowerShell

```
#isolate job ID
$jobid = (ConvertFrom-Json ($response.Content)).jobids[0]

# Get output
```

```
Get-AzAutomationJobOutput `
    -AutomationAccountName $automationAccount `
    -Id $jobid `
    -ResourceGroupName $resourceGroup `
    -Stream Output
```

Cuando se desencadena un runbook creado en el paso anterior, se crea un trabajo y la salida debe ser similar a la siguiente:

```
ResourceGroupName     : ContosoLab
AutomationAccountName : myAutomationAccount
JobId                 : 22c3c9c8-33e5-4f00-81a4-f2f16d5be3d7
StreamRecordId        : 22c3c9c8-33e5-4f00-81a4-f2f16d5be3d7_0063761208322810537 9_00000000000000000005
Time                  : 7/6/2021 10:45:22 PM +00:00
Summary               : Hello Hawaii
Type                  : Output

ResourceGroupName     : ContosoLab
AutomationAccountName : myAutomationAccount
JobId                 : 22c3c9c8-33e5-4f00-81a4-f2f16d5be3d7
StreamRecordId        : 22c3c9c8-33e5-4f00-81a4-f2f16d5be3d7_0063761208322890581 8_00000000000000000006
Time                  : 7/6/2021 10:45:22 PM +00:00
Summary               : Hello Seattle
Type                  : Output

ResourceGroupName     : ContosoLab
AutomationAccountName : myAutomationAccount
JobId                 : 22c3c9c8-33e5-4f00-81a4-f2f16d5be3d7
StreamRecordId        : 22c3c9c8-33e5-4f00-81a4-f2f16d5be3d7_0063761208322960541 9_00000000000000000007
Time                  : 7/6/2021 10:45:22 PM +00:00
Summary               : Hello Florida
Type                  : Output
```

Actualización de un webhook

Cuando se crea un webhook, tiene un período de validez de 10 años, tras el cual expira automáticamente. Una vez que un webhook ha expirado, no se puede reactivar. Solo se puede quitar y volver a crear. Puede extender un webhook que no haya alcanzado su fecha de expiración. Para ampliar un webhook, realice los pasos siguientes:

1. Vaya al runbook que contiene el webhook.
2. En **Recursos**, seleccione **Webhooks** y, a continuación, el webhook que quiera extender.
3. En la página **Webhook**, elija una nueva fecha y hora de expiración y haga clic en **Guardar**.

Revise la llamada API Webhook: Actualización y el cmdlet de PowerShell Set-AzAutomationWebhook para ver otras modificaciones posibles.

Limpieza de recursos

Estos son ejemplos de cómo quitar un webhook de un runbook de Automation.

- Con PowerShell, se puede usar el cmdlet Remove-AzAutomationWebhook como se muestra a continuación. No se devuelve salida.

PowerShell

```
Remove-AzAutomationWebhook `
    -ResourceGroup $resourceGroup `
    -AutomationAccountName $automationAccount `
    -Name $psWebhook
```

- Con REST, se puede usar la API de REST Webhook: Eliminación como se muestra a continuación.

  PowerShell

  ```
  Invoke-WebRequest -Method Delete -Uri $restURI -Headers $authHeader
  ```

  Una salida de `StatusCode : 200` significa una eliminación correcta.

Creación de un runbook y un webhook con una plantilla de ARM

Los webhooks de Automatización también se pueden crear mediante plantillas de Azure Resource Manager. Esta plantilla de ejemplo crea una cuenta de Automation, cuatro runbooks y un webhook para el runbook con nombre.

Para crear un webhook mediante una plantilla de ARM, siga estos pasos:

1. Cree un archivo denominado `webhook_deploy.json` y pegue el código siguiente:

   JSON

   ```
   {
       "$schema": "https://schema.management.azure.com/schemas/2019-04-
   01/deploymentTemplate.json#",
       "contentVersion": "1.0.0.0",
       "parameters": {
           "automationAccountName": {
               "type": "String",
               "metadata": {
                   "description": "Automation account name"
               }
           },
           "webhookName": {
               "type": "String",
               "metadata": {
                   "description": "Webhook Name"
               }
           },
           "runbookName": {
   ```

```
            "type": "String",
            "metadata": {
                "description": "Runbook Name for which webhook will be created"
            }
        },
        "WebhookExpiryTime": {
            "type": "String",
            "metadata": {
                "description": "Webhook Expiry time"
            }
        },
        "_artifactsLocation": {
            "defaultValue": "https://raw.githubusercontent.com/Azure/azure-
quickstart-templates/master/quickstarts/microsoft.automation/101-automation/",
            "type": "String",
            "metadata": {
                "description": "URI to artifacts location"
            }
        }
    },
    "resources": [
        {
            "type": "Microsoft.Automation/automationAccounts",
            "apiVersion": "2020-01-13-preview",
            "name": "[parameters('automationAccountName')]",
            "location": "[resourceGroup().location]",
            "properties": {
                "sku": {
                    "name": "Free"
                }
            },
            "resources": [
                {
                    "type": "runbooks",
                    "apiVersion": "2018-06-30",
                    "name": "[parameters('runbookName')]",
                    "location": "[resourceGroup().location]",
                    "dependsOn": [
                        "[parameters('automationAccountName')]"
                    ],
                    "properties": {
                        "runbookType": "Python2",
                        "logProgress": "false",
                        "logVerbose": "false",
                        "description": "Sample Runbook",
                        "publishContentLink": {
                            "uri": "[uri(parameters('_artifactsLocation'),
'scripts/AzureAutomationTutorialPython2.py')]",
                            "version": "1.0.0.0"
                        }
                    }
                },
                {
                    "type": "webhooks",
                    "apiVersion": "2018-06-30",
```

```
                    "name": "[parameters('webhookName')]",
                    "dependsOn": [
                        "[parameters('automationAccountName')]",
                        "[parameters('runbookName')]"
                    ],
                    "properties": {
                        "isEnabled": true,
                        "expiryTime": "[parameters('WebhookExpiryTime')]",
                        "runbook": {
                            "name": "[parameters('runbookName')]"
                        }
                    }
                }
            ]
        }
    ],
    "outputs": {
        "webhookUri": {
            "type": "String",
            "value": "[reference(parameters('webhookName')).uri]"
        }
    }
}
```

2. En el siguiente ejemplo de código de PowerShell se implementa la plantilla desde el equipo. Proporcione un valor adecuado para las variables y, a continuación, ejecute el script.

PowerShell

```
$resourceGroup = "resourceGroup"
$templateFile = "path\webhook_deploy.json"
$armAutomationAccount = "automationAccount"
$armRunbook = "ARMrunbookName"
$armWebhook = "webhookName"
$webhookExpiryTime = "12-31-2022"

New-AzResourceGroupDeployment `
    -Name "testDeployment" `
    -ResourceGroupName $resourceGroup `
    -TemplateFile $templateFile `
    -automationAccountName $armAutomationAccount `
    -runbookName $armRunbook `
    -webhookName $armWebhook `
    -WebhookExpiryTime $webhookExpiryTime
```

Nota:

Por motivos de seguridad, el URI solo se devuelve la primera vez que se implementa una plantilla.

Pasos siguientes

- Para desencadenar un libro de ejecución desde una alerta, consulte Usar una alerta para desencadenar un libro de ejecución de Automatización de Azure.
- Para solucionar problemas relacionados con el inicio de runbooks de Azure Automation mediante webhooks, consulte Solucionar problemas de runbook.

Last updated on 15/04/2026

Uso del SDK de runbooks gráficos de Azure Automation (versión preliminar)

Los runbooks gráficos ayudan a administrar las complejidades del código subyacente de Windows PowerShell o del flujo de trabajo de PowerShell. El SDK de creación de gráficos en Microsoft Azure Automation permite a los desarrolladores crear y editar runbooks gráficos para usarlos con Azure Automation. En este artículo se describen los pasos básicos para crear un runbook gráfico a partir del código.

Requisitos previos

Importación del paquete `Orchestrator.GraphRunbook.Model.dll` descargando el SDK.

Creación de una instancia de objeto de runbook

Haga referencia al ensamblado `Orchestrator.GraphRunbook.Model` y cree una instancia de la clase `Orchestrator.GraphRunbook.Model.GraphRunbook`:

C#

```
using Orchestrator.GraphRunbook.Model;
using Orchestrator.GraphRunbook.Model.ExecutableView;

var runbook = new GraphRunbook();
```

Adición de parámetros de runbook

Cree una instancia de objetos `Orchestrator.GraphRunbook.Model.Parameter` y agréguelos al runbook:

C#

```
runbook.AddParameter(
 new Parameter("YourName") {
  TypeName = typeof(string).FullName,
   DefaultValue = "World",
   Optional = true
 });

runbook.AddParameter(
 new Parameter("AnotherParameter") {
```

```
  TypeName = typeof(int).FullName, ...
 });
```

Adición de actividades y vínculos

Cree una instancia de actividades de tipos apropiados y agréguelas al runbook:

C#

```
var writeOutputActivityType = new CommandActivityType {
 CommandName = "Write-Output",
  ModuleName = "Microsoft.PowerShell.Utility",
 InputParameterSets = new [] {
  new ParameterSet {
   Name = "Default",
    new [] {
     new Parameter("InputObject"), ...
    }
  },
  ...
 }
};

var outputName = runbook.AddActivity(
 new CommandActivity("Output name", writeOutputActivityType) {
  ParameterSetName = "Default",
   Parameters = new ActivityParameters {
    {
     "InputObject",
     new RunbookParameterValueDescriptor("YourName")
    }
   },
   PositionX = 0,
   PositionY = 0
 });

var initializeRunbookVariable = runbook.AddActivity(
 new WorkflowScriptActivity("Initialize variable") {
  Process = "$a = 0",
   PositionX = 0,
   PositionY = 100
 });
```

Las actividades las implementan las clases siguientes en el espacio de nombres `Orchestrator.GraphRunbook.Model`.

Expandir tabla

Clase	Actividad
CommandActivity	Invoca un comando de PowerShell (cmdlet, función, etc.).
InvokeRunbookActivity	Invoca otro runbook alineado.
JunctionActivity	Espera a que todas las ramas entrantes finalicen.
WorkflowScriptActivity	Ejecuta un bloque de código de PowerShell o del flujo de trabajo de PowerShell (según el tipo de runbook) en el contexto del runbook. Se trata de una herramienta eficaz, pero no se debe abusar de su uso: la interfaz de usuario mostrará este bloque de script como texto; el motor de ejecución tratará el bloque proporcionado como una caja negra y no intentará analizar su contenido, excepto para una comprobación de sintaxis básica. Si desea invocar un único comando de PowerShell, se prefiere CommandActivity.

> **Nota:**
>
> No derive sus propias actividades de las clases proporcionadas. Azure Automation no puede usar runbooks con tipos de actividades personalizadas.

Debe proporcionar los parámetros `CommandActivity` y `InvokeRunbookActivity` como descriptores de valor, y no como valores directos. Los descriptores de valor especifican cómo se deben crear los valores de parámetros reales. Actualmente se proporcionan los siguientes descriptores de valor:

Expandir tabla

Descriptor	Definición
ConstantValueDescriptor	Hace referencia a un valor constante codificado de forma rígida.
RunbookParameterValueDescriptor	Hace referencia a un parámetro de runbook por nombre.
ActivityOutputValueDescriptor	Hace referencia a una salida de actividad ascendente, lo que permite que una actividad "se suscriba" a los datos creados por otra actividad.
AutomationVariableValueDescriptor	Hace referencia a un recurso de variable de Automation por nombre.
AutomationCredentialValueDescriptor	Hace referencia a un recurso de certificado de Automation por nombre.
AutomationConnectionValueDescriptor	Hace referencia a un recurso de conexión de Automation por nombre.

Descriptor	Definición
PowerShellExpressionValueDescriptor	Especifica una expresión de PowerShell de forma libre que se va a evaluar justo antes de invocar la actividad. Se trata de una herramienta eficaz, pero no se debe abusar de su uso: la interfaz de usuario mostrará esta expresión como texto; el motor de ejecución tratará el bloque proporcionado como una caja negra y no intentará analizar su contenido, excepto para una comprobación de sintaxis básica. Cuando sea posible, se prefieren descriptores de valor más específicos.

> ⓘ **Nota:**
>
> No derive sus propios descriptores de valores de las clases proporcionadas. Azure Automation no puede usar runbooks con tipos de descriptores de valores personalizados.

Cree instancias de los vínculos que conectan las actividades y agréguelos al runbook:

C#

```
runbook.AddLink(new Link(activityA, activityB, LinkType.Sequence));

runbook.AddLink(
 new Link(activityB, activityC, LinkType.Pipeline) {
  Condition = string.Format("$ActivityOutput['{0}'] -gt 0", activityB.Name)
 });
```

Guardar el runbook en un archivo

Use `Orchestrator.GraphRunbook.Model.Serialization.RunbookSerializer` para serializar un runbook en una cadena:

C#

```
var serialized = RunbookSerializer.Serialize(runbook);
```

Puede guardar esta cadena en un archivo con la extensión **.graphrunbook**. El runbook correspondiente se puede importar en Azure Automation. El formato serializado puede cambiar en las futuras versiones de `Orchestrator.GraphRunbook.Model.dll`. Prometemos compatibilidad con versiones anteriores: cualquier runbook serializado con una versión anterior de `Orchestrator.GraphRunbook.Model.dll` se puede deserializar en cualquier versión más reciente. No se garantiza la compatibilidad con versiones posteriores: puede que un runbook serializado con una versión más reciente no se deserialice en versiones anteriores.

Pasos siguientes

Para más información, consulte Control de errores en runbooks gráficos de Azure Automation.

Last updated on 17/11/2025

Configurar flujos de salida y mensajes de runbook

La mayoría de los runbooks de Azure Automation tienen alguna forma de salida. Esta salida puede ser un mensaje de error para el usuario o un objeto complejo que se va a usar con otro runbook. Windows PowerShell proporciona múltiples flujos para enviar la salida desde un script o flujo de trabajo. Azure Automation funciona con cada uno de estos flujos de forma diferente. Debe seguir los procedimientos recomendados sobre cómo usar los flujos al crear un runbook.

En la tabla siguiente se describe brevemente cada secuencia con su comportamiento en el portal de Azure para runbooks publicados y durante las pruebas de un libro de ejecución. El flujo de salida es el flujo principal que se utiliza para la comunicación entre los runbooks. El resto de los flujos se clasifican como flujos de mensajes, pensados para comunicar información al usuario.

Expandir tabla

Flujo	Descripción	Publicado	Prueba
Error	Mensaje de error destinado al usuario. A diferencia de lo que sucede con las excepciones, el manual de operaciones continúa después de un mensaje de error por defecto.	Se ha escrito en el historial de trabajos.	Se muestra en el panel de salida de la prueba.
Depurar	Mensajes dirigidos a un usuario interactivo. No debe usarse en runbooks.	No se registra en el historial de trabajos.	No se muestra en el panel de salida de la prueba.
Salida	Objetos diseñados para ser utilizados por otros runbooks.	Se ha escrito en el historial de trabajos.	Se muestra en el panel de salida de la prueba.
Progreso	Registros generados automáticamente antes y después de cada actividad en el runbook. El runbook no debe intentar crear sus propios registros de progreso, ya que están dirigidos a un usuario interactivo.	Solo se escribe en el historial de tareas si está activado el registro de avance del runbook.	No se muestra en el panel de salida de la prueba.
Verbose	Mensajes que proporcionan información general o de depuración.	Solo se escribe en el historial de trabajos si el	Se muestra en el panel de salida de la prueba solo si la variable `VerbosePreference`

Flujo	Descripción	Publicado	Prueba
		registro detallado del runbook está activado.	está establecida en Continue en el runbook.
Advertencia	Mensaje de advertencia destinado al usuario.	Se ha escrito en el historial de trabajos.	Se muestra en el panel de salida de la prueba.

Utilice el flujo de salida

El flujo de salida se usa para la salida de los objetos creados por un script o flujo de trabajo, cuando se ejecuta correctamente. Azure Automation usa principalmente este flujo para que los runbooks ascendientes consuman los objetos cuando llaman al runbook actual current. Cuando un runbook principal llama a un runbook en línea, el runbook hijo devuelve los datos del flujo de salida al principal.

El runbook utiliza el canal de salida para comunicar información general al cliente, únicamente cuando no se le llama desde otro runbook. Como procedimiento recomendado, sin embargo, le recomendamos que los runbooks usen la opción de flujo detallado para comunicar información general al usuario.

Para que el runbook escriba datos en el flujo de salida, utilice Write-Output. Como alternativa, puede colocar el objeto en su propia línea en el script.

PowerShell

```
#The following lines both write an object to the output stream.
Write-Output -InputObject $object
$object
```

Gestión de la salida de una función

Cuando una función de runbook escribe en el flujo de salida, dicha salida se devuelve al runbook. Si el runbook asigna esa salida a una variable, la salida no se escribe en el flujo de salida. Asimismo, si se escribe en cualquier otro flujo desde la función, se escribirá en el flujo correspondiente del runbook. Considere el siguiente ejemplo de runbook del flujo de trabajo de PowerShell.

PowerShell

```
Workflow Test-Runbook
{
  Write-Verbose "Verbose outside of function" -Verbose
  Write-Output "Output outside of function"
  $functionOutput = Test-Function
  $functionOutput

  Function Test-Function
  {
    Write-Verbose "Verbose inside of function" -Verbose
    Write-Output "Output inside of function"
  }
}
```

El flujo de salida del trabajo del runbook es:

Resultados

```
Output inside of function
Output outside of function
```

El flujo detallado del trabajo del runbook es:

Resultados

```
Verbose outside of function
Verbose inside of function
```

Una vez que haya publicado el runbook y antes de iniciarlo, debe activar también el registro detallado en la configuración del runbook para obtener la salida del flujo detallado.

Declaración del tipo de datos de salida

A continuación se muestran ejemplos de tipos de datos de salida:

- `System.String`
- `System.Int32`
- `System.Collections.Hashtable`
- `Microsoft.Azure.Commands.Compute.Models.PSVirtualMachine`

Declaración del tipo de datos de salida de un flujo de trabajo

Un flujo de trabajo especifica el tipo de datos de su salida mediante el atributo OutputType. Este atributo no tiene ningún efecto en tiempo de ejecución, pero en tiempo de diseño proporciona una indicación de la salida esperada del runbook. A medida que el conjunto de herramientas

para los runbooks sigue evolucionando, la importancia de declarar los tipos de datos de salida durante el diseño aumenta. Por tanto, es una buena práctica incluir esta declaración en cualquier libro de operaciones que se cree.

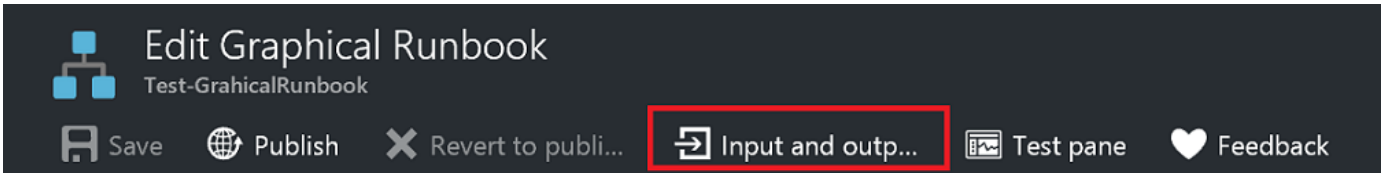
El siguiente runbook de ejemplo genera un objeto de cadena e incluye una declaración de su tipo de salida. Si su runbook genera una matriz de un tipo determinado, deberá especificar el tipo en lugar de una matriz de ese tipo.

PowerShell

```
Workflow Test-Runbook
{
  [OutputType([string])]

  $output = "This is some string output."
  Write-Output $output
}
```

Declara el tipo de datos de salida en un runbook gráfico

Para declarar un tipo de salida en un runbook gráfico o en un flujo de trabajo gráfico de PowerShell, seleccione la opción de menú **Entrada y salida** y especifique el tipo de salida. Se recomienda usar el nombre completo de la clase .NET para que el tipo se pueda identificar fácilmente cuando un runbook principal hace referencia a él. Al usar el nombre completo, se exponen todas las propiedades de la clase al bus de datos del runbook. Además, se aumenta la flexibilidad cuando se usan las propiedades para la lógica condicional, el registro y la referencia como valores para otras actividades de runbooks.

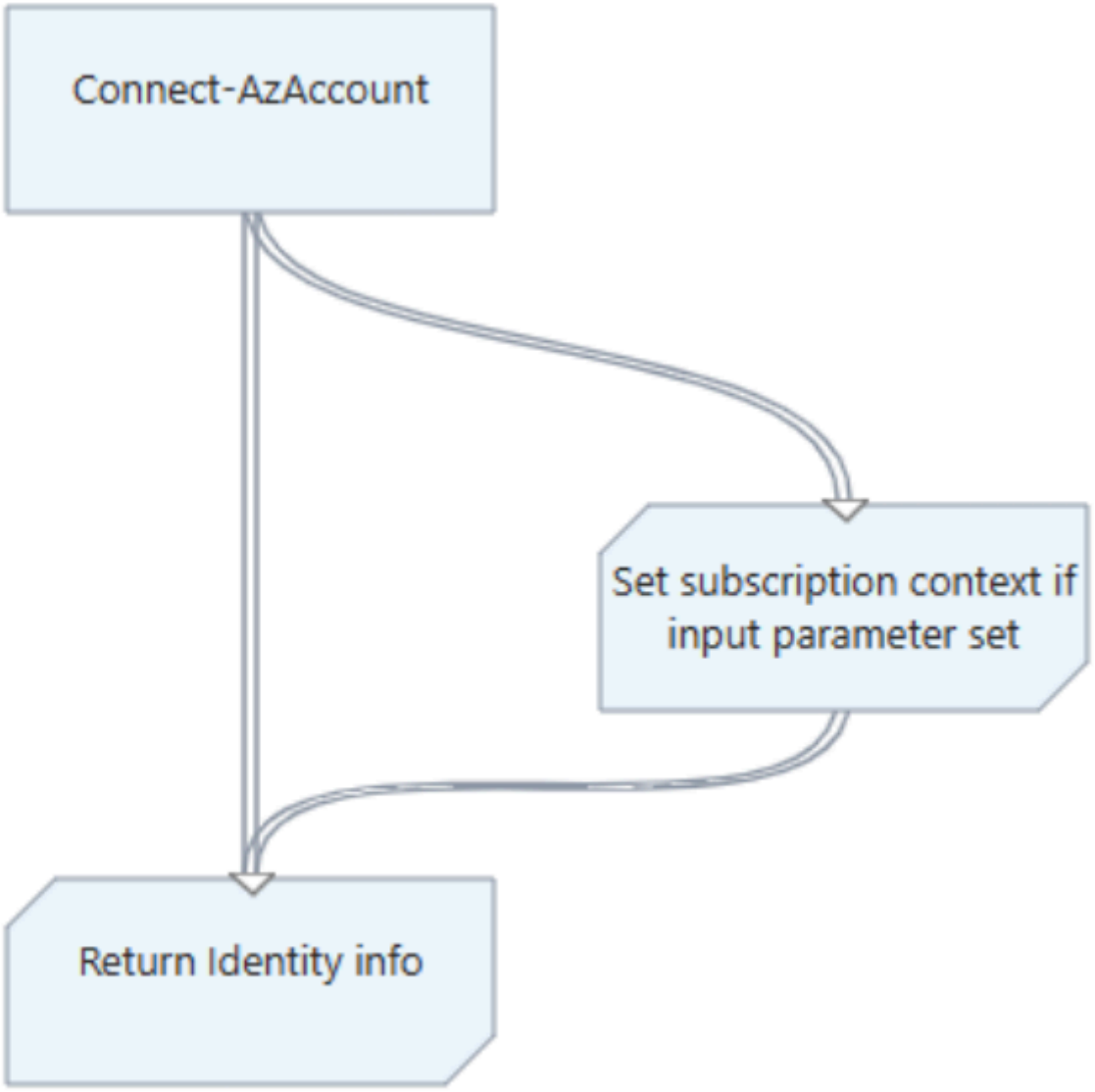


> **Nota:**
>
> Después de escribir un valor en el cuadro **Tipo de salida** del panel Propiedades de entrada y salida, asegúrese de hacer clic fuera del control para que reconozca la entrada.

En el ejemplo siguiente se utilizan dos runbooks gráficos para mostrar la característica de entrada y salida. Al aplicar el modelo de diseño de runbook modular, tiene un runbook como plantilla Runbook de Autenticación que gestiona la autenticación con Azure utilizando Managed identities. El segundo runbook, que normalmente ejecuta la lógica principal para automatizar un

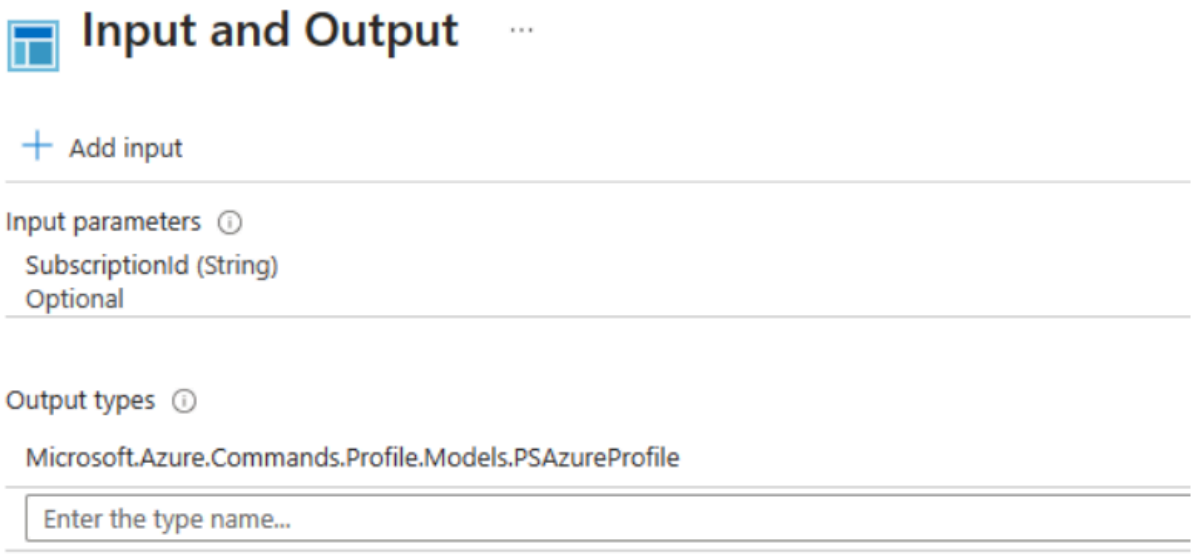
determinado escenario, en este caso, ejecuta la plantilla de Autenticación Runbook. Muestra los resultados en el panel de salida de la prueba. En circunstancias normales, harías que este runbook realizara alguna acción contra un recurso utilizando la salida del runbook secundario.

Esta es la lógica básica del runbook **AuthenticateTo-Azure**.



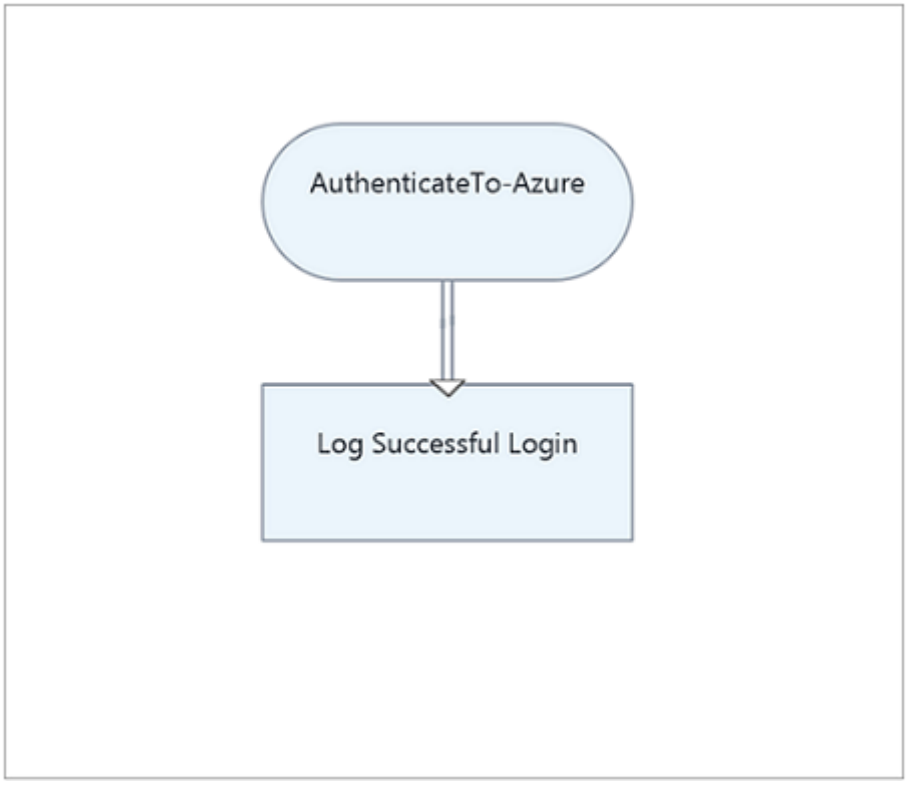
.

El runbook incluye un tipo de salida `Microsoft.Azure.Commands.Profile.Models.PSAzureProfile`, que devuelve las propiedades del perfil de autenticación.



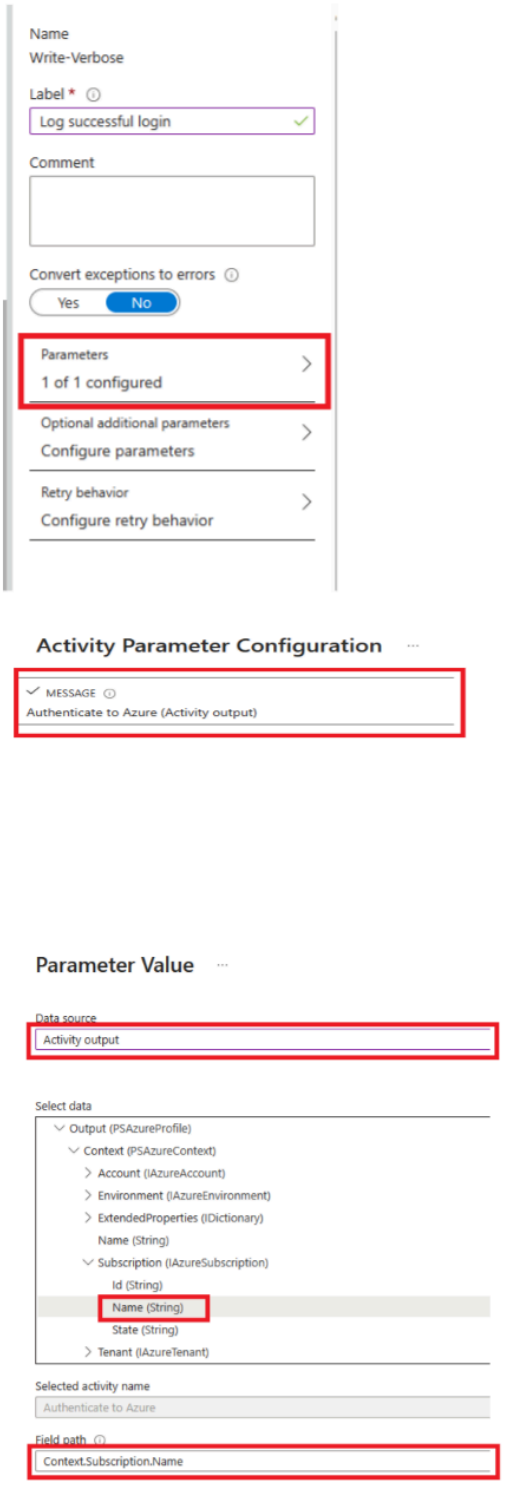
Aunque este runbook es sencillo, hay un elemento de configuración que merece destacarse. La última actividad ejecuta el cmdlet `Write-Output` para escribir datos de perfil en una variable mediante una expresión de PowerShell para el parámetro `Inputobject`. Este parámetro es obligatorio para `Write-Output`.

El segundo runbook de este ejemplo, denominado **Test-ChildOutputType**, simplemente define dos actividades.

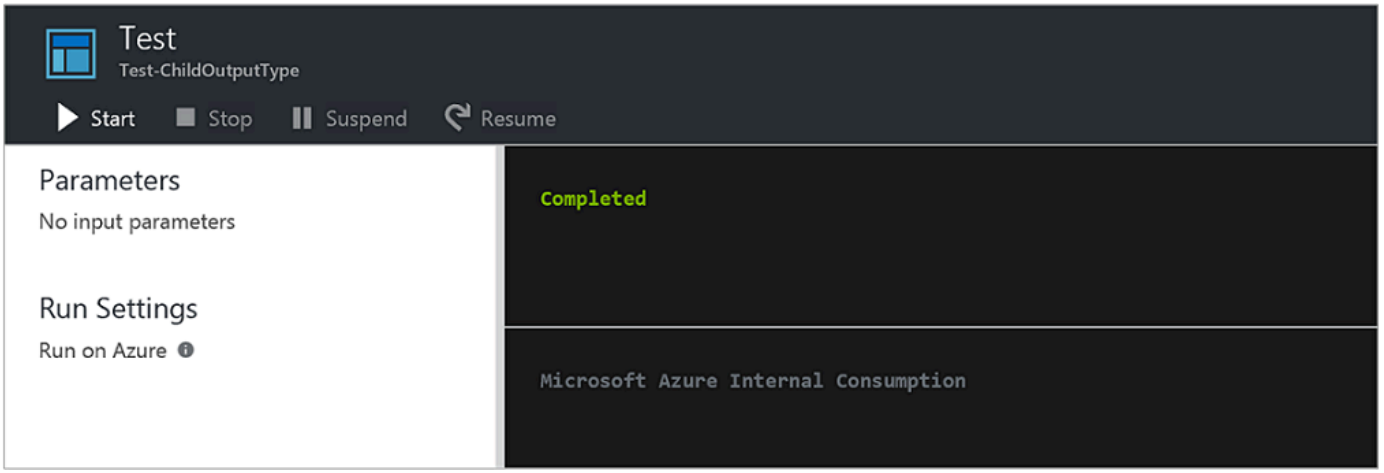


La primera actividad llama al runbook **AuthenticateTo-Azure**. La segunda actividad ejecuta el cmdlet `Write-Verbose` con la opción **Origen de datos** establecida en **Resultado de la actividad**.

Además de esto, **Field path** se establece en **Context.Subscription.Name**, la salida de contexto del runbook **AuthenticateTo-Azure**.



La salida resultante es el nombre de la suscripción.



Uso de flujos de mensajes

A diferencia del flujo de salida, los flujos de mensajes comunican información al usuario. Hay varios flujos de mensajes para diferentes tipos de información y Azure Automation controla cada secuencia de forma diferente.

Escribir la salida en flujos de advertencia y de error

Los problemas que se producen en un libro de operación se registran en los flujos de advertencia y error. Azure Automation escribe estas secuencias en el historial de trabajos al ejecutar un runbook. Automation incluye los flujos en el panel Salida de prueba del portal de Azure cuando se prueba un runbook.

De forma predeterminada, un libro de operaciones continúa ejecutándose después de una advertencia o un error. Puede especificar que el runbook se suspenda en caso de advertencia o error haciendo que el runbook establezca una variable de preferencia antes de crear el mensaje. Por ejemplo, para hacer que el runbook se suspenda en caso de que se produzca un error, al igual que lo hace en caso de una excepción, establezca la variable `ErrorActionPreference` en Stop.

Cree un mensaje de advertencia o de error mediante los cmdlets Write-Warning o Write-Error. Las actividades también pueden escribir en los flujos de advertencia y de error.

PowerShell

```
#The following lines create a warning message and then an error message that will
suspend the runbook.

$ErrorActionPreference = "Stop"
Write-Warning -Message "This is a warning message."
```

```
Write-Error -Message "This is an error message that will stop the runbook because of the preference variable."
```

Escritura de la salida en el flujo de depuración

Azure Automation usa la secuencia de mensajes de depuración para los usuarios interactivos. De forma predeterminada, Azure Automation no captura ningún dato de flujo de depuración, solo se capturan datos de salida, error y advertencia, así como datos detallados si el runbook está configurado para capturarlos.

Para capturar datos de flujos de depuración, debe realizar dos acciones en los runbooks:

1. Establezca la variable `$GLOBAL:DebugPreference="Continue"`, que indica a PowerShell que continúe cada vez que se encuentre un mensaje de depuración. La parte **$GLOBAL:** indica a PowerShell que lo haga en el ámbito global, en lugar de en el ámbito local en el que se encuentre el script en el momento de ejecución de la instrucción.

2. Redirija el flujo de depuración que no capturamos a un flujo que sí capturamos, como *output*. Para lograr esto, configure el redireccionamiento de PowerShell para la instrucción que se debe ejecutar. Para obtener más información sobre el redireccionamiento de PowerShell, consulte Acerca del redireccionamiento.

Ejemplos

En este ejemplo, el runbook se configura mediante los cmdlets `Write-Output` y `Write-Debug` con la intención de generar dos flujos diferentes.

PowerShell

```
Write-Output "This is an output message."
Write-Debug "This is a debug message."
```

Si este runbook fuera a ejecutarse tal cual, el panel de salida del trabajo del runbook transmitiría el siguiente resultado:

Resultados

```
This is an output message.
```

En este ejemplo, el runbook se ha configurado de forma similar al ejemplo anterior, salvo que la instrucción `$GLOBAL:DebugPreference="Continue"` se incluye con la adición de `5>&1` al final de la instrucción `Write-Debug`.

PowerShell

```
Write-Output "This is an output message."
$GLOBAL:DebugPreference="Continue"
Write-Debug "This is a debug message." 5>&1
```

Si se ejecutara este runbook, el panel de salida del trabajo del runbook mostraría el siguiente resultado:

Resultados

```
This is an output message.
This is a debug message.
```

Esto se debe a que la instrucción `$GLOBAL:DebugPreference="Continue"` indica a PowerShell que muestre los mensajes de depuración, y la adición de `5>&1` al final de la instrucción `Write-Debug` indica a PowerShell que redirija el flujo 5 (debug) al flujo 1 (output).

Escribir la salida en un flujo detallado

El flujo de mensajes detallados ofrece información general acerca del funcionamiento del runbook. Dado que la secuencia de depuración no está disponible para un runbook, el runbook puede usar mensajes verbosos para la información de depuración.

De forma predeterminada, el historial de trabajos no almacena mensajes verbosos de runbooks publicados por motivos de rendimiento. Para almacenar mensajes detallados, use la pestaña **Configure** del portal de Azure con la configuración **Log Verbose Records** para que sus runbooks publicados registren mensajes detallados. Active esta opción solo para solucionar problemas o para depurar un runbook. En la mayoría de los casos, debería mantener la configuración predeterminada de no registrar registros detallados.

Cuando se prueba un runbook, los mensajes detallados no se muestran aunque el runbook esté configurado para escribir registros detallados. Para mostrar mensajes detallados al probar un runbook, debe establecer la variable `VerbosePreference` en "Continue". Con esa variable establecida, los mensajes verbosos se muestran en el panel de salida de prueba del portal de Azure.

En el código siguiente se crea un mensaje detallado mediante el cmdlet Write-Verbose.

PowerShell

```
#The following line creates a verbose message.
```

```
Write-Verbose -Message "This is a verbose message."
```

Manejo de registros de progreso

Puede usar la pestaña **Configure** del portal de Azure para configurar un runbook para registrar registros de progreso. En la configuración predeterminada no se escriben los registros, con el fin de maximizar el rendimiento. En la mayoría de los casos, debe mantener la configuración predeterminada. Active esta opción solo para solucionar problemas o para depurar un runbook.

Si habilita el registro de progreso de la actividad, el runbook escribe un registro en el historial de trabajos antes y después de ejecutar cada actividad. Cuando se prueba un runbook, no se muestran los mensajes de progreso, aunque el runbook esté configurado para registrar estos registros.

> **Nota:**
>
> El cmdlet Write-Progress no es válido en un runbook, ya que está pensado para usarse con un usuario interactivo.

Trabajo con variables de preferencia

Puede establecer determinadas variables de Windows PowerShell preference en los runbooks para controlar la respuesta a los datos enviados a diferentes flujos de salida. En la siguiente tabla se enumeran las variables de preferencia que pueden usarse en runbooks, junto con sus valores predeterminados y válidos. Hay valores adicionales disponibles para las variables de preferencia cuando se usan en Windows PowerShell fuera de Azure Automation.

Expandir tabla

Variable	Valor predeterminado	Valores válidos
`WarningPreference`	Continuar	Parar Continuar ContinuarSinMostrar
`ErrorActionPreference`	Continuar	Parar Continuar ContinuarSinMostrar
`VerbosePreference`	ContinuarSinMostrar	Parar Continuar

Variable	Valor predeterminado	Valores válidos
		ContinuarSinMostrar

En la siguiente tabla se muestra el comportamiento de los valores de las variables de preferencia que son válidos en los runbooks.

Expandir tabla

Valor	Comportamiento
Continuar	Registra el mensaje y continúa ejecutando el runbook.
ContinuarSinMostrar	Continúa la ejecución del runbook sin registrar el mensaje. Este valor hace que se ignore el mensaje.
Parar	Registra el mensaje y suspende el runbook.

Recupera la salida del runbook y los mensajes

Recuperar la salida y los mensajes del runbook en el portal de Azure.

Puede ver los detalles de una tarea del runbook en el portal de Azure mediante la pestaña **Jobs** del runbook. El resumen del trabajo muestra los parámetros de entrada y el flujo de salida, además de información general sobre el trabajo y las excepciones que se hayan producido. El historial de trabajos incluye los mensajes del flujo de salida, así como de los flujos de advertencia y de error. También incluye los mensajes del flujo detallado y los registros de progreso si el runbook está configurado para registrar el flujo detallado y los registros de progreso.

> **Nota:**
>
> Los flujos de trabajo para los runbooks de Python actualmente son compatibles con la salida en inglés.

Obtener la salida y los mensajes del runbook en Windows PowerShell

En Windows PowerShell, puede recuperar la salida y los mensajes de un runbook utilizando el cmdlet Get-AzAutomationJobOutput. Este cmdlet requiere la identificación del trabajo y tiene un parámetro denominado `Stream` en el cual se especifica qué flujo hay que recuperar. Puede

especificar un valor de Any para este parámetro, con el fin de recuperar todos los flujos del trabajo.

En el siguiente ejemplo, se inicia un runbook y luego se espera a que finalice. Una vez que el runbook finaliza la ejecución, el script recopila su flujo de salida del trabajo.

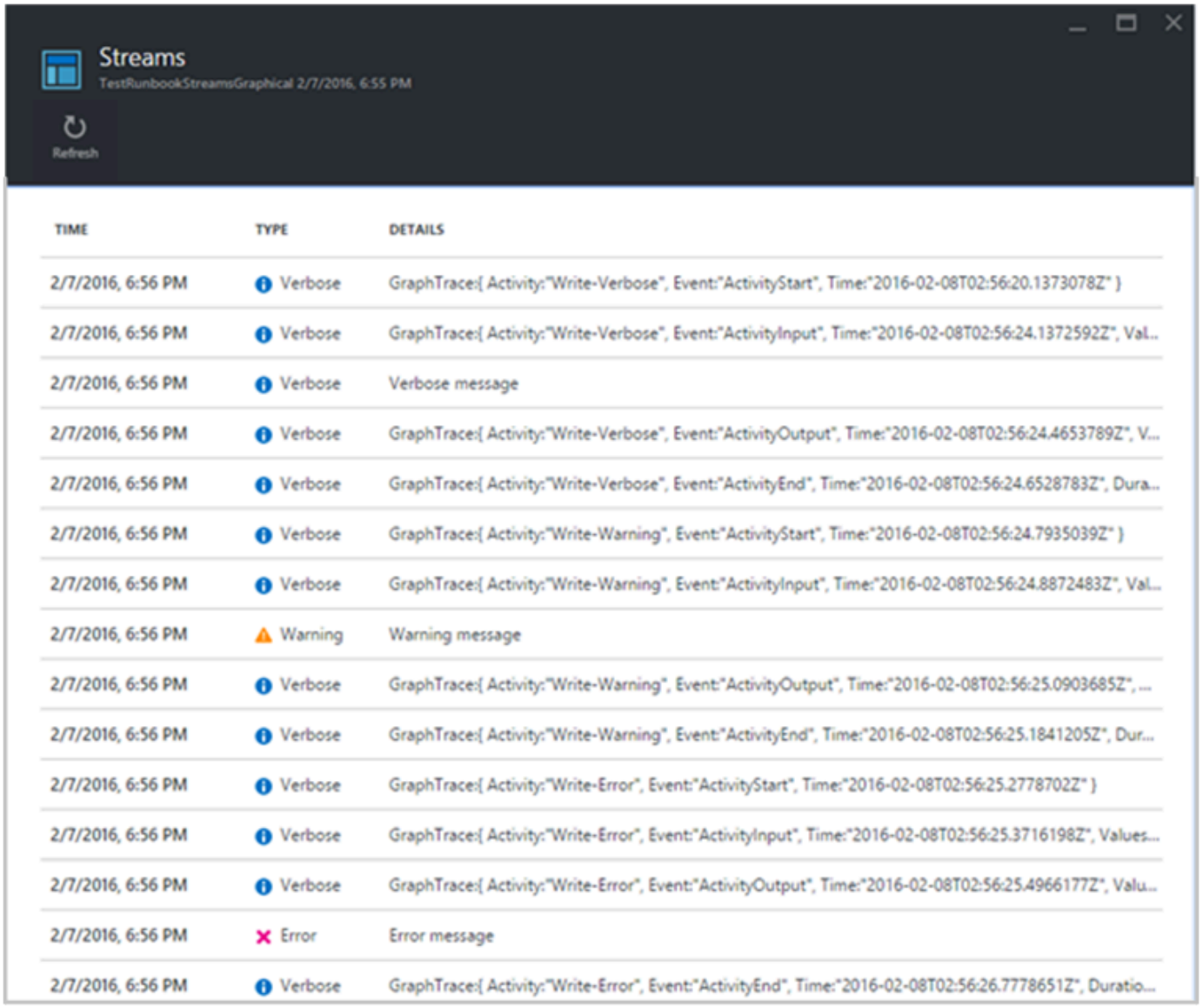
PowerShell

```
$job = Start-AzAutomationRunbook -ResourceGroupName "ResourceGroup01" `
  -AutomationAccountName "MyAutomationAccount" -Name "Test-Runbook"

$doLoop = $true
While ($doLoop) {
  $job = Get-AzAutomationJob -ResourceGroupName "ResourceGroup01" `
    -AutomationAccountName "MyAutomationAccount" -Id $job.JobId
  $status = $job.Status
  $doLoop = (($status -ne "Completed") -and ($status -ne "Failed") -and ($status -ne
"Suspended") -and ($status -ne "Stopped"))
}

Get-AzAutomationJobOutput -ResourceGroupName "ResourceGroup01" `
  -AutomationAccountName "MyAutomationAccount" -Id $job.JobId -Stream Output

# For more detailed job output, pipe the output of Get-AzAutomationJobOutput to Get-
AzAutomationJobOutputRecord
Get-AzAutomationJobOutput -ResourceGroupName "ResourceGroup01" `
  -AutomationAccountName "MyAutomationAccount" -Id $job.JobId -Stream Any | Get-
AzAutomationJobOutputRecord
```

Recuperación de salidas de runbook y mensajes en runbooks gráficos

En los runbooks gráficos, hay disponible un registro adicional de salida y mensajes en forma de seguimiento del nivel de actividad. Hay dos niveles de seguimiento: básico y detallado. El seguimiento básico muestra la hora de inicio y de finalización de cada actividad en el runbook, así como información relacionada con los reintentos de cualquier actividad. como el número de intentos y la hora de inicio de la actividad. El seguimiento detallado incluye las características del seguimiento básico más el registro de los datos de entrada y salida de cada actividad.

Actualmente, el seguimiento a nivel de actividad escribe registros utilizando el canal de flujo detallado. Por lo tanto, debe habilitar el registro detallado cuando habilite el seguimiento. Para los runbooks gráficos con el seguimiento habilitado, no hay necesidad de registrar registros de progreso. El seguimiento básico tiene la misma finalidad y es más informativo.



En la imagen puede ver que, cuando se habilita el registro detallado y el seguimiento para los runbooks gráficos, hay mucha más información disponible en la vista **Transmisiones de trabajo** de producción. Esta información adicional puede ser esencial para solucionar problemas de producción con un "runbook" o manual de operaciones.

Sin embargo, a menos que necesite esta información para realizar un seguimiento del progreso de un runbook para la resolución de problemas, como práctica general le conviene mantener el seguimiento desactivado. Los registros de seguimiento pueden ser particularmente numerosos. Con el trazado gráfico de runbooks, puede obtener de dos a cuatro registros por actividad, según la configuración de trazado básico o detallado.

Para habilitar el seguimiento en el nivel de actividad:

1. En el portal de Azure, abra la cuenta de Automation.

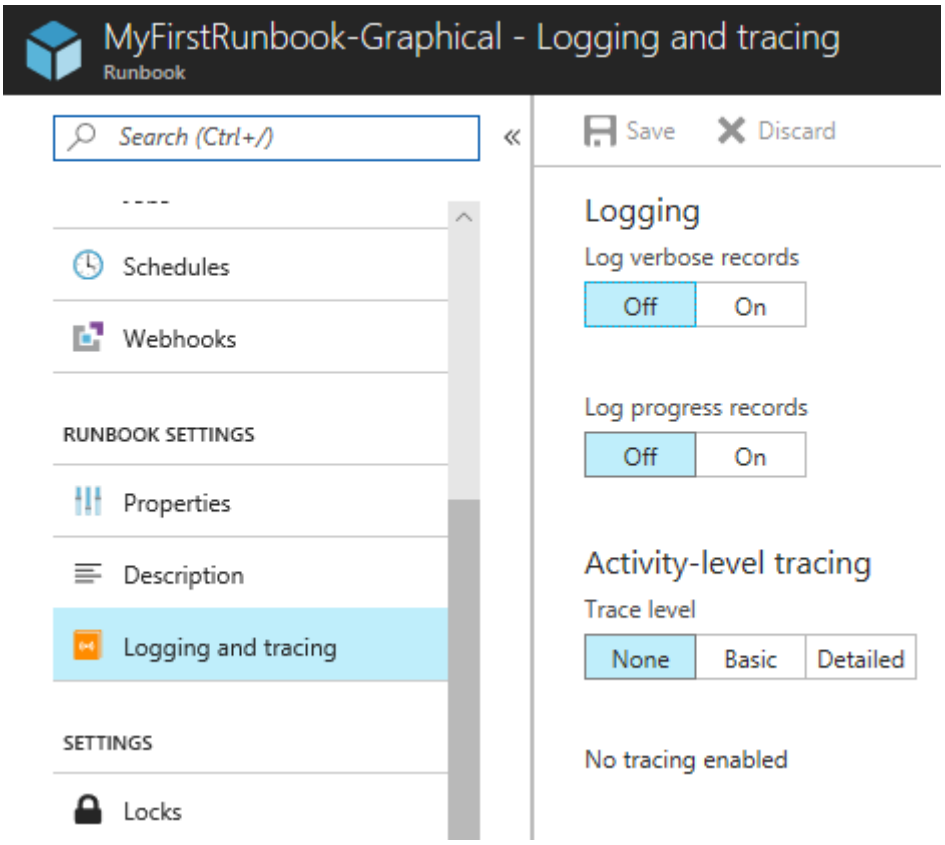
2. En **Automatización de procesos**, haga clic en **Runbooks** para abrir la lista de runbooks.

3. En la página de Runbooks, seleccione un runbook gráfico de su lista de runbooks.

4. En **Configuración**, haga clic en **registro y seguimiento**.

5. En la página Registro y seguimiento, en **Registrar registros detallados**, haga clic en **Activar** para habilitar el registro detallado.

6. En **Seguimiento a nivel de actividad**, cambie el nivel de seguimiento a **Básico** o **Detallado**, en función del nivel de seguimiento que necesite.



Recuperar la salida y los mensajes del runbook en los registros de Microsoft Azure Monitor

Azure Automation puede enviar el estado de los trabajos de runbook y los flujos de trabajo a su espacio de trabajo de Log Analytics. Azure Monitor admite registros que le permiten:

- Obtener información sobre los trabajos de automatización.
- Desencadenar un correo electrónico o una alerta en función del estado del trabajo del runbook (por ejemplo, Error o Suspendido).
- Escribir consultas avanzadas en los flujos de trabajo.
- Correlacionar trabajos entre cuentas de Automation.
- Visualizar el historial de trabajos.

Para obtener más información sobre cómo configurar la integración con registros de Azure Monitor para recopilar, correlacionar y actuar sobre los datos de trabajo, consulte Forward job

status and job streams from Automation to Azure Monitor Logs.

Pasos siguientes

- Para ver consultas de ejemplo, consulte Consultas de ejemplo para registros de trabajos y flujos de trabajo
- Para trabajar con runbooks, consulte Administrar runbooks en Azure Automation.
- Si no está familiarizado con el scripting de PowerShell, consulte la documentación de PowerShell.
- Para obtener la referencia del cmdlet de PowerShell de Azure Automation, consulte Az.Automation.
- Para solucionar problemas relacionados con la salida del runbook y los flujos de mensajes, consulte Solución de problemas de runbook.

Last updated on 15/04/2026

Control de errores en runbooks gráficos

Un principio de diseño clave que se debe tener en cuenta para su runbook gráfico de Azure Automation es la identificación de problemas que puede experimentar el runbook durante la ejecución. Dichos problemas pueden incluir condiciones de correcto, estados de error esperado y condiciones de error inesperado.

A menudo, aunque se produzca un error de no terminación en una actividad de runbook, Windows PowerShell controla la actividad procesando cualquier actividad posterior, independientemente del error. Es probable que el error genere una excepción, pero se podrá ejecutar la siguiente actividad.

Su runbook gráfico debe incluir el código de control de errores para lidiar con problemas de ejecución. Para validar el resultado de una actividad o controlar un error, se puede usar una actividad de código de PowerShell, definir una lógica condicional en el vínculo de salida de la actividad o aplicar otro método.

Los runbooks gráficos de Azure Automation se han mejorado con la funcionalidad de inclusión del control de errores. Ahora es posible convertir las excepciones en errores de no terminación y crear vínculos de error entre las actividades. El proceso mejorado permite a su runbook detectar errores y administrar cualquier condición realizada o inesperada.

Tipos de errores de PowerShell

Los tipos de errores de PowerShell que se pueden producir durante la ejecución de runbooks son: errores de terminación y errores de no terminación.

Error de terminación

Un error de terminación es un error grave durante la ejecución que detiene un comando o la ejecución del script por completo. Entre los ejemplos se incluyen cmdlets inexistentes, errores de sintaxis que impiden la ejecución de un cmdlet y otros errores irrecuperables.

Error de no terminación

Un error de no terminación es un error que no es grave, lo que permite que la ejecución no se interrumpa pese a la condición de error. Entre los ejemplos se incluyen errores operativos, como los errores de archivos no encontrados y problemas de permisos.

Cuándo se debe utilizar el control de errores

Use el control de errores en su runbook cuando una actividad crítica produzca un error o una excepción. Es importante impedir que se procese la siguiente actividad del runbook, así como controlar el error adecuadamente. Controlar el error es especialmente crítico cuando los runbooks dan soporte a un proceso de operaciones de un negocio o un servicio.

Adición de vínculos de error

Por cada actividad que pueda generar un error, puede agregar un vínculo de error que señale a cualquier otra actividad. La actividad de destino puede ser de cualquier tipo, entre las que se incluyen la actividad de código, la invocación de un cmdlet, la invocación de otro runbook, etc. La actividad de destino también puede tener vínculos salientes, ya sean vínculos regulares o de error. Los vínculos permiten al runbook implementar una lógica de control de errores compleja sin tener que recurrir a una actividad de código.

La práctica recomendada consiste en crear un runbook específico para el control de errores con una funcionalidad común, pero dicha práctica no es obligatoria. Por ejemplo, piense en un runbook que intenta iniciar una máquina virtual e instalar una aplicación en ella. Si la máquina virtual no se inicia correctamente, realiza las siguientes acciones:

1. Envía una notificación acerca de este problema.
2. Inicia otro runbook que aprovisiona automáticamente una nueva máquina virtual.

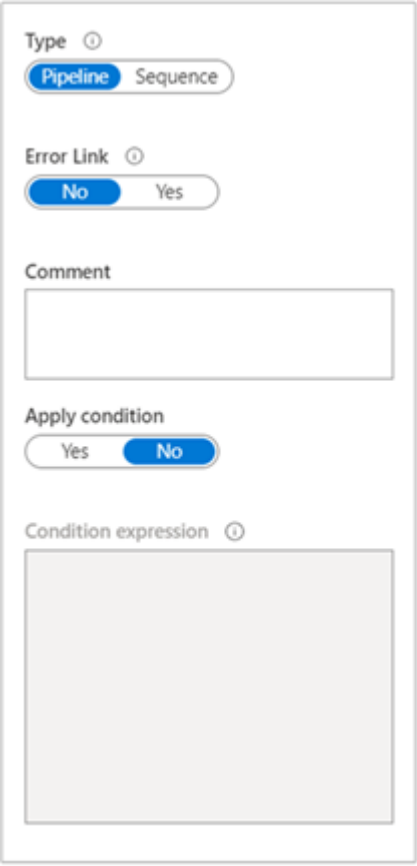
Una solución es tener un vínculo de error en el runbook que apunte a una actividad que controle el paso uno. Por ejemplo, el runbook puede conectar el cmdlet `Write-Warning` a una actividad para el paso dos, como el cmdlet Start-AzAutomationRunbook.

Este comportamiento también se puede generalizar para que se pueda usar en muchos runbooks. Solo es preciso colocar estas dos actividades en un runbook de control de errores independiente. Antes de que su runbook original llame a este runbook de control de errores se puede construir un mensaje personalizado a partir de sus datos y, después, pasarlo como parámetro al runbook de control de errores.

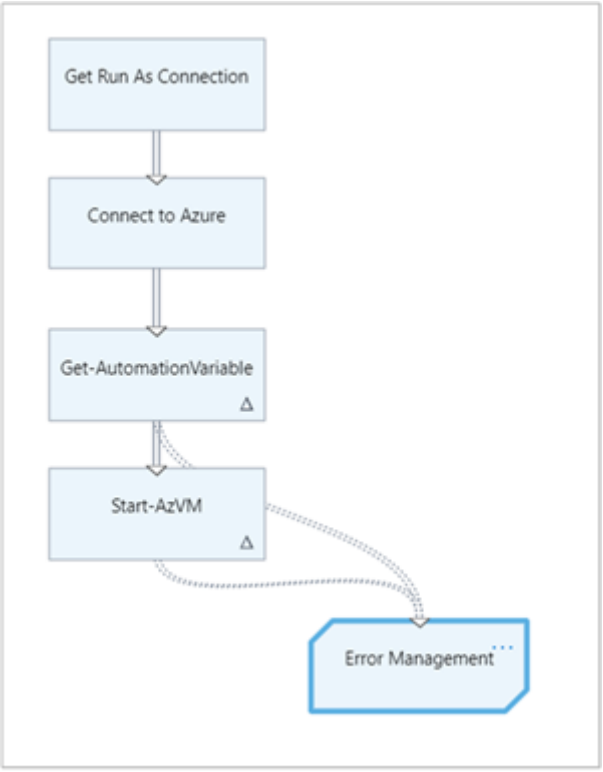
Conversión de excepciones en errores de no terminación

Todas las actividades de su runbook tienen una opción de configuración que convierte las excepciones en errores que no provocan la terminación. Esta opción está deshabilitada de forma predeterminada. Se recomienda habilitar esta opción en cualquier actividad en la que el runbook controle los errores. Esta opción garantiza que el runbook trate tanto los errores de la actividad que provocan terminación como los que no la provocan como errores que no la provocan, con un vínculo de error.

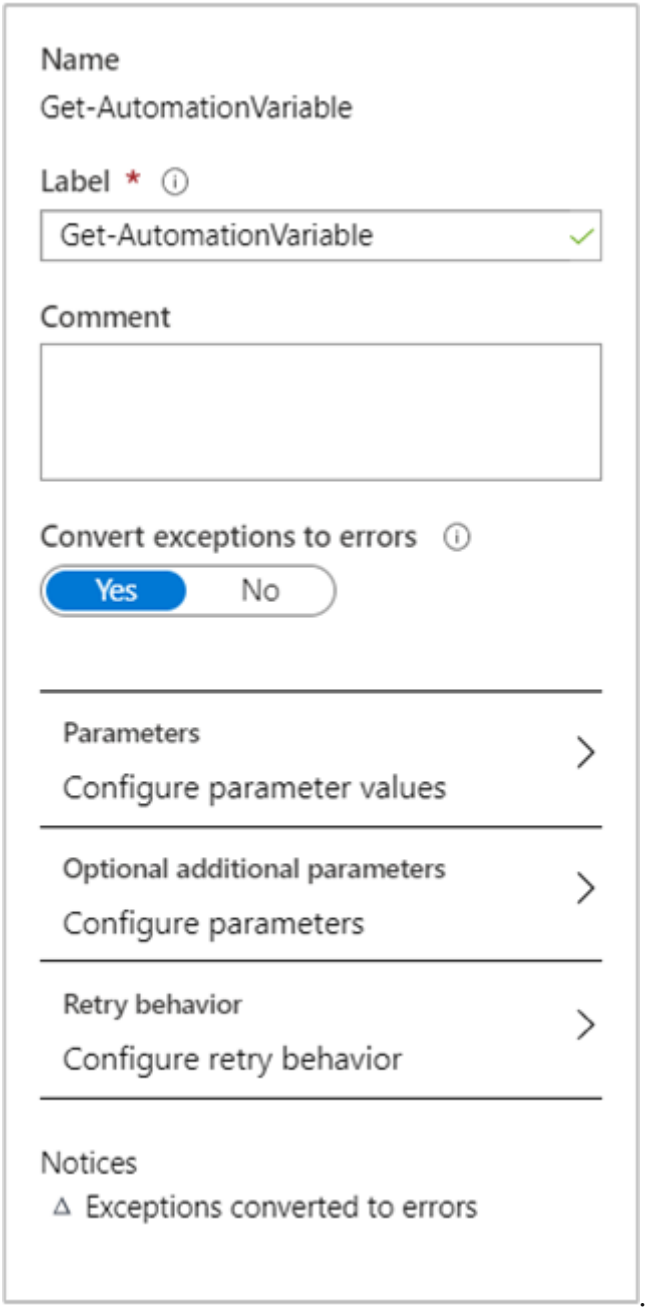
Después de habilitar la opción de configuración, haga que el runbook cree una actividad que controle el error. Si la actividad produce algún error, se seguirán los vínculos de error salientes. Los vínculos regulares no se seguirán, incluso si la actividad también produce la salida regular.



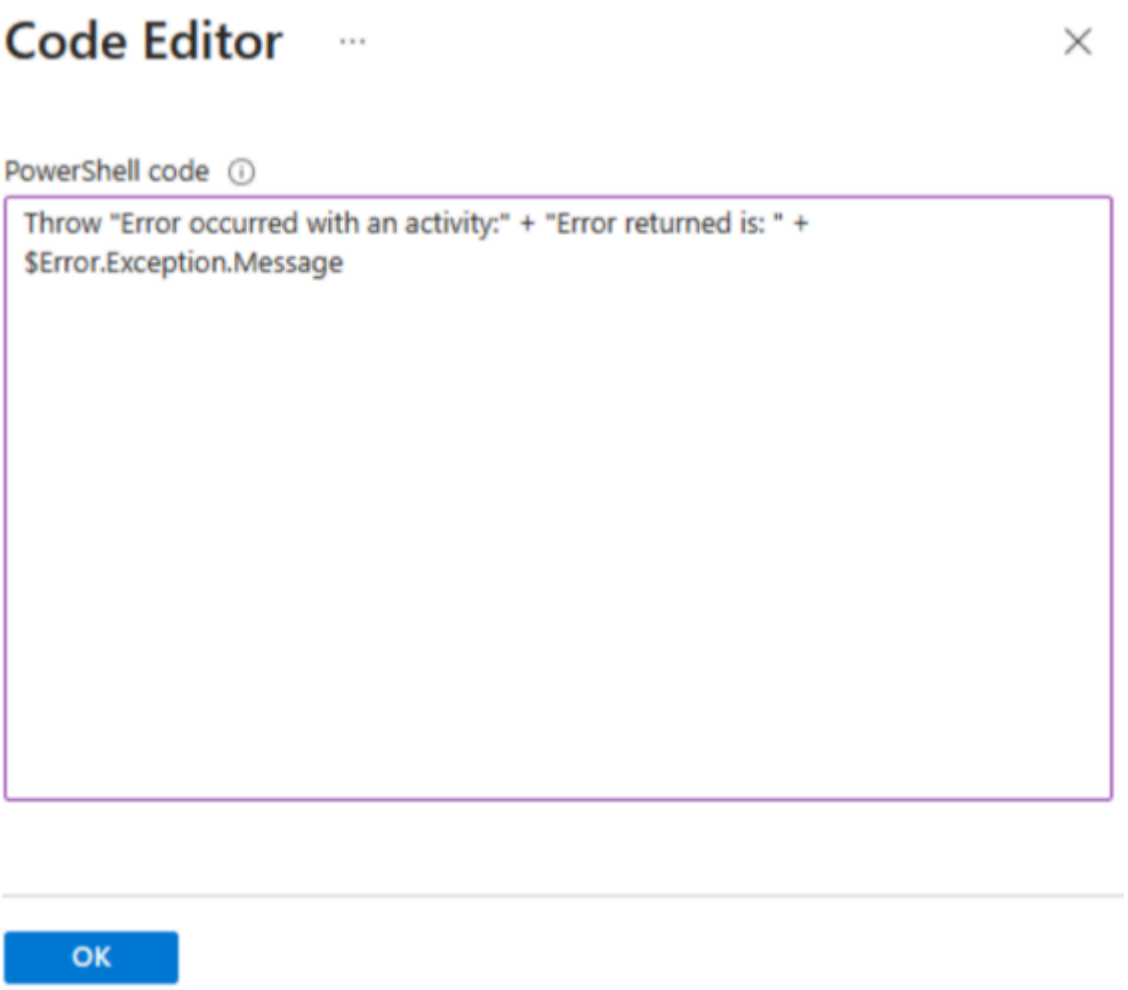
En el ejemplo siguiente, un runbook recupera una variable que contiene el nombre del equipo de una máquina virtual. A continuación, intenta iniciar la máquina virtual con la siguiente actividad.



La actividad `Get-AutomationVariable` y el cmdlet Start-AzVM están configurados para convertir excepciones en errores. Si hay problemas para obtener la variable o para iniciar la máquina virtual, el código genera errores.



Los vínculos de errores fluyen de estas actividades a una única actividad de código de `error management`. Esta actividad se configura con una expresión simple de PowerShell que utiliza la palabra clave `throw` para detener el procesamiento, junto con `$Error.Exception.Message` para obtener el mensaje que describe la excepción actual.



Pasos siguientes

- Para más información sobre cómo resolver errores de runbook gráficos, consulte Solución de problemas relativos a errores con runbooks.

Last updated on 17/11/2025

Reenvío de los registros de diagnóstico de Azure Automation a Azure Monitor

Azure Automation puede enviar el estado de un trabajo del runbook y de flujos de trabajos al área de trabajo de Log Analytics. Este proceso no implica la vinculación del área de trabajo, es completamente independiente y permite realizar investigaciones sencillas. Los registros de trabajo y los flujos de trabajo están visibles en Azure Portal, o con PowerShell para los trabajos individuales. Con registros de Azure Monitor para la cuenta de Automation, podrá hacer lo siguiente:

- Obtener información sobre el estado de los trabajos de Automation.
- Desencadenar un correo electrónico o una alerta en función del estado de trabajo del runbook (por ejemplo, error o en suspensión).
- Escribir consultas avanzadas en las transmisiones de trabajos.
- Correlacionar trabajos en cuentas de Automation.
- Usar vistas personalizadas y consultas de búsqueda para visualizar los resultados de runbook, el estado del trabajo de runbook y otras métricas o indicadores clave relacionados mediante un panel de Azure.
- Obtenga los registros de auditoría relacionados con las cuentas de Automation, runbooks y otras operaciones de creación, modificación y eliminación de recursos.

Mediante los registros de Azure Monitor, puede consolidar registros de distintos recursos en la misma área de trabajo, donde se pueden analizar con consultas para recuperar, consolidar y analizar rápidamente los datos recopilados. Puede crear y probar consultas mediante Log Analytics en Azure Portal y después analizar los datos directamente mediante estas herramientas o guardar las consultas para usarlas con la visualización o las reglas de alertas.

Azure Monitor usa una versión del lenguaje de consulta Kusto (KQL) que usa Azure Data Explorer y que es adecuada para consultas de registro sencillas. También incluye funciones avanzadas, como agregaciones, combinaciones y análisis inteligentes. Puede aprender rápidamente el lenguaje de consulta con diversas lecciones.

Configuración de diagnóstico de Azure Automation

Puede reenviar los registros de plataforma y datos de métricas siguientes mediante la compatibilidad con la configuración de diagnóstico de Automation:

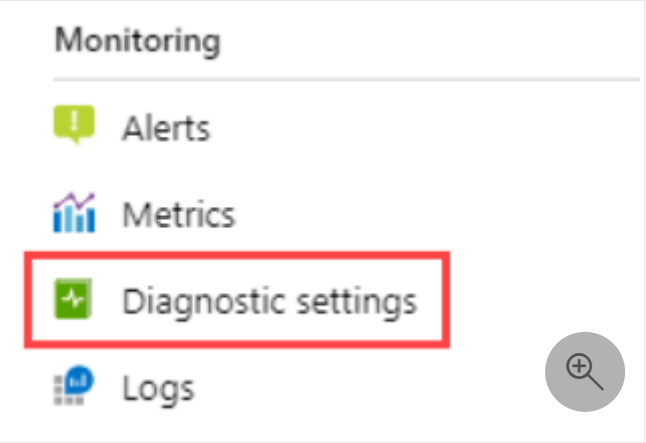
Expandir tabla

Tipos de datos	Descripción
Registros de trabajo	Estado del trabajo de runbook en la cuenta de Automation.
Flujos de trabajo	Estado de los flujos de trabajo en el runbook definido en la cuenta de Automation.
DSCNodeStatus	Estado del nodo DSC.
AuditEvent	Todos los registros de recursos que registran las interacciones de los clientes con los datos o la configuración del servicio Azure Automation.
Métricas	Total de trabajos, actualizaciones totales, ejecuciones de máquinas de implementación, ejecuciones de implementación de actualizaciones totales

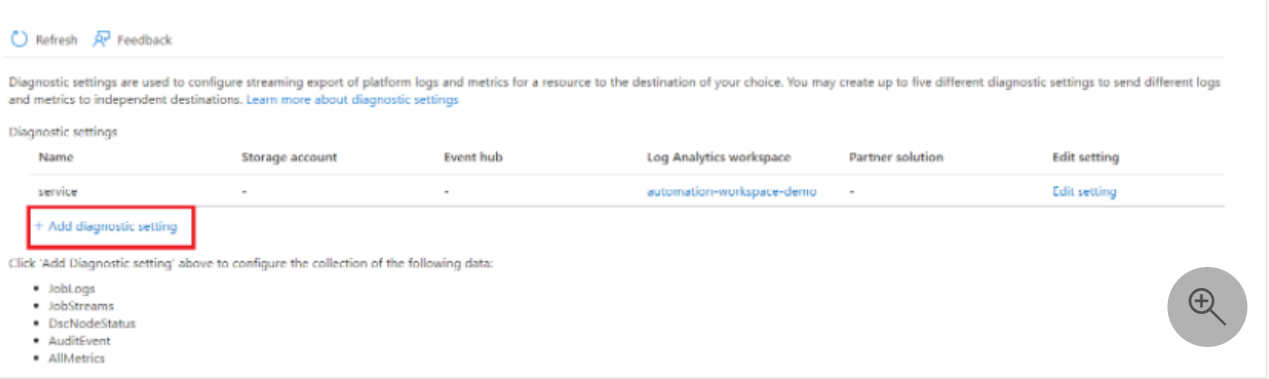
Configuración de los valores de diagnóstico en Azure Portal

Puede configurar los valores de diagnóstico en Azure Portal desde el menú del recurso de la cuenta de Automation.

1. En el menú de la cuenta de Automation, seleccione **Configuración de diagnóstico** en **Supervisión**.

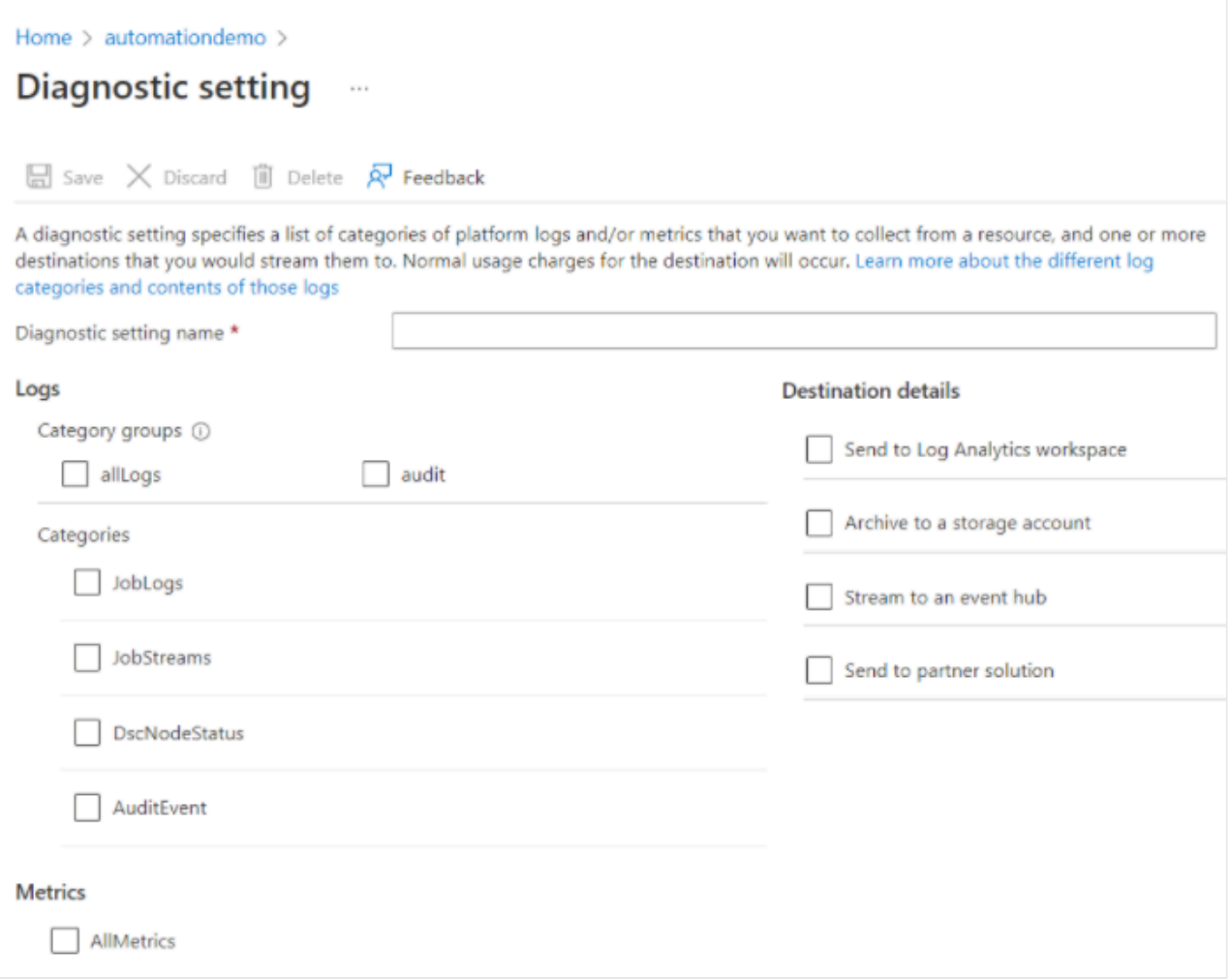


2. Haga clic en **Agregar configuración de diagnóstico**.



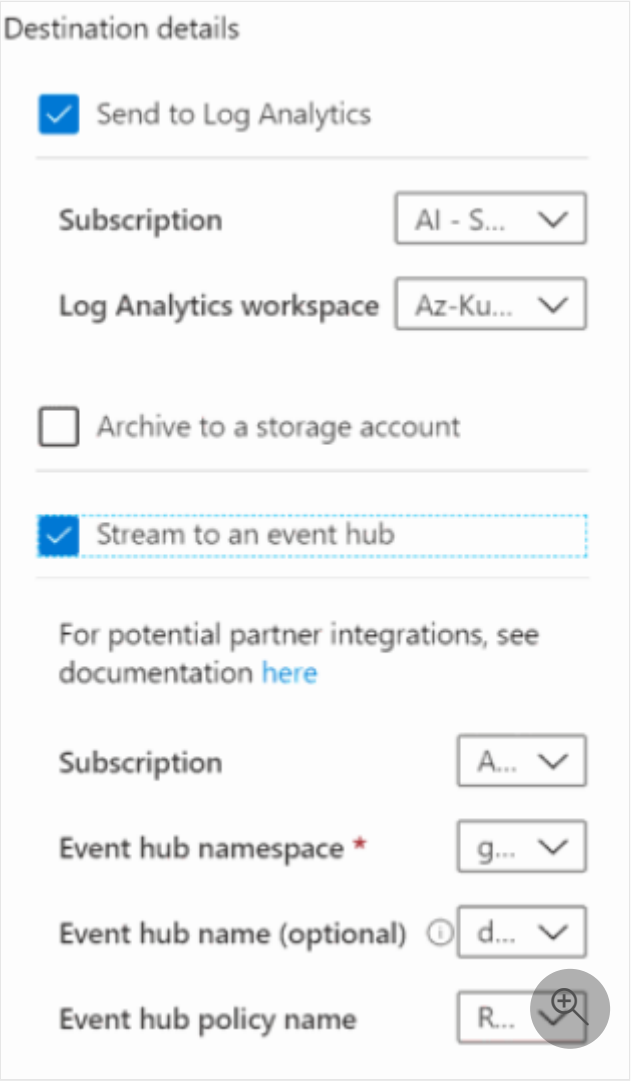
3. Escriba un nombre de configuración en el **Nombre de configuración de diagnóstico** si aún no tiene uno.

   También puede ver todas las categorías de registros y métricas.



- **Registros y métricas para enrutar**: en el caso de los registros, elija un grupo de categorías o seleccione las casillas de verificación individuales de cada categoría de datos que quiera enviar a los destinos especificados. Elija **AllMetrics** si quiere almacenar métricas en registros de Azure Monitor.

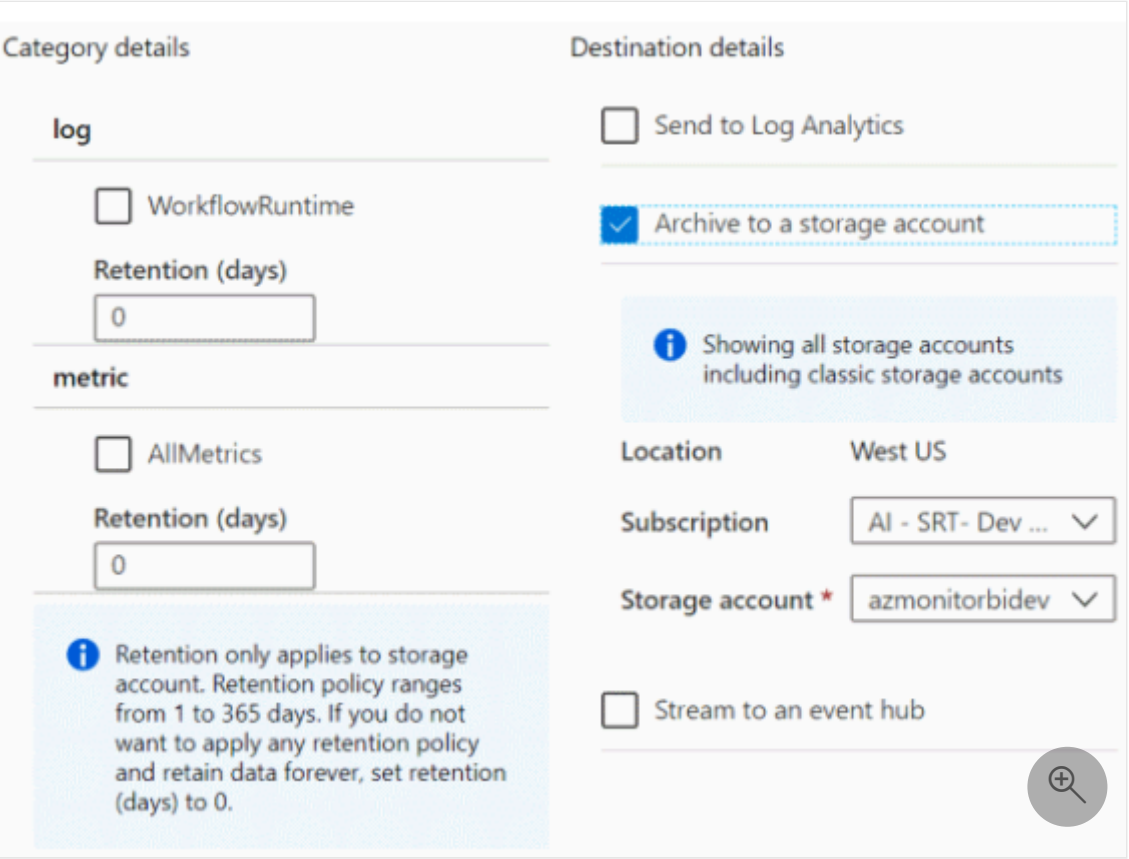
- **Detalles de destino**: seleccione la casilla de verificación para cada destino. Según la selección de cada cuadro, las opciones parecen permitirle agregar información adicional.



- **Log Analytics**: escriba el id. de suscripción y el nombre del área de trabajo. Si no tiene una área de trabajo, tendrá que crear una antes de continuar.

- **Event Hubs**: especifique los criterios siguientes:
  - Suscripción: la misma que la del centro de eventos.
  - Espacio de nombres del centro de eventos: cree un centro de eventos si aún no tiene uno.
  - Nombre del centro de eventos (opcional): si no especifica uno, se crea un centro de eventos para cada categoría de registro. Si va a enviar varias categorías, especifique un nombre para limitar el número de instancias de Event Hubs creadas. Consulte Cuotas y límites de Azure Event Hubs para más detalles.
  - Directiva del centro de eventos (opcional): una directiva define los permisos que tiene el mecanismo de streaming. Vea característica de Event Hubs.

- **Storage**: elija una suscripción, una cuenta de almacenamiento y una directiva de retención.



- **Integración de asociados**: primero debe instalar una integración de asociados en la suscripción. Las opciones de configuración variarán según el asociado. Para obtener más información, vea Integración de Azure Monitor.

4. Haga clic en **Save**(Guardar).

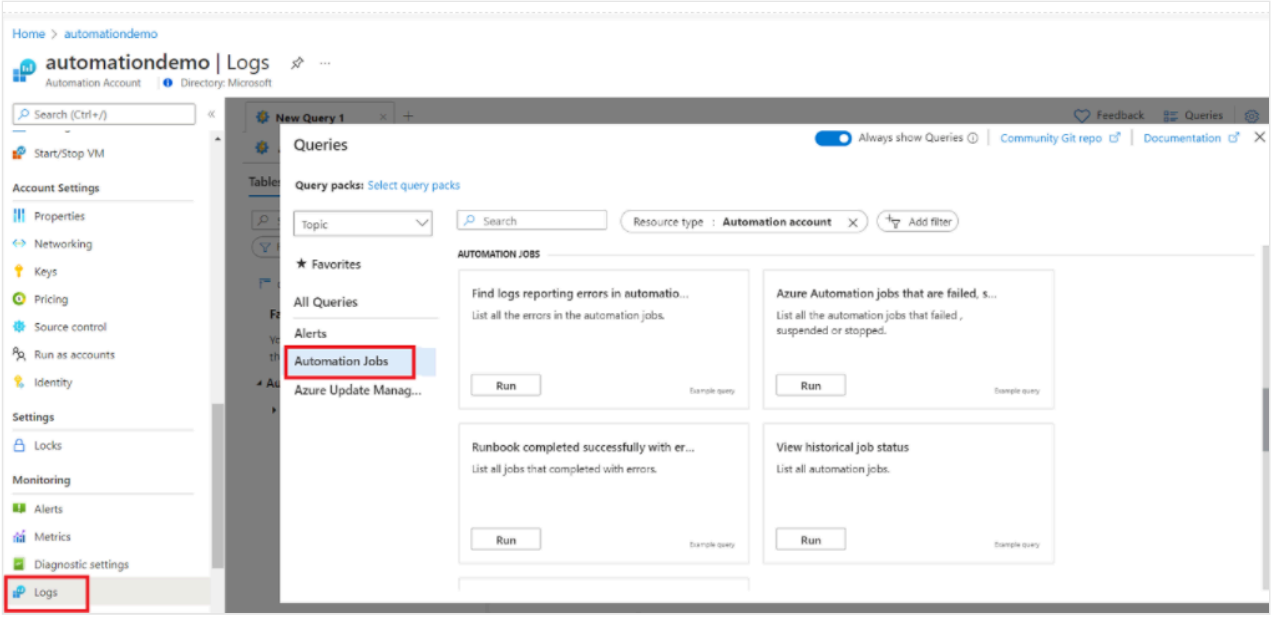
Transcurridos unos instantes, la nueva configuración aparece en la lista de configuraciones para este recurso y los registros se transmiten a los destinos especificados en cuanto se generan nuevos datos de eventos. Puede haber una diferencia de 15 minutos entre el evento emitido y su aparición en el área de trabajo de Log Analytics.

Consulta de los registros

Para consultar los registros generados:

1. En la cuenta de Automation, en **Supervisión**, seleccione **Registros**.

2. En **Todas las consultas**, seleccione **Automation Jobs** (Trabajos de Automation).

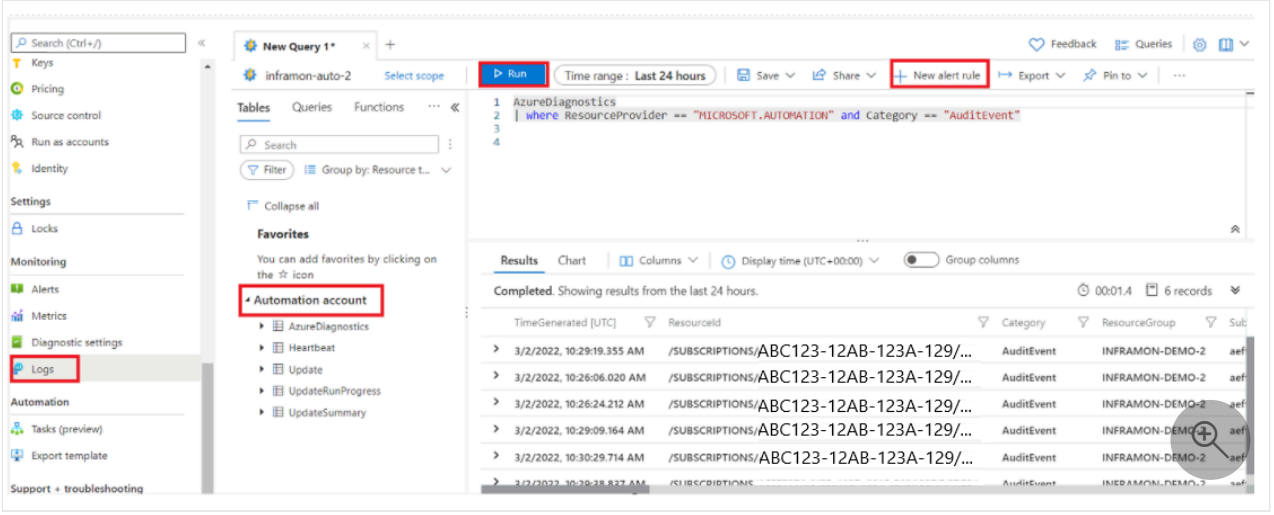


3. Seleccione una de las consultas que quiere ejecutar y haga clic en **Ejecutar**.

4. Para ejecutar una consulta personalizada, cierre la ventana **Consultas**, pegue la consulta personalizada en la nueva ventana de consulta y haga clic en **Ejecutar**.

   La salida de la consulta se muestra en el panel **Resultados**.

5. Haga clic en **Nueva regla de alertas** para configurar una alerta de Azure Monitor para esta consulta.



Registros de Azure Monitor

Los diagnósticos de Azure Automation crean los tipos de registros siguientes en los registros de Azure Monitor, etiquetados como `AzureDiagnostics`. Las tablas de las secciones siguientes son ejemplos de registros que genera Azure Automation y los tipos de datos que aparecen en los resultados de búsqueda de los registros.

Registros de trabajo

Expandir tabla

Propiedad	Descripción
TimeGenerated	Fecha y hora en que se ejecuta el trabajo de Runbook.
RunbookName_s	Nombre o nombres del runbook.
Caller_s	Autor de la llamada que inició la operación. Los valores posibles son una dirección de correo electrónico o el sistema para los trabajos programados.
Tenant_g	GUID (identificador único global) que identifica el inquilino para el autor de la llamada.
JobId_g	El GUID que identifica el trabajo de runbook.
ResultType	Estado del trabajo de runbook. Los valores posibles son: - Created Started Stopped Suspended Con error Completado
Category	Clasificación del tipo de datos. Para Automation, el valor será JobLogs.
OperationName	El tipo de operación realizada en Azure. En Automation, el valor es Job.
Resource	Nombre de la cuenta de Automation
SourceSystem	El sistema que utilizan los registros de Azure Monitor para recopilar los datos. En los diagnósticos de Azure, el valor siempre es "Azure".
ResultDescription	Estado de resultado del trabajo de runbook. Los valores posibles son: - Se inicia el trabajo - Error del trabajo - Trabajo completado
CorrelationId	GUID de correlación del trabajo de runbook.
ResourceId	Id. de recurso de la cuenta de Azure Automation del runbook.
SubscriptionId	GUID de la suscripción de Azure para la cuenta de Automation.
Grupo de recursos	Nombre del grupo de recursos de la cuenta de Automation.
ResourceProvider	Nombre del proveedor de recursos El valor es MICROSOFT.AUTOMATION.

Propiedad	Descripción
ResourceType	Tipo de recurso. El valor es AUTOMATIONACCOUNTS.

Transmisiones de trabajo

Expandir tabla

Propiedad	Descripción
TimeGenerated	Fecha y hora en que se ha ejecutado el trabajo de runbook.
RunbookName_s	Nombre del Runbook.
Caller_s	Autor de la llamada que inició la operación. Los valores posibles son una dirección de correo electrónico o el sistema para los trabajos programados.
StreamType_s	Tipo de flujo de trabajo. Los valores posibles son: progreso - Salida - Advertencia Error DEBUG - Detallado
Tenant_g	El GUID que identifica al inquilino del autor de la llamada.
JobId_g	El GUID que identifica el trabajo de runbook.
ResultType	El estado del trabajo de Runbook. Los valores posibles son: - In Progress
Category	Clasificación del tipo de datos. Para Automation, el valor es JobStreams.
OperationName	El tipo de operación realizada en Azure. En Automation, el valor es Job.
Resource	Nombre de la cuenta de Automation.
SourceSystem	El sistema que utilizan los registros de Azure Monitor para recopilar los datos. En los diagnósticos de Azure, el valor siempre es "Azure".
ResultDescription	Descripción que incluye el flujo de salida del runbook.
CorrelationId	GUID de correlación del trabajo de runbook.
ResourceId	Id. de recurso de la cuenta de Azure Automation del runbook.
SubscriptionId	GUID de la suscripción de Azure para la cuenta de Automation.
Grupo de	Nombre del grupo de recursos de la cuenta de Automation.

Propiedad	Descripción
recursos	
ResourceProvider	Proveedor de recursos. El valor es MICROSOFT.AUTOMATION.
ResourceType	Tipo de recurso. El valor es AUTOMATIONACCOUNTS.

> **Nota:**
>
> Asegúrese de que las credenciales no se envían a flujos de trabajo. El servicio quita las credenciales antes de mostrar flujos de trabajo en los registros de diagnóstico.

Eventos de auditoría

Expandir tabla

Propiedad	Descripción
TenantID	El GUID que identifica al inquilino del autor de la llamada.
TimeGenerated (UTC)	Fecha y hora en que se ejecuta el trabajo de runbook.
Category	AuditEvent
Grupo de recursos	Nombre del grupo de recursos de la cuenta de Automation.
Id. de suscripción	GUID de la suscripción de Azure para la cuenta de Automation.
ResourceProvider	MICROSOFT.AUTOMATION
Resource	Nombre de la cuenta de Automation
ResourceType	AUTOMATIONACCOUNTS
OperationName	Los valores posibles son Update, Create y Delete.
ResultType	Estado del trabajo de runbook. El valor posible es: Completed.
CorrelationId	GUID de correlación del trabajo de runbook.
ResultDescription	Estado de resultado del trabajo de runbook. Los valores posibles son Update, Create y Delete.
Tenant_g	El GUID que identifica al inquilino del autor de la llamada.
SourceSystem	Sistema que utilizan los registros de Azure Monitor para recopilar los datos. En los diagnósticos de Azure, el valor siempre es "Azure".

Propiedad	Descripción
clientInfo_IpAddress_s	{scrubbed}
clientInfo_PrincipalName_s	{scrubbed}
clientInfo_TenantId_g	Id. de inquilino del cliente.
clientInfo_Issuer_s	
clientInfo_ObjectId_g	Id. de objeto del cliente.
clientInfo_AppId_g	AppID del cliente.
clientInfo_ClientRequestId_g	RequestID del cliente.
targetResources_Resource_s	Cuenta, Trabajo, Credencial, Conexiones, Variables, Runbook.
Tipo	AzureDiagnostics
_ResourceId	Id. de recurso de la cuenta de Azure Automation del runbook.

Visualización de registros de Automation en registros de Azure Monitor

Ahora que ha iniciado el envío de los registros y los flujos de trabajo de Automation a los registros de Azure Monitor, veamos lo que puede hacer con ellos en los registros de Azure Monitor.

Para ver los registros, ejecute la consulta siguiente: `kusto AzureDiagnostics | where ResourceProvider == "MICROSOFT.AUTOMATION"`

Consultas de ejemplo para registros de trabajo y flujos de trabajo

Búsqueda de todos los trabajos completados con error

Además de los escenarios como las alertas al producirse errores, puede averiguar cuándo un trabajo de runbook tiene un error de no terminación. En estos casos, PowerShell genera un flujo de errores, aunque los errores de no terminación no causan la suspensión ni producen un error en el trabajo.

1. En el área de trabajo de Log Analytics, haga clic en **Registros**.
2. En el campo de consulta, escriba:

Kusto

```
AzureDiagnostics | where ResourceProvider == "MICROSOFT.AUTOMATION" and
Category == "JobStreams" and StreamType_s == "Error" | summarize
AggregatedValue = count () by JobId_g.
```

3. Haga clic en **Buscar**.

Visualización de las transmisiones de un trabajo

Cuando se depura un trabajo, también se pueden examinar sus transmisiones. La siguiente consulta muestra todos los flujos de un solo trabajo con GUID `2ebd22ea-e05e-4eb9-9d76-d73cbd4356e0`:

Kusto

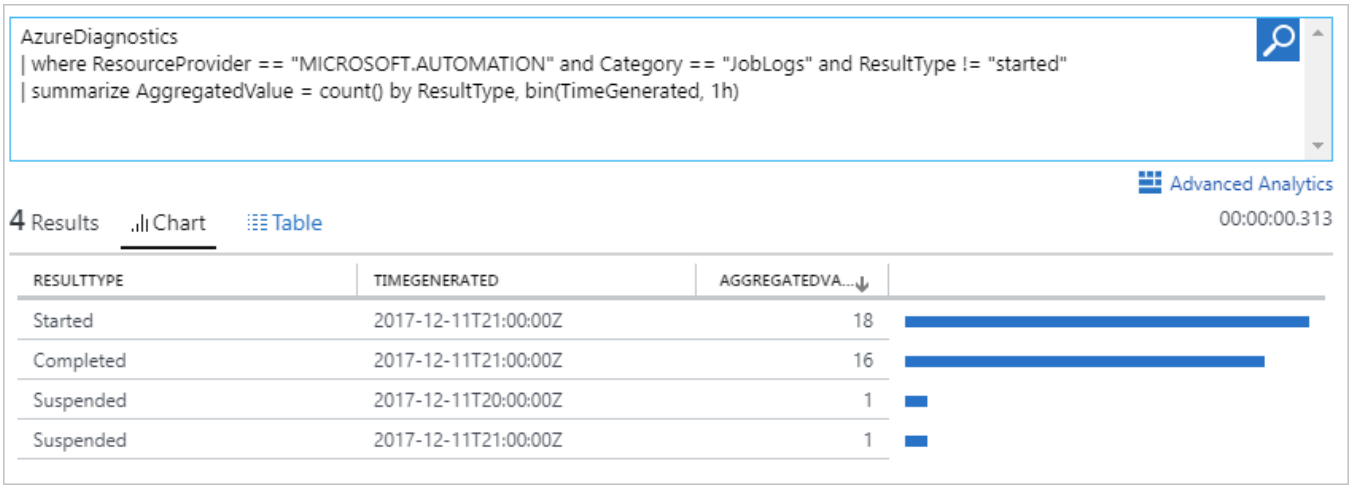
```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "JobStreams" and
JobId_g == "2ebd22ea-e05e-4eb9-9d76-d73cbd4356e0"
| sort by TimeGenerated asc
| project ResultDescription
```

Visualización del estado del historial de trabajos

Por último, puede que desee visualizar el historial de trabajos a lo largo del tiempo. Esta consulta se puede usar para buscar el estado de los trabajos con el paso del tiempo.

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "JobLogs" and
ResultType != "started"
| summarize AggregatedValue = count() by ResultType, bin(TimeGenerated, 1h)
```



Busque los registros que informan de errores en los trabajos de automatización.

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION"
| where StreamType_s == "Error"
| project TimeGenerated, Category, JobId_g, OperationName, RunbookName_s,
ResultDescription, _ResourceId
```

Búsqueda de trabajos completados de Azure Automation

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "JobLogs" and
ResultType == "Completed"
| project TimeGenerated, RunbookName_s, ResultType, _ResourceId, JobId_g
```

Búsqueda de trabajos de Azure Automation con errores, suspendidos o detenidos

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "JobLogs" and
(ResultType == "Failed" or ResultType == "Stopped" or ResultType == "Suspended")
| project TimeGenerated, RunbookName_s, ResultType, _ResourceId, JobId_g
```

Enumeración de todos los runbooks y trabajos que se han completado correctamente con errores

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "JobStreams" and
StreamType_s == "Error"
| project TimeGenerated, RunbookName_s, StreamType_s, _ResourceId,
ResultDescription, JobId_g
```

Envío de un correo electrónico cuando se produce un error en un trabajo del runbook o se suspende

En los pasos siguientes se explica cómo configurar alertas de correo electrónico en Azure Monitor para notificar cuando se produzca algún problema con un trabajo de runbook.

Para crear una regla de alerta, cree una búsqueda de registros del trabajo del runbook que deben invocar la alerta, tal como se describe en Consulta de los registros. Haga clic en **+Nueva regla de alertas** para configurar la regla de alertas.

1. En la cuenta de Automation, en **Supervisión**, seleccione **Registros**.

2. Cree una consulta de búsqueda de registros para la alerta; para ello, escriba un criterio de búsqueda en el campo de la consulta.

   Kusto

   ```
   AzureDiagnostics | where ResourceProvider == "MICROSOFT.AUTOMATION" and
   Category == "JobLogs" and (ResultType == "Failed" or ResultType == "Suspended")
   ```

   También puede agrupar por el nombre del runbook, para lo que debe usar:

   Kusto

   ```
   AzureDiagnostics | where ResourceProvider == "MICROSOFT.AUTOMATION" and
   Category == "JobLogs" and (ResultType == "Failed" or ResultType == "Suspended")
   | summarize AggregatedValue = count() by RunbookName_s
   ```

3. Para abrir la pantalla **Crear regla de alertas**, haga clic en **+Nueva regla de alertas** en la parte superior de la página. Para obtener más información sobre las opciones a fin de configurar las alertas, vea Alertas de registro en Azure.

Registros de auditoría de diagnóstico de Azure Automation

Ahora puede enviar registros de auditoría también al área de trabajo de Azure Monitor. Esto permite a las empresas supervisar las actividades clave de la cuenta de automatización para garantizar la seguridad y el cumplimiento. Cuando se habilita mediante la configuración de Azure Diagnostics, podrá recopilar telemetría sobre las operaciones de creación, actualización y eliminación de runbooks de Automation, trabajos y recursos de automatización, como la conexión, las credenciales, la variable y el certificado. También puede configurar las alertas para las condiciones del registro de auditoría como parte de los requisitos de supervisión de seguridad.

Diferencia entre los registros de actividad y los registros de auditoría

El registro de actividad es un registro de plataforma de Azure que proporciona información sobre los eventos del nivel de suscripción. El registro de actividad de la cuenta de Automation incluye información sobre cuándo se modifica, se crea o elimina un recurso de automatización. Pero no captura el nombre ni el id. del recurso.

Los registros de auditoría de las cuentas de Automation capturan el nombre y el id. del recurso, como la variable de automatización, las credenciales, la conexión, entre otros, junto con el tipo de operación realizada para el recurso, y Azure Automation limpiaría algunos detalles, como los datos IP de cliente que se ajustan al cumplimiento del RGPD.

Los registros de actividad mostrarían detalles como la dirección IP del cliente porque un registro de actividad es un registro de plataforma que proporciona información detallada de diagnóstico y auditoría para los recursos de Azure. Se generan automáticamente para las actividades que se producen en ARM y se insertan en el proveedor de recursos del registro de actividad. Puesto que los registros de actividad forman parte de la supervisión de Azure, mostraría algunos datos de cliente para proporcionar información sobre la actividad de este.

Consultas de ejemplo para registros de auditoría

Consulta para ver registros de auditoría de recursos de Automation

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "AuditEvent"
```

Consulta para supervisar cualquier operación de actualización, creación o eliminación de variables

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "AuditEvent" and
targetResources_Resource_s == "Variable"
```

Consulta para supervisar cualquier operación de runbook, como de creación, redacción o actualización

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "AuditEvent" and
targetResources_Resource_s contains "Runbook"
```

Consulta para supervisar la creación, actualización o eliminación de certificados

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "AuditEvent" and
targetResources_Resource_s contains "Certificate"
```

Consulta para supervisar la creación, actualización o eliminación de credenciales

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION" and Category == "AuditEvent" and
targetResources_Resource_s contains "Credential"
```

Filtrado de la salida de estado del trabajo convertida en un objeto JSON

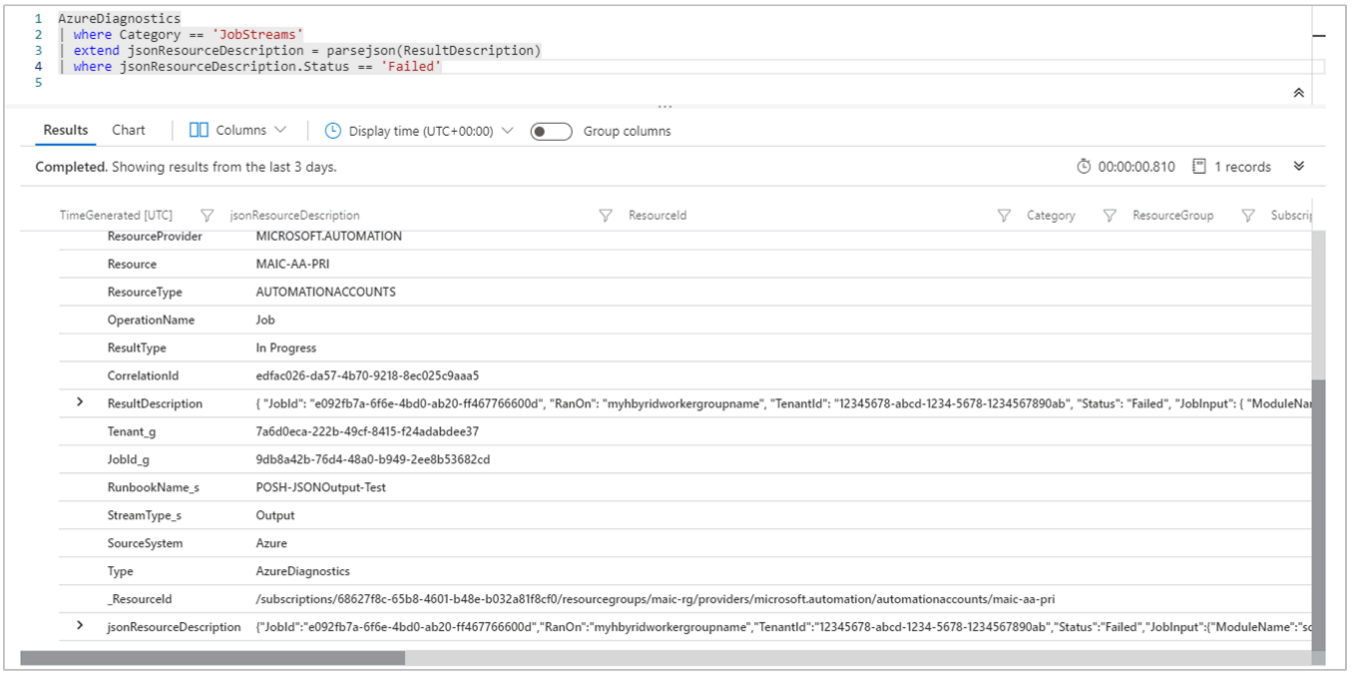
Recientemente hemos cambiado el comportamiento de la escritura de los datos de registros de Automation en la tabla `AzureDiagnostics` del servicio Log Analytics, donde ya no se desglosan las propiedades JSON en campos independientes. Si configuró el runbook para dar formato a los objetos del flujo de salida en formato JSON como columnas independientes, es necesario volver a configurar las consultas para analizar ese campo en un objeto JSON con el fin de acceder a esas propiedades. Esto se logra mediante parse json para acceder a un elemento JSON específico en una ruta de acceso conocida.

Por ejemplo, un runbook da formato a la propiedad *ResultDescription* en el flujo de salida en formato JSON con varios campos. Para buscar el estado de los trabajos que se encuentran en

estado de error, tal y como se especifica en un campo denominado **Estado**, use esta consulta de ejemplo para buscar el objeto *ResultDescription* con el estado **Error**:

Kusto

```
AzureDiagnostics
| where Category == 'JobStreams'
| extend jsonResourceDescription = parse_json(ResultDescription)
| where jsonResourceDescription.Status == 'Failed'
```



Pasos siguientes

- Para aprender a crear consultas de búsqueda y a revisar los registros de trabajos de Automation con registros de Azure Monitor, consulte Introducción a las consultas de registro en Azure Monitor.
- Para comprender la creación y recuperación de mensajes de error y de salida de runbooks, consulte Salidas de runbook y mensajes en Azure Automation.
- Para más información sobre la ejecución de runbooks, la supervisión de trabajos de runbook y otros detalles técnicos, consulte Ejecución de un runbook en Azure Automation.
- Para obtener más información sobre los registros de Azure Monitor y los orígenes de recopilación de datos, consulte Introducción a la recopilación de datos de almacenamiento en los registros de Azure Monitor.
- Si necesita ayuda con la solución de problemas de análisis de registros, consulte el artículo sobre por qué el análisis de registros ya no recopila datos.

Last updated on 17/11/2025

Desplegar un Trabajador de Runbook Híbrido para Usuarios de Windows o Linux basado en extensiones en Azure Automation

La incorporación basada en una extensión es solo para instancias de Hybrid Runbook Worker de **usuario**. En este artículo se describe cómo: implementar un trabajador híbrido de Runbook en una máquina Windows o Linux, quitar el trabajador y quitar un grupo de trabajadores híbridos de Runbook.

Para la incorporación de una instancia de Hybrid Runbook Worker de **sistema**, consulte Implementación de Hybrid Runbook Worker en Windows basado en agente en Automation o Implementación de Hybrid Runbook Worker en Linux basado en agente en Automation.

Puede usar la característica Hybrid Runbook Worker de usuario de Azure Automation para ejecutar runbooks directamente en una máquina de Azure o que no sea de Azure, incluidos servidores habilitados para Azure Arc, VMware vSphere habilitado para Arc, y SCVMM habilitado para Arc. Desde el equipo o servidor que hospeda el rol, puede ejecutar runbooks directamente en él y en recursos del entorno para administrar esos recursos locales. Azure Automation almacena y administra runbooks y, a continuación, los entrega a una o varias máquinas elegidas. Después de implementar correctamente un trabajo de runbook, revise la ejecución de runbooks en Hybrid Runbook Worker para más información sobre cómo configurar los runbooks para automatizar los procesos del centro de datos local o en otro entorno de nube.

> ⓘ **Nota:**
>
> Una instancia de Hybrid Worker puede coexistir con ambas plataformas: **basada en agente (V1)** y **basada en extensión (V2)**. Si opta por la opción Basada en extensión (V2) en una instancia de Hybrid Worker que ya se ejecuta Basada en agente (V1), podría ver dos entradas de la instancia de Hybrid Runbook Worker en el grupo. Una basada en la extensión de la plataforma (V2) y otra basada en el agente (V1). **Más información**.

Prerrequisitos

Requisitos mínimos de la máquina

- Dos núcleos

- 4 GB de RAM
- **Non-Azure machines** deben tener instalado el agente Azure Connected Machine. Para instalar el `AzureConnectedMachineAgent`, vea Conexión de máquinas híbridas a Azure desde Azure Portal para servidores habilitados para Arc. Vea Instalación del agente de Arc para máquinas virtuales de VMware habilitadas para Arc para habilitar la administración de invitados para máquinas virtuales VMware vSphere habilitadas para Arc e instalar agente de Arc para SCVMM habilitado para Arc para habilitar la administración de invitados para máquinas virtuales SCVMM habilitadas para Arc.
- La identidad administrada asignada por el sistema debe estar habilitada en la máquina virtual Azure, el servidor habilitado para Arc, la máquina virtual VMware vSphere habilitada para Arc o la máquina virtual SCVMM habilitada para Arc. Si la identidad administrada asignada por el sistema no está habilitada, se habilitará como parte del proceso de adición.

Sistemas operativos compatibles

> **Nota:**
>
> Windows Server 2012 y 2012 R2 alcanzaron el final del soporte (EOS). Revise el uso y planee las actualizaciones y migraciones del sistema operativo en consecuencia. Para obtener más información, consulte Fin del soporte técnico para:
>
> - Windows Server 2012
> - Windows Server 2012 R2
>
> Actualización local a Windows Server 2016, 2019, 2022 o 2025.

Expandir tabla

Windows (x64)	Linux (x64)
● Windows Server 2022 (incluido Server Core) ● Windows Server 2019 (incluido Server Core) ● Windows Server 2016, versión 1709 y 1803 (excepto Server Core) ● Windows Server 2012, 2012 R2 (excepto Server Core) ● Windows 10 Enterprise (incluida la multisesión) y Pro ● Windows 11 Enterprise (incluida la multisesión) y Pro	● Debian GNU/Linux 8, 9, 10 y 11 ● Ubuntu 18.04 LTS, 20.04 LTS y 22.04 LTS ● SUSE Linux Enterprise Server 15.2, 15.3, 15.4, 15.5 y 15.6 ● Red Hat Enterprise Linux Server 7, 8 y 9 ● Rocky Linux 9 ● Oracle Linux 7, 8 y 9 *La extensión de Hybrid Worker seguiría las escalas de tiempo de soporte técnico del proveedor del sistema operativo.* Python versión 3.11+ no son compatibles con Hybrid Runbook Worker de Linux.

Otros requisitos

Expandir tabla

Windows (x64)	Linux (x64)
Windows PowerShell 5.1 (descargue WMF 5.1). PowerShell Core no es compatible.	La protección de Linux no debe estar habilitada.
.NET Framework 4.6.2 o posterior.	

Requisitos de paquetes para Linux

Expandir tabla

Paquete requerido	Descripción	Versión mínima
Glibc	Biblioteca GNU C	2.5-12
OpenSSL	Bibliotecas OpenSSL	1.0 (se admiten TLS 1.1 y TLS 1.2)
Curl	Cliente web de cURL	7.15.5
Python-ctypes	Biblioteca de funciones externas para Python	se requieren Python 2.x o Python 3.x
PAM	Módulos de autenticación conectables	

Expandir tabla

Paquete opcional	Descripción	Versión mínima
PowerShell Core	Para ejecutar runbooks de PowerShell, es necesario instalar PowerShell Core. Para obtener instrucciones, consulte Instalación de PowerShell Core en Linux.	6.0.0

Nota:

- Hybrid Runbook Worker no se admite actualmente para Virtual Machine Scale Sets (VMSS).
- Se recomienda encarecidamente que nunca configure la extensión de Hybrid Worker en una máquina virtual que hospeda el controlador de dominio. Los procedimientos recomendados de seguridad no aconsejan esta configuración debido a la naturaleza de alto riesgo de exponer controladores de dominio a posibles vectores de ataque a través de trabajos de Azure Automation. Los controladores de dominio deben estar altamente protegidos y aislados de servicios no esenciales para evitar el acceso no autorizado y mantener la integridad del entorno de Active Directory Domain Services (ADDS).

Permisos para credenciales de trabajador híbrido

Si Hybrid Worker basado en extensiones usa credenciales personalizadas de Hybrid Worker, asegúrese de que los siguientes permisos de carpeta se asignan al usuario personalizado para evitar que los trabajos se suspendan.

Expandir tabla

Tipo de recurso	Permisos de carpeta
máquina virtual de Azure	C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows (lectura y ejecución)
Servidor habilitado para Arc	C:\ProgramData\AzureConnectedMachineAgent\Tokens (lectura) C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows (leer y ejecutar).

> **Nota:**
>
> - Cuando un sistema tiene UAC/LUA en su lugar, se deben conceder permisos directamente y no a través de ninguna pertenencia a grupos. Más información.
> - Debido a una limitación actual, estos permisos de carpeta se quitan de la carpeta C:\ProgramData\AzureConnectedMachineAgent\Tokens en máquinas habilitadas para Azure Arc cuando se actualiza el agente de Azure Connected Machine. La resolución actual consiste en volver a aplicar estos permisos a la carpeta. Más información.

Dirección URL del servicio híbrido de la cuenta de Automation

Debe recuperar y usar el "AutomationHybridServiceURL" para implementar la extensión Hybrid Worker en la máquina virtual o Arc.

> **Nota:**
>
> La dirección URL necesaria es automationHybridServiceUrl, NO registrationUrl.

Hay varias maneras de recuperar el valor de AutomationHybridServiceUrl:

- Cópielo desde el portal **Azure**, **Automation Account**, **Properties**, **Automation hybrid service URL**.

O

- Cópielo desde **Azure Portal**, **Cuenta de Automation**, **Información general**, **JSON**. Seleccione la versión más reciente de la API; de lo contrario, es posible que AutomationHybridServiceUrl no se muestre.

  O

- La siguiente llamada a la API REST:

  rest

  ```rest
  GET
  https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Automation/automationAccounts/{automationAccountName}?api-version=2023-11-01
  ```

Requisitos de red

Uso del servidor proxy

Si utiliza un servidor proxy para la comunicación entre Azure Automation y las máquinas que ejecutan el Hybrid Runbook Worker basado en extensiones, asegúrese de que los recursos apropiados sean accesibles. El tiempo de espera de las solicitudes de Hybrid Runbook Worker y los servicios de Automation es de 30 segundos. Después de tres intentos, se produce un error en una solicitud.

> ⓘ **Nota:**
>
> Para máquinas virtuales de Azure y servidores con compatibilidad Arc, puede configurar la configuración del proxy mediante cmdlets de PowerShell o API. Actualmente no se admite para máquinas virtuales VMware vSphere compatibles con Arc.

Para instalar la extensión mediante cmdlets:

1. Obtenga los detalles de la cuenta de automatización utilizando la siguiente llamada API.

   HTTP

   ```http
   GET
   https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.Automation/automationAccounts/{automationAccountName}?api-version=2023-11-01
   ```

La llamada API proporcionará el valor con la clave `AutomationHybridServiceUrl`. Use la dirección URL en el paso siguiente para habilitar la extensión en la máquina virtual.

2. Instale la extensión de Hybrid Worker en la máquina virtual mediante la ejecución del siguiente cmdlet de PowerShell (módulo requerido: Az.Compute). Utilice el valor de `properties.automationHybridServiceUrl` proporcionado por la llamada a la API anterior. Asegúrese de que la dirección URL de ProxyServer no contiene espacios en blanco.

Windows

Azure PowerShell

```
$settings = @{
    "AutomationAccountURL"  = "<automationHybridServiceUrl>";
    "ProxySettings" = @{
        "ProxyServer" = "<ipaddress>:<port>";
        "UserName"="test";
    }
};
$protectedsettings = @{
"ProxyPassword" = "password";
};
```

Máquinas Virtuales de Azure

PowerShell

```
Set-AzVMExtension -ResourceGroupName <VMResourceGroupName> -Location <VMLocation>
-VMName <VMName> -Name "HybridWorkerExtension" -Publisher
"Microsoft.Azure.Automation.HybridWorker" -ExtensionType HybridWorkerForWindows -
TypeHandlerVersion 1.1 -Settings $settings -ProtectedSettings $protectedsettings
-EnableAutomaticUpgrade $true/$false
```

Máquinas virtuales habilitadas por Azure Arc

PowerShell

```
New-AzConnectedMachineExtension -ResourceGroupName <VMResourceGroupName> -
Location <VMLocation> -MachineName <VMName> -Name "HybridWorkerExtension" -
Publisher "Microsoft.Azure.Automation.HybridWorker" -ExtensionType
HybridWorkerForWindows -TypeHandlerVersion 1.1 -Setting $settings -
ProtectedSetting $protectedsettings -NoWait -EnableAutomaticUpgrade
```

Uso del firewall

Si usa un firewall para restringir el acceso a Internet, tendrá que configurarlo para que permita el acceso. Los siguientes puertos y direcciones URL son necesarios para Hybrid Runbook Worker y

para Automation State Configuration para comunicarse con Azure Automation.

Expandir tabla

Propiedad	Descripción
Puerto	443 para el acceso a Internet de salida
Dirección URL global	*.azure-automation.net
Dirección URL global de US Gov Virginia	*.azure-automation.us

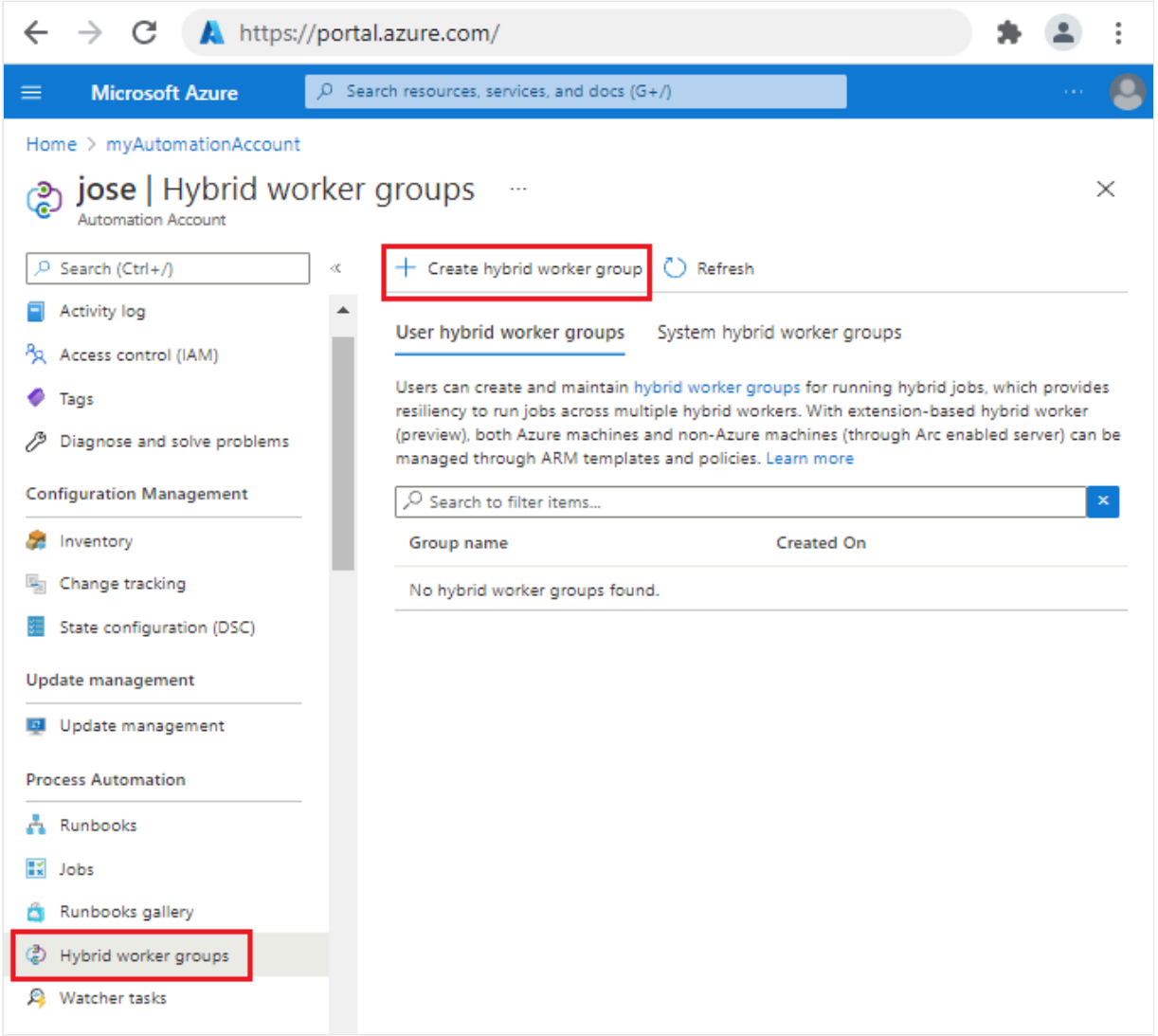
Límite de cuota de CPU

Hay un límite de cuota de CPU del 25% al configurar un Hybrid Runbook Worker basado en extensiones. No hay ningún límite de este tipo para Windows Hybrid Runbook Worker.

Crear grupo de trabajadores híbridos

Para crear un grupo de Hybrid Worker en el portal de Azure, siga estos pasos:

1. Inicie sesión en el portal Azure.
2. Vaya a su cuenta de Automation.
3. En **Automatización de procesos**, seleccione **Grupos de Hybrid Worker**.
4. Seleccione **+ Crear grupo de Trabajador Híbrido**.

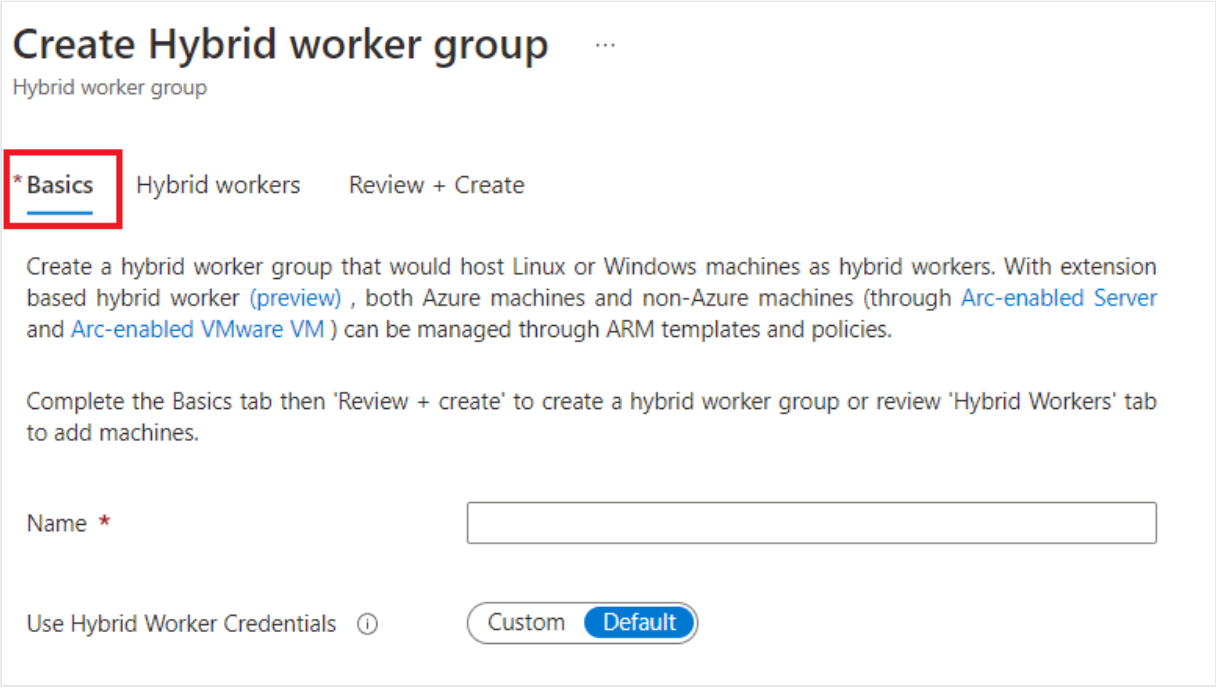


5. En la pestaña **Aspectos básicos**, en el cuadro de texto **Nombre**, escriba un nombre para su grupo de Hybrid Worker.

6. Para la opción **Use Hybrid Worker Credentials** (Usar credenciales de Hybrid Worker):

    - Si selecciona **Default** (Predeterminado), la extensión híbrida se instalará con la cuenta del sistema local.
    - Si selecciona **Custom** (Personalizado), seleccione el recurso de credencial en la lista desplegable.

7. Seleccione **Siguiente** para avanzar a la pestaña **Trabajadores híbridos**. Puede seleccionar máquinas virtuales de Azure, servidores habilitados para Azure Arc, VMware vSphere habilitado para Azure Arc y SCVMM habilitado para Azure Arc para agregarlos a este grupo de trabajadores híbridos. Si no selecciona ninguna máquina, se creará un grupo de Hybrid Worker vacío. Todavía puede agregar máquinas más adelante.



8. Seleccione **Agregar máquinas** para ir a la página **Agregar máquinas como trabajador híbrido**. Solo verá las máquinas que no forman parte de ningún otro grupo de Hybrid Worker.

9. Active la casilla situada junto a las máquinas que desea agregar al grupo de Hybrid Worker. Si no ve su máquina externa a Azure listada, asegúrese de que el agente de Azure Arc Connected Machine esté instalado en la máquina.

10. Selecciona **Agregar**.

11. Seleccione **Siguiente** para ir a la pestaña **Revisar y crear**.

12. Selecciona **Crear**.

    La extensión del trabajador híbrido se instala en la máquina y el trabajador híbrido se registra en el grupo de trabajadores híbridos. La adición del Hybrid Worker al grupo se produce inmediatamente, mientras que la instalación de la extensión puede tardar unos minutos. Seleccione **Actualizar** para ver el nuevo grupo. Seleccione el nombre del grupo para ver los detalles del Hybrid Worker.

> ⓘ **Nota:**
>
> Una máquina seleccionada no se agregará a un grupo de Hybrid Worker si ya forma parte de otro grupo de Hybrid Worker.

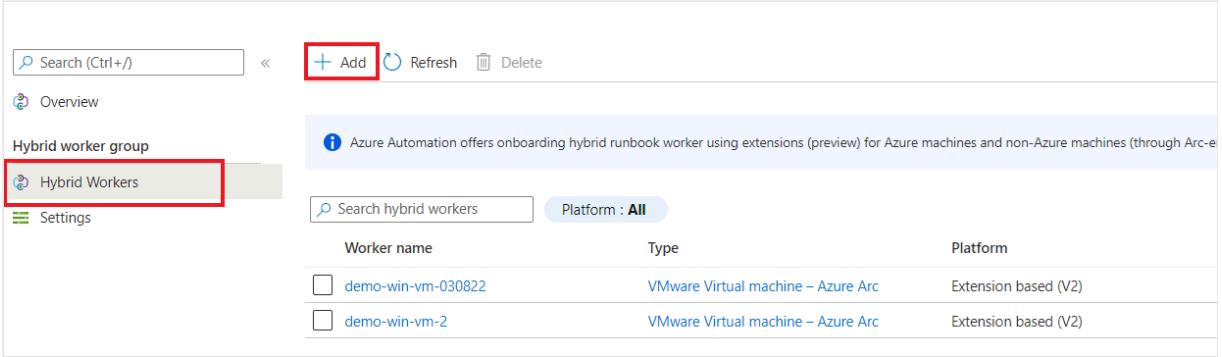
Añadir una máquina a un grupo de Trabajadores Híbridos

También puede agregar máquinas a un grupo de Hybrid Worker existente.

1. En **Automatización de** procesos, seleccione **Grupos de Hybrid Worker** y, a continuación, el grupo de Hybrid Worker existente para ir a la página **Grupo de Hybrid Worker** .

2. En **Grupo de Hybrid Worker**, seleccione **Hybrid Workers**.

3. Seleccione **+ Agregar** para ir a la página **Agregar máquinas como Trabajador Híbrido**. Solo verá las máquinas que no forman parte de ningún otro grupo de Hybrid Worker.

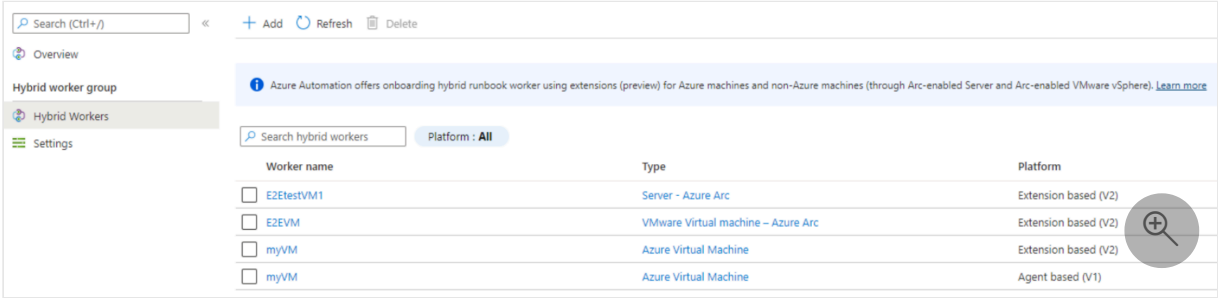


4. Active la casilla situada junto a las máquinas que desea agregar al grupo de Hybrid Worker.

   Si no ve su máquina externa a Azure listada, asegúrese de que el agente de Azure Arc Connected Machine esté instalado en la máquina. Para instalar el `AzureConnectedMachineAgent` vea Conexión de máquinas híbridas a Azure desde Azure Portal para servidores habilitados para Arc. Vea Instalación del agente de Arc para máquinas virtuales habilitadas para Arc para habilitar la administración de invitados para VMware vSphere habilitado para Arc y Instalación del agente de Arc para SCVMM habilitado para Arc para habilitar la administración de invitados para máquinas virtuales SCVMM habilitadas para Arc.

5. Seleccione **Agregar** para agregar la máquina al grupo.

   Después de agregar, puede ver el tipo de máquina como máquina virtual Azure, Máquina – Azure Arc, Máquina – Azure Arc (VMware) o Máquina – Azure Arc SCVMM. El campo **Plataforma** muestra al trabajador como **basado en agentes (V1)** o **basado en extensiones (V2)**.



Migración de un agente existente a Hybrid Worker basado en extensiones

Para utilizar las ventajas de los Trabajadores Híbridos basados en extensiones, debe migrar todos los Trabajadores Híbridos de usuario basados en agente existentes a Trabajadores basados en extensiones. Una máquina de trabajo híbrida puede coexistir en ambas plataformas: **basada en agente (V1)** y **basada en extensión (V2)**. La instalación basada en la extensión no afecta a la instalación o administración de una instancia de Worker basada en agente.

Para instalar la extensión de Hybrid Worker en una instancia de Hybrid Worker basada en agente existente, asegúrese de que se cumplen los requisitos previos antes de seguir estos pasos:

1. En **Automatización de procesos**, seleccione **grupos de trabajadores híbridos** y luego seleccione su grupo de trabajadores híbridos existente para ir a la página de **grupos de trabajadores híbridos**.
2. En **Grupo de Hybrid Worker**, seleccione **Hybrid Workers** > **+ Agregar** para ir a la página **Agregar máquinas como Hybrid Worker**.
3. Active la casilla junto a la instancia de Hybrid Worker existente basada en agente (V1).
4. Seleccione **Agregar** para agregar la máquina al grupo.

La columna Plataforma muestra la instancia de Hybrid Worker como **Basada en agente (V1)** o **Basada en extensión (V2)**. Una vez que se sienta cómodo con la experiencia y el uso de Hybrid Worker basado en una extensión, puede quitar la instancia de Worker basada en agente.

Para la migración a gran escala de varias instancias de Hybrid Worker basadas en agente, también puede usar otros canales, como : Bicep, plantillas de ARM, cmdlets de PowerShell, API REST y la CLI de Azure.

Eliminación de una instancia de Hybrid Runbook Worker

Puede eliminar Hybrid Runbook Worker desde el portal.

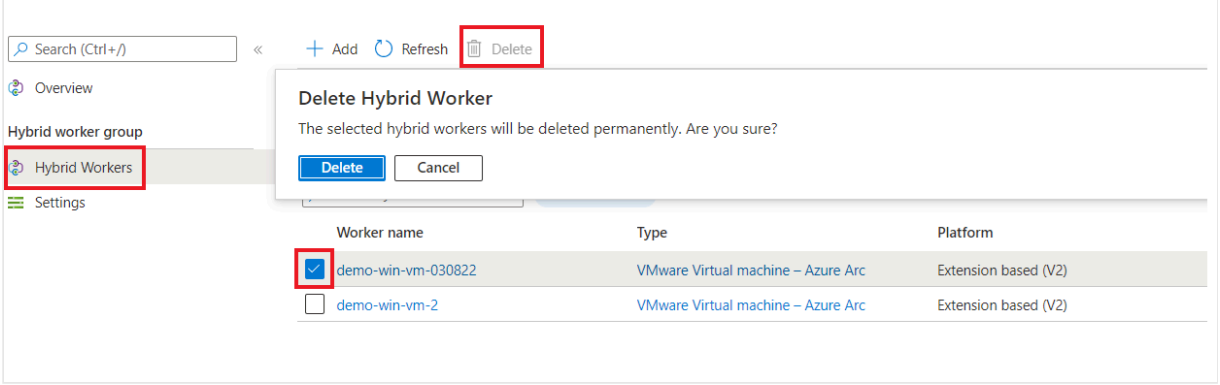
1. En **Automatización de procesos**, seleccione **Grupos de Hybrid Worker** y, a continuación, el grupo de Hybrid Worker para ir a la página **Grupo de Hybrid Worker**.

2. En **Grupo de Hybrid Worker**, seleccione **Hybrid Workers**.

3. Active la casilla situada junto a las máquinas que desea eliminar del grupo de Hybrid Worker.

4. Seleccione **Eliminar**.

Aparecerá una advertencia en un cuadro de diálogo **Eliminar Hybrid Worker** que la instancia de Hybrid Worker seleccionada se eliminaría permanentemente. Seleccione **Eliminar**. Esta operación eliminará la extensión del trabajo **basado en extensión (V2)** o quitará la entrada **basada en agente (V1)** del portal. Sin embargo, deja al trabajador híbrido obsoleto en la máquina virtual. Para desinstalar manualmente el agente, consulte Desinstalación del agente.



> **Nota:**
>
> - Una instancia de Hybrid Worker puede coexistir con ambas plataformas: **basada en agente (V1)** y **basada en extensión (V2)**. Si opta por la opción **Basada en extensión (V2)** en una instancia de Hybrid Worker que ya se ejecuta **Basada en agente (V1)**, podría ver dos entradas de la instancia de Hybrid Runbook Worker en el grupo. una con la plataforma **Basada en extensión (V2)** y otra **Basada en agente (V1)**.
>
> - Después de deshabilitar el Private Link en la cuenta de Automation, puede tardar hasta 60 minutos en quitar Hybrid Runbook Worker.
> - La configuración del proxy de Hybrid Runbook Worker se puede eliminar de HKLM\SOFTWARE\Microsoft\Azure\HybridWorker\Parameters, "Dirección URL del proxy de conexión Http".

Eliminación de un grupo de Hybrid Runbook Worker

Puede eliminar un grupo de Hybrid Runbook Worker vacío desde el portal.

1. En **Automatización de procesos**, seleccione **Grupos de Hybrid Worker** y, a continuación, el grupo de Hybrid Worker para ir a la página **Grupo de Hybrid Worker**.

2. Seleccione **Eliminar**.

Aparece un mensaje de advertencia para quitar todas las máquinas que se definen como trabajo híbrido en el grupo de trabajo híbrido. Si ya hay un trabajador agregado al grupo, primero tendrás que eliminar al trabajador del grupo.

3. Seleccione **Sí**.

   Se eliminará el grupo de Hybrid Worker.

Actualización automática de la extensión

La extensión de Hybrid Worker admite la actualización automática de versiones secundarias de forma predeterminada. Se recomienda habilitar las actualizaciones automáticas para aprovechar las actualizaciones de seguridad o características sin sobrecarga manual. Sin embargo, para evitar que la extensión se actualice automáticamente (por ejemplo, si hay ventanas de cambios estrictas y solo se puede actualizar en momentos específicos), puede optar por no usar esta funcionalidad configurando la propiedad `enableAutomaticUpgrade` en los archivos ARM, Bicep, o cmdlets de PowerShell en *false*. Establezca la misma propiedad en *true* siempre que quiera volver a habilitar la actualización automática.

PowerShell

```
$extensionType = "HybridWorkerForLinux/HybridWorkerForWindows"
$extensionName = "HybridWorkerExtension"
$publisher = "Microsoft.Azure.Automation.HybridWorker"
Set-AzVMExtension -ResourceGroupName <RGName> -Location <Location>  -VMName <vmName> -
Name $extensionName -Publisher $publisher -ExtensionType $extensionType -
TypeHandlerVersion 1.1 -Settings $settings -EnableAutomaticUpgrade $true/$false
```

Las actualizaciones de la versión principal deben administrarse manualmente. Ejecute los siguientes cmdlets con la versión más reciente de TypeHandlerVersion.

> ⓘ **Nota:**
>
> Si ha instalado la extensión de Hybrid Worker durante la versión preliminar pública, asegúrese de actualizarla a la versión principal más reciente.

Máquinas Virtuales de Azure

PowerShell

```
Set-AzVMExtension -ResourceGroupName <VMResourceGroupName> -Location <VMLocation> -
VMName <VMName> -Name "HybridWorkerExtension" -Publisher
"Microsoft.Azure.Automation.HybridWorker" -ExtensionType HybridWorkerForWindows -
TypeHandlerVersion 1.1 -Settings $settings -EnableAutomaticUpgrade $true/$false
```

Máquinas virtuales habilitadas por Azure Arc

PowerShell

```
New-AzConnectedMachineExtension -ResourceGroupName <VMResourceGroupName> -Location
<VMLocation> -MachineName <VMName> -Name "HybridWorkerExtension" -Publisher
"Microsoft.Azure.Automation.HybridWorker" -ExtensionType HybridWorkerForWindows -
TypeHandlerVersion 1.1 -Setting $settings -NoWait -EnableAutomaticUpgrade
```

Administrar la extensión de *Hybrid Worker* utilizando Bicep, plantillas ARM, API REST, CLI de Azure y PowerShell

PowerShell archivo

Puede usar cmdlets de PowerShell para crear un nuevo grupo de Hybrid Worker, crear una nueva máquina virtual de Azure, agregarla a un grupo de Hybrid Worker existente e instalar la extensión hybrid Worker.

Siga los pasos mencionados a continuación como ejemplo:

1. Cree un grupo de Trabajadores Híbridos.

   PowerShell

   ```
   New-AzAutomationHybridRunbookWorkerGroup -AutomationAccountName
   "Contoso17" -Name "RunbookWorkerGroupName" -ResourceGroupName
   "ResourceGroup01"
   ```

2. Cree una máquina virtual Azure o un servidor habilitado para Arc y agréguela al grupo de Hybrid Worker creado anteriormente. Use el comando siguiente para agregar una máquina virtual de Azure existente o un servidor habilitado para Arc al grupo de Hybrid Worker. Genere un nuevo GUID y páselo como nombre del Hybrid Worker. Para capturar `vmResourceId`, vaya a la pestaña **Properties** de la máquina virtual en Azure portal.

   Azure PowerShell

   ```
   #To fetch vmResourceId, go to the Properties tab of the VM on Azure
   portal.
   $hwVM = "VmResourceId"

   # Generate a new GUID and pass it as the name of the Hybrid Worker
   $hwguid = New-Guid

   # Create the Hybrid Worker
   New-AzAutomationHybridRunbookWorker `
   ```

```
 -Name $hwguid `
 -VmResourceId $hwVM `
 -HybridRunbookWorkerGroupName "RunbookWorkerGroupName" `
 -AutomationAccountName "Contoso17" `
 -ResourceGroupName "AutomationAccountResourceGroup01"
```

3. Siga los pasos que se indican aquí para habilitar la identidad administrada asignada por el sistema en la máquina virtual.

4. Instale la extensión de Hybrid Worker en la máquina virtual.

> ⓘ **Nota:**
>
> Si desea agregar un servidor proxy, consulte uso del servidor proxy.

PowerShell

```
  $settings = @{
"AutomationAccountURL"  = "<automationHybridServiceUrl>";
};
```

Máquinas Virtuales de Azure

PowerShell

```
 Set-AzVMExtension -ResourceGroupName <VMResourceGroupName> -Location
<VMLocation> -VMName <VMName> -Name "HybridWorkerExtension" -Publisher
"Microsoft.Azure.Automation.HybridWorker" -ExtensionType
HybridWorkerForWindows -TypeHandlerVersion 1.1 -Settings $settings -
EnableAutomaticUpgrade $true/$false
```

Máquinas virtuales habilitadas por Azure Arc

PowerShell

```
  New-AzConnectedMachineExtension -ResourceGroupName <VMResourceGroupName> -
Location <VMLocation> -MachineName <VMName> -Name "HybridWorkerExtension" -
Publisher "Microsoft.Azure.Automation.HybridWorker" -ExtensionType
HybridWorkerForWindows -TypeHandlerVersion 1.1 -Setting $settings -NoWait -
EnableAutomaticUpgrade
```

5. Para confirmar si la extensión se ha instalado correctamente en la máquina virtual, en **Azure portal**, vaya a la máquina virtual > **Extensions** y compruebe el estado de la extensión de Hybrid Worker instalada en la máquina virtual.

Administración de la extensión de Hybrid Worker

Para administrar el Hybrid Runbook Worker y los grupos de Hybrid Runbook Worker, puede usar los siguientes cmdlets de PowerShell:

Expandir tabla

Cmdlet de PowerShell	Descripción
Get-AzAutomationHybridRunbookWorkerGroup	Obtiene un grupo de Hybrid Runbook Worker
Remove-AzAutomationHybridRunbookWorkerGroup	Quita el grupo Hybrid Runbook Worker.
Set-AzAutomationHybridRunbookWorkerGroup	Actualiza el grupo de Hybrid Worker con credenciales de Hybrid Worker
New-AzAutomationHybridRunbookWorkerGroup	Crea un nuevo grupo de Hybrid Runbook Worker
Get-AzAutomationHybridRunbookWorker	Obtiene Hybrid Runbook Worker
Move-AzAutomationHybridRunbookWorker	Mueve a Trabajador Híbrido de un grupo a otro
New-AzAutomationHybridRunbookWorker	Crea un nuevo Hybrid Runbook Worker
Remove-AzAutomationHybridRunbookWorker	Elimina Hybrid Runbook Worker

Después de crear una instancia de Hybrid Runbook Worker, debe instalar la extensión en Hybrid Worker.

Configuración de la extensión de Hybrid Worker

PowerShell

```
$settings = @{
    "AutomationAccountURL"  = "<automationHybridServiceUrl>";
};
```

Máquinas Virtuales de Azure

PowerShell

```
Set-AzVMExtension -ResourceGroupName <VMResourceGroupName> -Location <VMLocation>
-VMName <VMName> -Name "HybridWorkerExtension" -Publisher
"Microsoft.Azure.Automation.HybridWorker" -ExtensionType HybridWorkerForWindows -
TypeHandlerVersion 1.1 -Settings $settings -EnableAutomaticUpgrade $true/$false
```

Máquinas virtuales habilitadas por Azure Arc

PowerShell

```
New-AzConnectedMachineExtension -ResourceGroupName <VMResourceGroupName> -
Location <VMLocation> -MachineName <VMName> -Name "HybridWorkerExtension" -
```

```
Publisher "Microsoft.Azure.Automation.HybridWorker" -ExtensionType
HybridWorkerForWindows -TypeHandlerVersion 1.1 -Setting $settings -NoWait -
EnableAutomaticUpgrade
```

Gestión de permisos de rol para Grupos de Trabajadores Híbridos y Trabajadores Híbridos.

Puede crear roles personalizados de Azure Automation y otorgar los siguientes permisos a Grupos de Trabajadores Híbridos y Trabajadores Híbridos. Para obtener más información sobre cómo crear roles personalizados de Azure Automation, consulte Azure roles personalizados.

Expandir tabla

Acciones	Descripci
Microsoft.Automation/automationAccounts/hybridRunbookWorkerGroups/read	Lee un grupo de Hybrid Runbook Worker.
Microsoft.Automation/automationAccounts/hybridRunbookWorkerGroups/write	Crea un grupo de Hybrid Runbook Worker.
Microsoft.Automation/automationAccounts/hybridRunbookWorkerGroups/delete	Elimina u grupo de Hybrid Runbook Worker.
Microsoft.Automation/automationAccounts/hybridRunbookWorkerGroups/hybridRunbookWorkers/read	Lee una instancia Hybrid Runbook Worker.
Microsoft.Automation/automationAccounts/hybridRunbookWorkerGroups/hybridRunbookWorkers/write	Crea una instancia Hybrid Runbook Worker.
Microsoft. Automation/automationAccounts/hybridRunbookWorkerGroups/hybridRunbookWorkers/move/action	Mueve Hybrid Runbook Worker d un grupo

Acciones	Descripci
	de trabajo otro.
Microsoft. Automation/automationAccounts/hybridRunbookWorkerGroups/hybridRunbookWorkers/delete	Elimina u instancia Hybrid Runbook Worker.

Comprobación de la versión de Hybrid Worker

Para comprobar la versión de Hybrid Runbook Worker basada en la extensión:

Expandir tabla

Tipos de sistema operativo	Rutas	Descripción
Windows	`C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows\`	La ruta de acceso tiene una carpeta llamada *versión* donde se encuentra la información de la versión.
Linux	`/var/lib/waagent/Microsoft.Azure.Automation.HybridWorker.HybridWorkerForLinux-<version>`	El nombre de la carpeta termina con información de *versión*.

Supervisión del rendimiento de Hybrid Worker mediante VM Insights

Con VM Insights, puede supervisar el rendimiento de las máquinas virtuales Azure y los servidores habilitados para Arc implementados como Hybrid Runbook Worker. Entre varios elementos que se consideran durante los rendimientos, VM Insights supervisa los indicadores clave de rendimiento del sistema operativo relacionados con el procesador, la memoria, el adaptador de red y el uso del disco.

- Para máquinas virtuales de Azure, consulte Cómo graficar el rendimiento con VM Insights.
- Para servidores habilitados para Arc, consulte Tutorial: Supervisión de una máquina híbrida con VM Insights.

Pasos siguientes

- Para obtener información sobre cómo configurar los runbooks para automatizar procesos en el centro de datos local u otros entornos en la nube, consulte Ejecución de runbooks en Hybrid Runbook Worker.

- Para solucionar problemas relacionados con la implementación de los Hybrid Runbook Workers de usuario basados en extensiones de Windows o Linux en Azure Automation, consulte Solucionar problemas de los Hybrid Runbook Workers basados en extensiones de máquina virtual en Automation.

- Para obtener información sobre las extensiones de máquina virtual de Azure, consulte Azure extensiones y características de máquina virtual para Windows y Azure extensiones y características de máquina virtual para Linux.

- Para obtener información sobre las extensiones de máquina virtual para servidores habilitados para Arc, consulte administración de extensiones VM con servidores habilitados para Azure Arc.

- Para obtener información sobre los servicios de administración de Azure para máquinas virtuales de VMware con compatibilidad Arc, consulte Instalar agentes de Arc a escala para sus máquinas virtuales de VMware.

- Para obtener información sobre los servicios de administración de Azure para máquinas virtuales SCVMM habilitadas para Arc, consulte Instale agentes de Arc a escala para máquinas virtuales SCVMM habilitadas para Arc.

Last updated on 05/05/2026

Ejecución de runbooks de Automation en una instancia de Hybrid Runbook Worker

> ⓘ **Importante**
>
> - A partir del 1 de abril de 2025, se detendrán todos los trabajos que se ejecutan en Hybrid Worker basado en agente.
> - Azure Automation Hybrid Runbook Worker basado en agente (Windows y Linux) se ha retirado en **31 de agosto de 2024** y ya no se admite. Siga las instrucciones sobre cómo migrar de un trabajo híbrido de runbook existente basado en agentes a un trabajo híbrido basado en extensiones.

Los runbooks que se ejecutan en una instancia de Hybrid Runbook Worker normalmente administran recursos en el equipo local o en recursos del entorno local en el que se implementa el rol de trabajo. Los runbooks de Azure Automation suelen administrar recursos en la nube de Azure. Aunque se usan de forma diferente, los runbooks que se ejecutan en Azure Automation y los runbooks que se ejecutan en Hybrid Runbook Worker son idénticos en cuanto a estructura.

Cuando cree un runbook para que se ejecute en una instancia de Hybrid Runbook Worker, debe editarlo y probarlo en la máquina que hospeda el rol de trabajo. La máquina host tiene todos los módulos de PowerShell y el acceso de red necesarios para administrar los recursos locales. Una vez que pruebe el runbook en la máquina de Hybrid Runbook Worker, puede cargarlo en el entorno de Azure Automation, donde se puede ejecutar en el trabajo.

Planear los servicios de Azure protegidos por firewall

Habilitar el Azure Firewall en Azure Storage, Azure Key Vault o Azure SQL bloquea el acceso de los runbooks de Azure Automation para esos servicios. El acceso se bloqueará incluso cuando la excepción del firewall para permitir los servicios de Microsoft de confianza esté habilitada, ya que Automatización no forma parte de la lista de servicios de confianza. Con un firewall habilitado, el acceso solo se puede realizar mediante una instancia de Hybrid Runbook Worker y un punto de conexión de servicio de red virtual.

Planeación del comportamiento del trabajo de runbook

Azure Automation controla los trabajos en Hybrid Runbook Worker de forma diferente a los trabajos que se ejecutan en espacios aislados en la nube. Si tiene un runbook de larga ejecución, asegúrese de que sea resistente a posibles reinicios. Para obtener detalles del comportamiento del trabajo, consulte Trabajos de Hybrid Runbook Worker.

Cuentas de servicio

Windows Hybrid Worker

Los trabajos de instancias de Hybrid Runbook Worker se ejecutan en la cuenta de **sistema** local.

> ⓘ **Nota:**
>
> - Los runbooks de PowerShell 7.4 y Python 3.10 solo se admiten en trabajadores híbridos de Windows basados en extensiones. Asegúrese de que la versión de extensión de Hybrid Worker de Windows sea 1.3.63 o superior.
> - PowerShell 5.1, PowerShell 7.1 (versión preliminar), Python 2.7 y Python 3.8 se admiten en los Windows Hybrid Runbook Workers tanto basados en extensiones como basados en agentes. En el caso de los trabajadores basados en agentes, asegúrese de que la versión de Windows de Hybrid Worker sea 7.3.12960 o superior.
> - El runbook de PowerShell 7.2 solo se admite en Windows Hybrid Worker basados en extensiones. Asegúrese de que la versión de extensión de Hybrid Worker de Windows sea 1.1.11 o superior.

Hybrid Worker basado en extensiones

> ⓘ **Nota:**
>
> Para crear una variable de entorno en sistemas Windows, siga estos pasos:
>
> 1. Vaya a **Panel de control**>**System**>**Configuración del sistema avanzada**.
> 2. En **Propiedades del sistema** , seleccione **Variables de entorno**.
> 3. En **Variables del sistema**, seleccione **Nuevo**.
> 4. Proporcione **el nombre de** variable y el **valor de variable** y, a continuación, seleccione **Aceptar**.
> 5. Reinicie la máquina virtual o cierre la sesión desde el usuario actual e inicie sesión para implementar los cambios de la variable de entorno.

PowerShell 7.4

Para ejecutar runbooks de PowerShell 7.4 en una instancia de Hybrid Worker de Windows, instale *PowerShell* en Hybrid Worker. Consulte Instalar PowerShell en Windows.

Una vez completada la instalación de PowerShell 7.4, cree una variable de entorno cuyo nombre sea powershell_7_4_path y cuyo valor sea la ruta de acceso del archivo ejecutable *PowerShell*. Reinicie la instancia de Hybrid Runbook Worker una vez creada correctamente la variable de entorno.

PowerShell 7.2

Para ejecutar runbooks de PowerShell 7.2 en una instancia de Hybrid Worker de Windows, instale *PowerShell* en Hybrid Worker. Consulte Instalar PowerShell en Windows.

Una vez completada la instalación de PowerShell 7.2, cree una variable de entorno con el nombre de variable "powershell_7_2_path" y con el valor como la ruta de acceso del ejecutable *PowerShell*. Reinicie la instancia de Hybrid Runbook Worker una vez creada correctamente la variable de entorno.

PowerShell 7.1

Para ejecutar runbooks de PowerShell 7.1 en una instancia de Hybrid Worker de Windows, instale *PowerShell* en Hybrid Worker. Consulte Instalar PowerShell en Windows. Asegúrese de agregar el archivo *de PowerShell* a la variable de entorno PATH y reinicie Hybrid Runbook Worker después de la instalación.

Python 3.10

Para ejecutar runbooks de Python 3.10 en una instancia de Hybrid Worker de Windows, instale *Python* en Hybrid Worker. Consulte Install Python en Windows.

Una vez completada la instalación de Python 3.10, cree una variable de entorno con nombre de variable como python_3_10_path y valor variable como ruta de acceso del archivo ejecutable *Python*. Reinicie la instancia de Hybrid Runbook Worker una vez creada correctamente la variable de entorno.

Python 3.8

Para ejecutar runbooks de Python 3.8 en una instancia de Hybrid Worker Windows, instale Python en Hybrid Worker. Consulte Install Python en Windows. Cree la **variable de entorno***PYTHON_3_PATH* para los runbooks de Python 3.8 y asegúrese de agregar la ubicación

del ejecutable de Python como **valor de la variable**. Reinicie la instancia de Hybrid Runbook Worker una vez creada correctamente la variable de entorno.

Si el archivo ejecutable *Python* está en la ubicación predeterminada *C:\WPy64-3800\python-3.8.0.amd64\python.exe*, no es necesario crear la variable de entorno.

Python 2.7

Para ejecutar runbooks de Python 2.7 en una instancia de Hybrid Worker Windows, instale Python en Hybrid Worker. Consulte Install Python en Windows. Cree **variable de entorno***PYTHON_2_PATH* para runbooks de Python 2.7 y asegúrese de agregar la ubicación del archivo ejecutable Python como **valor de la variable**. Reinicie la instancia de Hybrid Runbook Worker una vez creada correctamente la variable de entorno.

Si el archivo ejecutable *Python* está en la ubicación predeterminada *C:\Python27\python.exe*, no es necesario crear la variable de entorno.

Linux Hybrid Worker

> **Nota:**
>
> - Los runbooks de PowerShell 7.4 y Python 3.10 solo se admiten en Hybrid Worker de Linux basado en extensiones. Asegúrese de que la versión de la extensión de Hybrid Worker de Linux es 1.1.23 o superior.
> - Se admiten los runbooks de PowerShell 7.1 (versión preliminar), Python 2.7 y Python 3.8 en los Trabajadores híbridos de runbooks basados en extensiones o en agentes. Para los trabajos basados en agente, asegúrese de que la versión de Linux Hybrid Runbook Worker es 1.7.5.0 o posterior.
> - El runbook de PowerShell 7.2 solo se admite en trabajadores híbridos de Linux basados en extensiones. Asegúrese de que la versión de la extensión de Linux Hybrid Worker sea 1.1.11 o posterior.

Hybrid Worker basado en extensiones

> **Nota:**
>
> Para crear una variable de entorno en sistemas Linux, siga estos pasos:
>
> 1. Abra /etc/environment.

2. Cree una nueva variable de entorno agregando VARIABLE_NAME="variable_value" en una nueva línea de /etc/environment (VARIABLE_NAME es el nombre de la nueva variable environment y variable_value representa el valor al que se va a asignar).
3. Reinicie la máquina virtual o cierre la sesión del usuario actual e inicie sesión después de guardar los cambios en /etc/environment para implementar los cambios en las variables de entorno.

PowerShell 7.4

Para ejecutar runbooks de PowerShell 7.4 en una instancia de Linux Hybrid Worker, instale el archivo *de PowerShell* en Hybrid Worker. Consulte Instalación de PowerShell en Linux.

Una vez completada la instalación de PowerShell 7.4, cree una variable de entorno con **Variable name** como powershell_7_4_path y **Variable value** como la ruta del archivo ejecutable de *PowerShell*, por ejemplo: `powershell_7_4_path="/usr/bin/pwsh"`

Reinicie Hybrid Runbook Worker una vez creada correctamente una variable de entorno.

PowerShell 7.2

Para ejecutar runbooks de PowerShell 7.2 en una instancia de Linux Hybrid Worker, instale el archivo *de PowerShell* en Hybrid Worker. Para obtener más información, consulte Instalación de PowerShell en Linux.

Una vez completada la instalación de PowerShell 7.2, cree una variable de entorno con el **nombre de la variable** como *powershell_7_2_path* y el **valor de la variable** como la ruta de acceso al archivo ejecutable de *PowerShell*. Reinicie Hybrid Runbook Worker una vez creada correctamente una variable de entorno.

Python 3.10

Para ejecutar runbooks de Python 3.10 en una instancia de Linux Hybrid Worker, instale *Python* en Hybrid Worker. Para obtener más información, consulte Install Python 3.10 en Linux.

Una vez completada la instalación de Python 3.10, cree una variable de entorno con **Variable name** como *python_3_10_path* y **Variable value** como ruta de acceso del archivo ejecutable *Python*, por ejemplo: `python_3_10_path="/usr/bin/python3.10"`

Reinicie la instancia de Hybrid Runbook Worker una vez creada correctamente la variable de entorno.

Python 3.8

Para ejecutar runbooks de Python 3.8 en una instancia de Linux Hybrid Worker, instale *Python* en Hybrid Worker. Asegúrese de agregar el archivo ejecutable *Python* a la variable de entorno PATH y reinicie Hybrid Runbook Worker después de la instalación.

Python 2.7

Para ejecutar runbooks de Python 2.7 en una instancia de Linux Hybrid Worker, instale *Python* en Hybrid Worker. Asegúrese de agregar el archivo ejecutable *Python* a la variable de entorno PATH y reinicie Hybrid Runbook Worker después de la instalación.

Configuración de permisos de runbooks

Defina los permisos para que el runbook se ejecute en la instancia de Hybrid Runbook de las siguientes formas:

- Haga que el runbook proporcione su propia autenticación a los recursos locales.
- Configure la autenticación mediante identidades administradas para Azure recursos.
- Especifique las credenciales de Hybrid Worker para proporcionar un contexto de usuario para todos los runbooks.

Uso de la autenticación de runbooks en los recursos locales

Si prepara un runbook que proporciona su propia autenticación a los recursos, use los recursos Credencial y Certificado en el runbook. Hay varios cmdlets que le permiten especificar las credenciales, para que el runbook pueda autenticarse en los distintos recursos. El ejemplo siguiente muestra una porción de un runbook que reinicia un equipo. Recupera las credenciales desde un recurso de credencial y el nombre del equipo desde un recurso de variable y, a continuación, usa estos valores con el cmdlet `Restart-Computer`.

PowerShell

```
$Cred = Get-AutomationPSCredential -Name "MyCredential"
$Computer = Get-AutomationVariable -Name "ComputerName"

Restart-Computer -ComputerName $Computer -Credential $Cred
```

También puede usar una actividad InlineScript . `InlineScript` permite ejecutar bloques de código en otro equipo con credenciales.

Uso de la autenticación de runbook con identidades administradas

Hybrid Runbook Workers en las máquinas virtuales de Azure pueden usar identidades administradas para autenticarse a recursos de Azure. El uso de identidades administradas para recursos de Azure en lugar de cuentas Run As proporciona ventajas porque no es necesario:

- Exportar el certificado de ejecución y después importarlo en Hybrid Runbook Worker.
- Renovar el certificado utilizado por la cuenta de ejecución.
- Administrar el objeto de conexión de ejecución en el código del runbook.

Hay dos maneras de usar las identidades administradas en scripts de Hybrid Runbook Worker.

1. Use una identidad administrada asignada por el sistema para una cuenta de Azure Automation:

   a. Configure una identidad administrada asignada por el sistema para la cuenta de Automation.

   b. Conceda a esta identidad los permisos necesarios dentro de la suscripción para realizar su tarea.

   c. Actualice el runbook para usar el cmdlet Connect-AzAccount con el parámetro `Identity` para autenticarse en recursos de Azure. Esta configuración reduce la necesidad de usar una cuenta de ejecución y realizar la administración de las cuentas asociadas.

   PowerShell

   ```
   # Ensures you do not inherit an AzContext in your runbook
   Disable-AzContextAutosave -Scope Process

   # Connect to Azure with system-assigned managed identity
   $AzureContext = (Connect-AzAccount -Identity).context

   # set and store context
   $AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
   DefaultProfile
   $AzureContext

   # Get all VM names from the subscription
   Get-AzVM -DefaultProfile $AzureContext | Select Name
   ```

> ⓘ **Nota:**

No es posible usar la identidad administrada de usuario de la cuenta de Automation en una instancia de Hybrid Runbook Worker; debe ser la identidad administrada del sistema de la cuenta de Automation.

2. Para una máquina virtual de Azure que se ejecuta como Hybrid Runbook Worker, use la identidad administrada de **VM**. En este caso, puede usar la identidad administrada asignada por el usuario de la máquina virtual **o** la identidad administrada asignada por el sistema de la máquina virtual.

> ⓘ **Nota:**
>
> Recuerde que esto **NO** funcionará en una cuenta de Automation que se haya configurado con una identidad administrada de cuenta de Automation. Tan pronto como la identidad administrada de la cuenta de Automation esté habilitada, ya no será posible usar la identidad administrada de la VM, por lo que solo es posible usar la cuenta de Identidad administrada asignada por el sistema de Automation, tal como se mencionó en la opción 1 anterior.

Use **cualquiera** de las siguientes identidades administradas:

Identidad administrada asignada por el sistema de la VM

a. Configure una identidad administrada por el sistema para la máquina virtual.
b. Conceda a esta identidad los permisos necesarios dentro de la suscripción para realizar sus tareas.
c. Actualice el runbook para usar el cmdlet Connect-Az-Account con el parámetro `Identity` para autenticarse en recursos de Azure. Esta configuración reduce la necesidad de usar una cuenta de ejecución y realizar la administración de las cuentas asociadas.

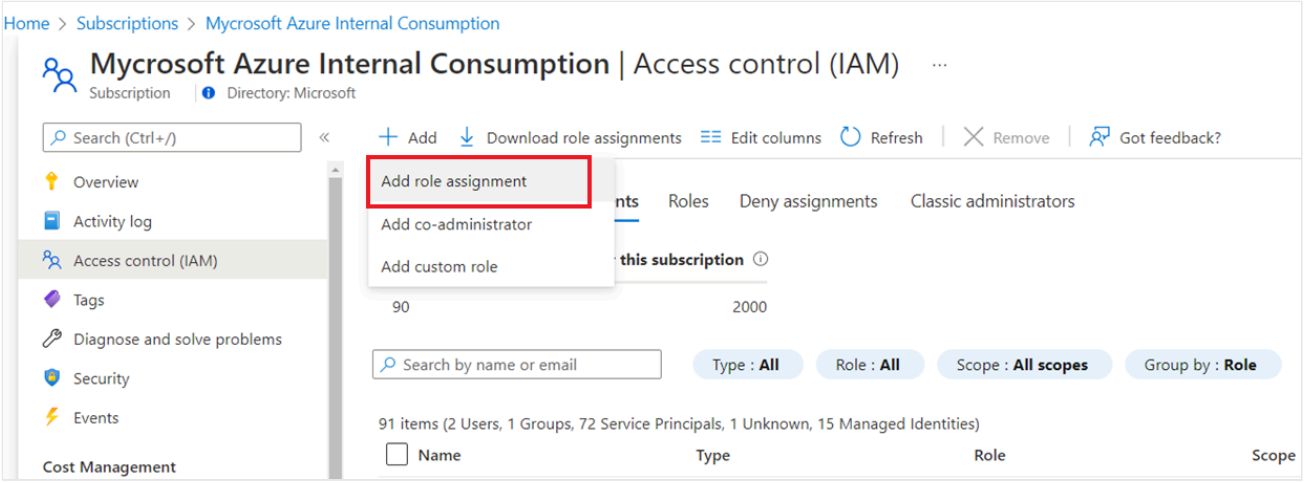
PowerShell

```
    # Ensures you do not inherit an AzContext in your runbook
    Disable-AzContextAutosave -Scope Process

    # Connect to Azure with system-assigned managed identity
    $AzureContext = (Connect-AzAccount -Identity).context

    # set and store context
    $AzureContext = Set-AzContext -SubscriptionName
$AzureContext.Subscription -DefaultProfile
```

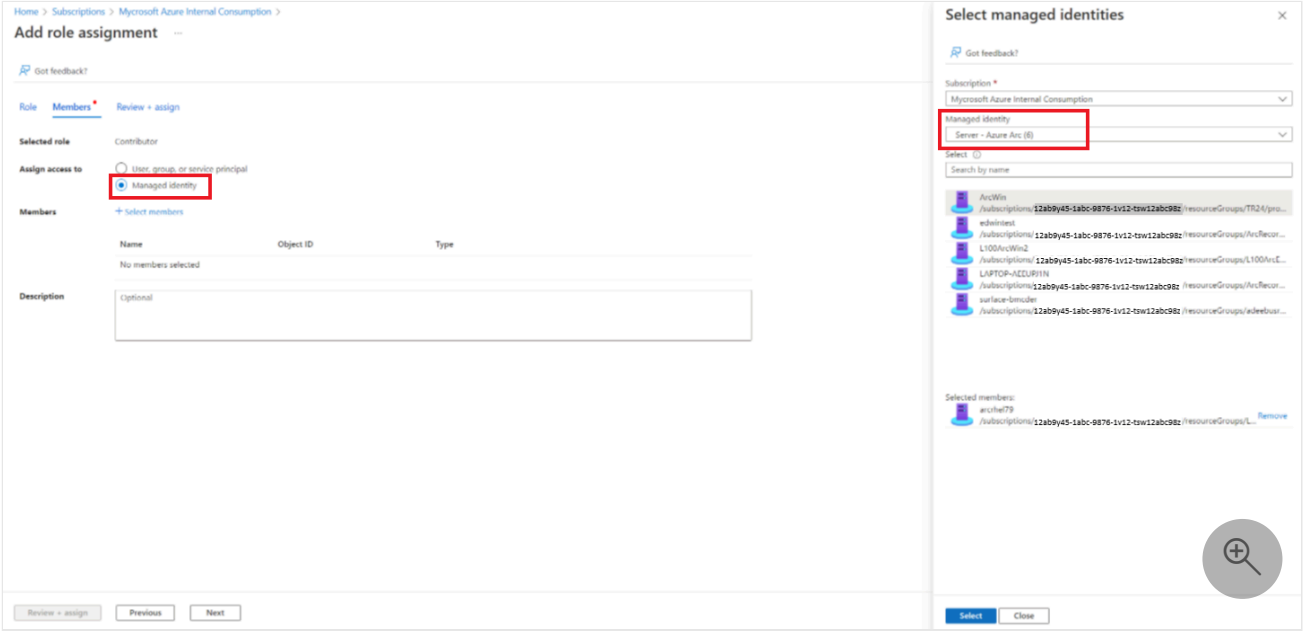
```
$AzureContext

# Get all VM names from the subscription
Get-AzVM -DefaultProfile $AzureContext | Select Name
```

Un servidor habilitado para Arc o una máquina virtual de VMware vSphere habilitada para Arc que se ejecuta como Hybrid Runbook Worker ya tiene asignada una identidad administrada del sistema integrada que se puede usar para la autenticación.

1. Puede conceder a esta identidad administrada acceso a los recursos de la suscripción en la hoja de Control de acceso (IAM) del recurso, si agrega la asignación de roles adecuada.



2. Agregue la Azure Arc identidad administrada al rol elegido según sea necesario.



> ⓘ **Nota:**

Recuerde que esto **NO** funcionará en una cuenta de Automation que se haya configurado con una identidad administrada de cuenta de Automation. Tan pronto como se habilite la Identidad Administrada de la cuenta de Automatización, ya no es posible usar la Identidad Administrada de Arc y, a partir de entonces, **solo** es posible usar la Identidad Administrada Asignada por el Sistema de la Cuenta de Automatización, como se mencionó en la opción 1 anterior.

ⓘ **Nota:**

De forma predeterminada, los contextos de Azure se guardan para su uso entre sesiones de PowerShell. Es posible que cuando se haya autenticado un runbook anterior en Hybrid Runbook Worker con Azure, ese contexto persista en el disco del perfil de PowerShell del sistema, según contextos y credenciales de inicio de sesión de Azure | Microsoft Docs. Por ejemplo, un runbook con `Get-AzVM` puede devolver todas las máquinas virtuales de la suscripción sin llamar a `Connect-AzAccount`, y el usuario podría acceder a recursos de Azure sin tener que autenticarse en ese runbook. Puede deshabilitar el guardado automático de contexto en Azure PowerShell, como se detalla here.

Uso de la autenticación de runbook con credenciales de Hybrid Worker

Prerequisite

- Hybrid Worker debe implementarse y la máquina debe estar en estado de ejecución antes de ejecutar un runbook.

Credenciales de Hybrid Worker En lugar de hacer que el runbook proporcione su propia autenticación a los recursos locales, puede especificar credenciales de Hybrid Worker para un grupo de Hybrid Runbook Worker. Para especificar las credenciales de Hybrid Worker, debe definir un recurso de credenciales que tenga acceso a los recursos locales. Estos recursos incluyen almacenes de certificados y todos los runbooks se ejecutan con estas credenciales en una instancia de Hybrid Runbook Worker del grupo.

- El nombre de usuario de la credencial debe tener uno de los siguientes formatos:
  - domain\username
  - username@domain
  - nombre de usuario (para cuentas locales en el equipo local)

- Para usar el runbook de PowerShell **Export-RunAsCertificateToHybridWorker**, es necesario instalar los módulos Az para Azure Automation en la máquina local.

Uso de un recurso de credencial para un grupo de Hybrid Runbook Worker

De manera predeterminada, los trabajos híbridos se ejecutan en el contexto de la cuenta del sistema. Sin embargo, para ejecutar trabajos híbridos en otro recurso de credencial, siga estos pasos:

1. Cree un recurso de credenciales con acceso a los recursos locales.
2. Abra la cuenta de Automation en el portal de Azure.
3. Seleccione **Grupos de Hybrid Worker** y elija el grupo concreto.
4. Haga clic en **Configuración**.
5. Cambie el valor de **Credenciales de Hybrid Worker** de **Predeterminado** a **Personalizado**.
6. Seleccione la credencial y seleccione **Guardar**.
7. Si no se asignan los permisos siguientes para usuarios personalizados, es posible que los trabajos se suspendan.

Expandir tabla

Tipo de recurso	Permisos de carpeta
máquina virtual de Azure	C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows (lectura y ejecución)
Servidor habilitado para Arc	C:\ProgramData\AzureConnectedMachineAgent\Tokens (lectura) C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows (lectura y ejecución)

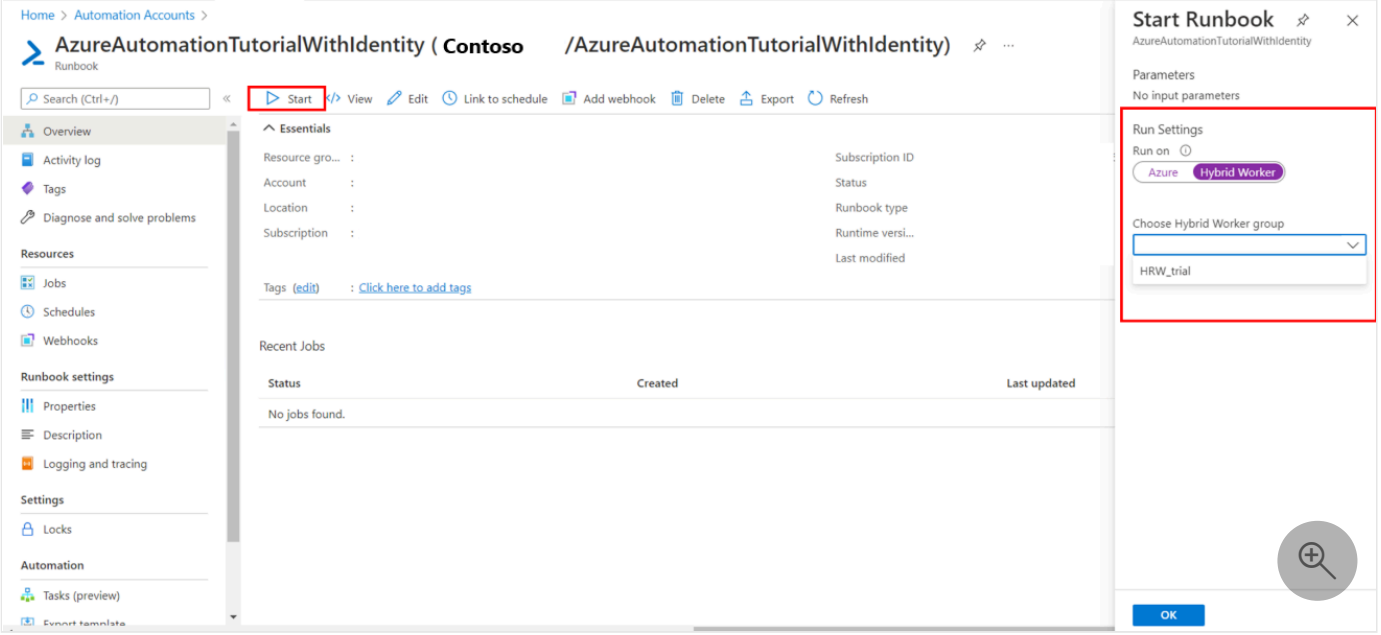
> **Nota:**
>
> Hybrid Worker de Linux no admite credenciales de Hybrid Worker.

Inicio de un runbook en una instancia de Hybrid Runbook Worker

Iniciar un runbook en Azure Automation describe distintos métodos para iniciar un runbook. El inicio de un runbook en una instancia de Hybrid Runbook Worker usa la opción **Ejecutar en**, que

le permite especificar el nombre de un grupo de Hybrid Runbook Worker. Si se especifica un grupo, uno de los roles de trabajo del grupo recupera y ejecuta el runbook. Si el runbook no especifica esta opción, Azure Automation ejecuta el runbook como de costumbre.

Al iniciar un runbook en el portal de Azure, se le presenta la opción **Run on** para la que puede seleccionar **Azure** o **Hybrid Worker**. Seleccione **Hybrid Worker** para elegir el grupo Hybrid Runbook Worker en una lista desplegable.



Al iniciar un runbook con PowerShell, use el parámetro `RunOn` con el cmdlet Start-AzAutomationRunbook. En el ejemplo siguiente se usa Windows PowerShell para iniciar un runbook denominado **Test-Runbook** en un grupo de Hybrid Runbook Worker denominado MyHybridGroup.

Azure PowerShell

```
Start-AzAutomationRunbook -AutomationAccountName "MyAutomationAccount" -Name "Test-Runbook" -RunOn "MyHybridGroup"
```

Trabajar con runbooks firmados en un Windows Hybrid Runbook Worker

Puede configurar un Windows Hybrid Runbook Worker para ejecutar solo runbooks firmados.

> **Importante**
>
> Cuando haya configurado una instancia de Hybrid Runbook Worker para que solo ejecute runbooks firmados, los runbooks que no estén firmados no se ejecutarán en el rol de

trabajo.

> **Nota:**
>
> PowerShell 7.x no admite runbooks firmados para Windows y Hybrid Runbook Worker de Linux.

Creación de certificado de firma

En el ejemplo siguiente se crea un certificado autofirmado que se puede usar para firmar runbooks. Este código crea el certificado y lo exporta para que Hybrid Runbook Worker pueda importarlo más adelante. También se devuelve la huella digital para su uso posterior al hacer referencia al certificado.

PowerShell

```
# Create a self-signed certificate that can be used for code signing
$SigningCert = New-SelfSignedCertificate -CertStoreLocation cert:\LocalMachine\my `
    -Subject "CN=contoso.com" `
    -KeyAlgorithm RSA `
    -KeyLength 2048 `
    -Provider "Microsoft Enhanced RSA and AES Cryptographic Provider" `
    -KeyExportPolicy Exportable `
    -KeyUsage DigitalSignature `
    -Type CodeSigningCert

# Export the certificate so that it can be imported to the hybrid workers
Export-Certificate -Cert $SigningCert -FilePath .\hybridworkersigningcertificate.cer

# Import the certificate into the trusted root store so the certificate chain can be
validated
Import-Certificate -FilePath .\hybridworkersigningcertificate.cer -CertStoreLocation
Cert:\LocalMachine\Root

# Retrieve the thumbprint for later use
$SigningCert.Thumbprint
```

Importación del certificado y configuración de los roles de trabajo para la validación de firmas

Copie el certificado que ha creado en cada instancia de Hybrid Runbook Worker de un grupo. Ejecute el script siguiente para importar el certificado y configurar los roles de trabajo para que usen la validación de firma en los runbooks.

PowerShell

```
# Install the certificate into a location that will be used for validation.
New-Item -Path Cert:\LocalMachine\AutomationHybridStore
Import-Certificate -FilePath .\hybridworkersigningcertificate.cer -CertStoreLocation
Cert:\LocalMachine\AutomationHybridStore

# Import the certificate into the trusted root store so the certificate chain can be
validated
Import-Certificate -FilePath .\hybridworkersigningcertificate.cer -CertStoreLocation
Cert:\LocalMachine\Root

# Configure the hybrid worker to use signature validation on runbooks.
Set-HybridRunbookWorkerSignatureValidation -Enable $true -TrustedCertStoreLocation
"Cert:\LocalMachine\AutomationHybridStore"
```

Firma de los runbooks con el certificado

Con las instancias de Hybrid Runbook Worker configuradas para usar solo runbooks firmados, debe firmar los runbooks que se van a utilizar en Hybrid Runbook Worker. Use el código de PowerShell de ejemplo siguiente para firmar estos runbooks.

PowerShell

```
$SigningCert = ( Get-ChildItem -Path cert:\LocalMachine\My\<CertificateThumbprint>)
Set-AuthenticodeSignature .\TestRunbook.ps1 -Certificate $SigningCert
```

Una vez firmado el runbook, debe importarlo en la cuenta de Automation y publicarlo con el bloque de firma. Para aprender a importar runbooks, consulte Importación de un runbook.

> **Nota:**
>
> Use solo caracteres de texto no cifrado en el código del runbook, incluidos los comentarios. El uso de caracteres con marcas diacríticas, como á o ñ, producirá un error. Cuando Azure Automation descarga tu código, los caracteres se reemplazarán por un signo de interrogación y la firma fallará con un mensaje de "fallo en la validación del hash de la firma".

Trabajo con runbooks firmados en Hybrid Runbook Worker en Linux

Para poder trabajar con runbooks firmados, una instancia de Hybrid Runbook Worker de Linux debe tener el ejecutable GPG en la máquina local.

> **Importante**
>
> Cuando haya configurado una instancia de Hybrid Runbook Worker para que solo ejecute runbooks firmados, los runbooks que no estén firmados no se ejecutarán en el rol de trabajo.

Realizará los pasos siguientes para completar esta configuración:

- Creación un conjunto de claves y un par de claves de GPG
- Puesta a disposición del conjunto de claves para Hybrid Runbook Worker
- Verificación de que la validación de firmas está activada
- Ejecución de un runbook

> **Nota:**
>
> - PowerShell 7.x no admite runbooks firmados para la Windows basada en agente y Hybrid Runbook Worker basado en agente de Linux.
> - Los runbooks de PowerShell y Python firmados no son compatibles en trabajadores híbridos de Linux basados en extensiones.

Creación un conjunto de claves y un par de claves de GPG

> **Nota:**
>
> La opción Crear un conjunto de claves y un par de claves de GPG solo es aplicable a los trabajos híbridos basados en agente.

Para crear un llavero GPG y un par de claves, use Hybrid Runbook Worker:

1. Utiliza la aplicación sudo para iniciar sesión en la cuenta **nxautomation**.

   Bash

   ```
   sudo su - nxautomation
   ```

2. Una vez que esté usando **nxautomation**, genere el par de claves GPG como usuario root. GPG le guía por los pasos. Debe proporcionar el nombre, la dirección de correo electrónico, la hora de expiración y la frase de contraseña. A continuación, espere hasta que haya suficiente entropía en la máquina para que se genere la clave.

   Bash
   ```
   sudo gpg --generate-key
   ```

3. Dado que el directorio GPG se generó con sudo, debe cambiar su propietario a **nxautomation** mediante el siguiente comando como raíz.

   Bash
   ```
   sudo chown -R nxautomation ~/.gnupg
   ```

Puesta a disposición del conjunto de claves para Hybrid Runbook Worker

Una vez creado el conjunto de claves, debe ponerlo a disposición de Hybrid Runbook Worker. Modifique el archivo de configuración **home/nxautomation/state/worker.conf** para incluir el siguiente código de ejemplo en la sección de archivo `[worker-optional]`.

Bash
```
gpg_public_keyring_path = /home/nxautomation/run/.gnupg/pubring.kbx
```

Verificación de que la validación de firmas está activada

Si se ha deshabilitado la validación de firmas en el equipo, debe activarla; para ello, ejecute el siguiente comando como raíz. Reemplace `<LogAnalyticsworkspaceId>` por el identificador del área de trabajo.

Bash
```
sudo python
/opt/microsoft/omsconfig/modules/nxOMSAutomationWorker/DSCResources/MSFT_nxOMSAutomati
onWorkerResource/automationworker/scripts/require_runbook_signature.py --true
<LogAnalyticsworkspaceId>
```

Ejecución de un runbook

Cuando haya configurado la validación de firmas, use el siguiente comando de GPG para firmar el runbook.

```bash
gpg --clear-sign <runbook name>
```

El runbook firmado se llama **<nombre del runbook>.asc**.

Ahora puede cargar el runbook firmado en Azure Automation y ejecutarlo como un runbook normal.

Logging

Para ayudar a solucionar problemas con los runbooks que se ejecutan en una instancia de Hybrid Runbook Worker basada en extensiones, los registros se almacenan localmente en la siguiente ubicación:

- Para registros detallados del proceso de ejecución de trabajos y extensiones en Windows, visite `C:\WindowsAzure\Logs\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows<version>`. Los eventos de estado del trabajo de runbook de nivel alto se escriben en el registro de eventos **Applications and Services Logs/Microsoft-SMA/Operational**.
- En Linux, los registros de Hybrid Worker basados en extensiones se pueden encontrar en `/home/hweautomation/run/worker.log` y `/var/log/azure/Microsoft.Azure.Automation.HybridWorker.HybridWorkerForLinux`.

Pasos siguientes

- Para obtener más información sobre Hybrid Runbook Worker, consulte Hybrid Runbook Worker de Automation.
- Para más información sobre PowerShell, incluidos los módulos de referencia de lenguaje y aprendizaje, consulte La documentación de PowerShell.
- Obtenga información sobre usar Azure Policy para administrar la ejecución del runbook con Hybrid Runbook Workers.
- Para obtener una referencia de cmdlet de PowerShell, consulte Az.Automation.
- Para solucionar problemas relacionados con la ejecución de runbooks de Azure Automation en Hybrid Runbook Workers basados en extensiones, consulte Troubleshoot VM extension-based Hybrid Runbook Worker issues in Automation.

- Para solucionar problemas relacionados con la ejecución de runbooks de Azure Automation en Hybrid Runbook Workers basados en agente, consulte Problemas de Hybrid Runbook Worker basados en agente en Automation.

Last updated on 20/04/2026

Uso de Azure Policy para exigir la ejecución de trabajos en Hybrid Runbook Worker

> ⓘ **Importante**
>
> - A partir del 1 de abril de 2025, se detendrán todos los trabajos que se ejecutan en Hybrid Worker basado en agente.
> - Hybrid Runbook Worker (Windows y Linux) basado en agente de Azure Automation se ha retirado el **31 de agosto de 2024** y ya no se admite. Siga las instrucciones sobre cómo migrar desde una instancia de Hybrid Runbook Worker basada en agente existente a Hybrid Worker basado en extensiones.

Al iniciar un runbook en una instancia de Hybrid Runbook Worker se usa la opción **Ejecutar en** que permite especificar el nombre de un grupo de Hybrid Runbook Worker al iniciarse desde Azure Portal, con Azure PowerShell o la API REST. Si se especifica un grupo, uno de los roles de trabajo del grupo recupera y ejecuta el runbook. Si el runbook no especifica esta opción, Azure Automation ejecuta el runbook en el espacio aislado de Azure.

Cualquier persona de su organización que sea miembro del Operador de trabajos de Automation o superior puede crear trabajos de runbook. Para administrar la ejecución de runbooks dirigida a un grupo de Hybrid Runbook Worker en la cuenta de Automation, puede usar Azure Policy. Esto ayuda a aplicar los estándares organizativos y a garantizar que los trabajos de automatización están controlados y administrados por las personas designadas. Además, se asegura de que nadie pueda ejecutar un runbook en un espacio aislado de Azure, solo en Hybrid Runbook Worker.

En este artículo se incluye una definición de Azure Policy personalizada para ayudarle a controlar estas actividades mediante las siguientes operaciones de la API REST de Automation. Específicamente:

- Crear el trabajo
- Create job schedule
- Create webhook

Esta directiva se basa en la propiedad `runOn`. La directiva valida el valor de la propiedad, que debe contener el nombre de un grupo de Hybrid Runbook Worker existente. Si el valor es NULL, se interpreta como si la solicitud de creación del trabajo, programación del trabajo o webhook estuviera dirigida al espacio aislado de Azure y se deniega la solicitud.

Permisos requeridos

Debe ser miembro del rol Propietario en el nivel de suscripción a fin de tener los permisos para acceder a recursos de Azure Policy.

Creación y asignación de la definición de directiva

En este punto se crea la regla de directiva y, a continuación, se asigna a un grupo de administración o una suscripción y, opcionalmente, se especifica un grupo de recursos en la suscripción. Si aún no está familiarizado con el lenguaje de directiva, consulte la estructura de definición de directivas para saber cómo estructurar la definición de directiva.

1. Utilice el siguiente fragmento de código JSON para crear un archivo JSON con el nombre AuditAutomationHRWJobExecution.json.

JSON

```
 {
   "properties": {
     "displayName": "Enforce job execution on Automation Hybrid Runbook
Worker",
     "description": "Enforce job execution on Hybrid Runbook Workers in your
Automation account.",
     "mode": "all",
     "parameters": {
       "effectType": {
         "type": "string",
         "defaultValue": "Deny",
         "allowedValues": [
            "Deny",
            "Disabled"
         ],
         "metadata": {
           "displayName": "Effect",
           "description": "Enable or disable execution of the policy"
         }
       }
     },
     "policyRule": {
       "if": {
         "anyOf": [
           {
             "allOf": [
               {
                 "field": "type",
                 "equals": "Microsoft.Automation/automationAccounts/jobs"
               },
               {
                 "value": "
```

```
[length(field('Microsoft.Automation/automationAccounts/jobs/runOn'))]",
                "less": 1
              }
            ]
          },
          {
            "allOf": [
              {
                "field": "type",
                "equals": "Microsoft.Automation/automationAccounts/webhooks"
              },
              {
                "value": "
[length(field('Microsoft.Automation/automationAccounts/webhooks/runOn'))]",
                "less": 1
              }
            ]
          },
          {
            "allOf": [
              {
                "field": "type",
                "equals":
"Microsoft.Automation/automationAccounts/jobSchedules"
              },
              {
                "value": "
[length(field('Microsoft.Automation/automationAccounts/jobSchedules/runOn'))]",
                "less": 1
              }
            ]
          }
        ]
      },
      "then": {
        "effect": "[parameters('effectType')]"
      }
    }
  }
}
```

2. Ejecute el siguiente comando de Azure PowerShell o la CLI de Azure para crear una definición de directiva mediante el archivo AuditAutomationHRWJobExecution.json.

CLI de Azure

Azure CLI

```
az policy definition create --name 'audit-enforce-jobs-on-automation-
hybrid-runbook-workers' --display-name 'Audit Enforce Jobs on Automation
Hybrid Runbook Workers' --description 'This policy enforces job execution
```

```
on Automation account user Hybrid Runbook Workers.' --rules
'AuditAutomationHRWJobExecution.json' --mode All
```

El comando crea una definición de directiva denominada **Audit Enforce Jobs on Automation Hybrid Runbook Workers** (auditar la aplicación de trabajos en Automation Hybrid Runbook Workers). Para más información acerca de otros parámetros que puede utilizar, consulte creación de la definición de directivas az.

Cuando se llama sin parámetros de ubicación, `az policy definition create` elige de forma predeterminada guardar la definición de directiva en la suscripción seleccionada del contexto de sesiones. Para guardar la definición en una ubicación diferente, use los siguientes parámetros:

- **subscription**: se guarda en otra suscripción. Requiere un valor de *GUID* como identificador de suscripción o un valor de *cadena* como nombre.
- **management-group**: se guarda en un grupo de administración. Requiere un valor de *cadena*.

---

3. Después de crear la definición de directiva, puede crear una asignación de directiva mediante la ejecución de los siguientes comandos:

CLI de Azure

Azure CLI

```
az policy assignment create --name '<name>' --scope '<scope>' --policy
'<policy definition ID>'
```

El parámetro **scope** de `az policy assignment create` funciona con un grupo de administración, una suscripción, un grupo de recursos o un único recurso. El parámetro utiliza una ruta de acceso de recurso completa. El patrón de **scope** para cada contenedor es el siguiente. Reemplace `{rName}`, `{rgName}`, `{subId}` y `{mgName}` por el nombre del recurso, el nombre del grupo de recursos, el identificador de suscripción y el nombre del grupo de administración, respectivamente. `{rType}` se reemplazaría por el **tipo de recurso** del recurso como, por ejemplo, `Microsoft.Compute/virtualMachines` para una máquina virtual.

- Recurso `/subscriptions/{subID}/resourceGroups/{rgName}/providers/{rType}/{rName}`

- Grupo de recursos `/subscriptions/{subID}/resourceGroups/{rgName}`
- Suscripción `/subscriptions/{subID}`
- Grupo de administración `/providers/Microsoft.Management/managementGroups/{mgName}`

Puede obtener el id. de definición de Azure Policy si usa PowerShell con el comando siguiente:

Azure CLI

```
az policy definition show --name 'Audit Enforce Jobs on Automation Hybrid Runbook Workers'
```

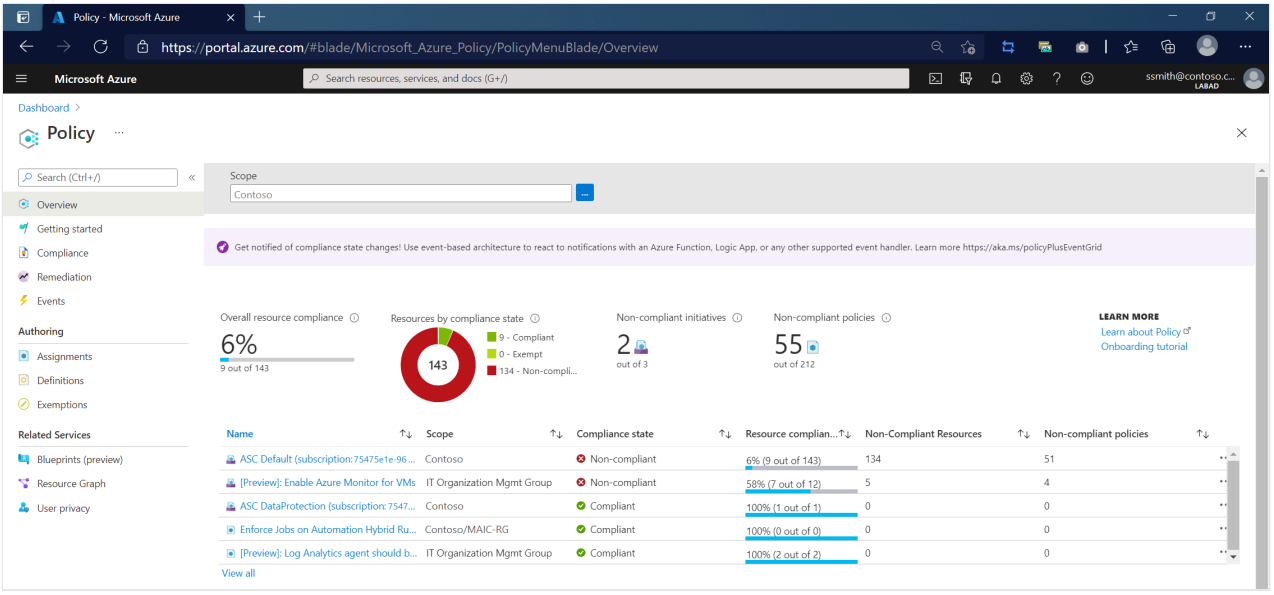
El identificador de definición de directiva para la definición de directiva que ha creado debe parecerse al ejemplo siguiente:

Resultados

```
"/subscription/<subscriptionId>/providers/Microsoft.Authorization/policyDefinitions/Audit Enforce Jobs on Automation Hybrid Runbook Workers"
```

---

4. Inicie sesión en Azure Portal.
5. Inicie el servicio Azure Policy en Azure Portal. Para ello, seleccione **Todos los servicios** y, a continuación, busque y seleccione **Directiva**.
6. Seleccione **Cumplimiento** en el panel izquierdo de la página. A continuación, busque la asignación de directiva que creó.



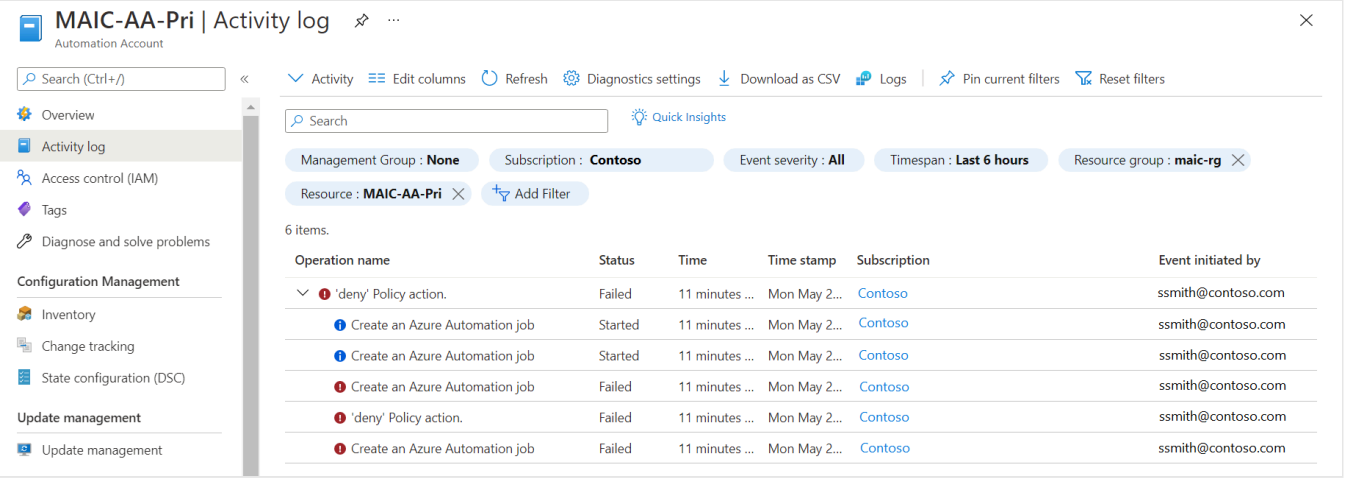
Cuando una de las operaciones REST de Automation se ejecuta sin hacer referencia a una instancia de Hybrid Runbook Worker en el cuerpo de la solicitud, se devuelve un código de respuesta 403 con un error similar al ejemplo siguiente que indica que la operación intentó ejecutarse en un espacio aislado de Azure:

rest

```
{
  "error": {
    "code": "RequestDisallowedByPolicy",
    "target": "Start_VMS",
    "message": "Resource 'Start_VMS' was disallowed by policy. Policy identifiers:
'[{\"policyAssignment\":{\"name\":\"Enforce Jobs on Automation Hybrid Runbook
Workers\",\"id\":\"/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/MAIC-
RG/providers/Microsoft.Authorization/policyAssignments/fd5e2cb3842d4eefbc857917\"},\
"policyDefinition\":{\"name\":\"Enforce Jobs on Automation Hybrid Runbook
Workers\",\"id\":\"/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/providers/Microsoft.Authorization/policyDefinitions/bbbb1b1b-cc2c-dd3d-
ee4e-ffffff5f5f5f\"}}]'.",
    "additionalInfo": [
      {
        "type": "PolicyViolation",
        "info": {
          "policyDefinitionDisplayName": "Enforce Jobs on Automation Hybrid Runbook
Workers",
          "evaluationDetails": {
            "evaluatedExpressions": [
              {
                "result": "True",
                "expressionKind": "Field",
                "expression": "type",
                "path": "type",
                "expressionValue": "Microsoft.Automation/automationAccounts/jobs",
                "targetValue": "Microsoft.Automation/automationAccounts/jobs",
                "operator": "Equals"
              },
              {
                "result": "True",
                "expressionKind": "Value",
                "expression": "
[length(field('Microsoft.Automation/automationAccounts/jobs/runOn'))]",
                "expressionValue": 0,
                "targetValue": 1,
                "operator": "Less"
              }
            ]
          },
          "policyDefinitionId": "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/providers/Microsoft.Authorization/policyDefinitions/bbbb1b1b-cc2c-dd3d-
ee4e-ffffff5f5f5f",
          "policyDefinitionName": "bbbb1b1b-cc2c-dd3d-ee4e-fffff5f5f5f",
          "policyDefinitionEffect": "Deny",
```

```
          "policyAssignmentId": "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/MAIC-
RG/providers/Microsoft.Authorization/policyAssignments/fd5e2cb3842d4eefbc857917",
          "policyAssignmentName": "fd5e2cb3842d4eefbc857917",
          "policyAssignmentDisplayName": "Enforce Jobs on Automation Hybrid Runbook
Workers",
          "policyAssignmentScope": "/subscriptions/aaaa0a0a-bb1b-cc2c-dd3d-
eeeeee4e4e4e/resourceGroups/MAIC-RG",
          "policyAssignmentParameters": {}
        }
      }
    ]
  }
}
```

La operación que se intentó realizar también se agrega en el registro de actividad de la cuenta de Automation, de manera similar al ejemplo siguiente.



Pasos siguientes

Para trabajar con runbooks, consulte Administración de runbooks en Azure Automation.

Last updated on 17/11/2025

Uso de la integración del control de código fuente

Artículo • 08/05/2025

La integración del control de código fuente en Azure Automation permite la sincronización en una sola dirección desde el repositorio de control de código fuente. El control de código fuente permite mantener actualizados los runbooks de la cuenta de Automation con los scripts del repositorio de control de código fuente de GitHub o de Azure DevOps. Esta característica facilita la promoción de código que se ha probado en el entorno de desarrollo a la cuenta de Automation de producción.

La integración del control de código fuente permite colaborar fácilmente con el equipo, realizar el seguimiento de los cambios y revertir a versiones anteriores de los runbooks. Por ejemplo, el control de código fuente permite sincronizar distintas ramas de control de código fuente con las cuentas de Automation de desarrollo, prueba o producción.

> **Nota**
>
> Los trabajos de sincronización de control de código fuente se ejecutan en la cuenta de Automation del usuario y se facturan con la misma tarifa que otros trabajos de Automation. Además, los trabajos de Azure Automation no admiten MFA (Multi-Factor Authentication).

Tipos de control de código fuente

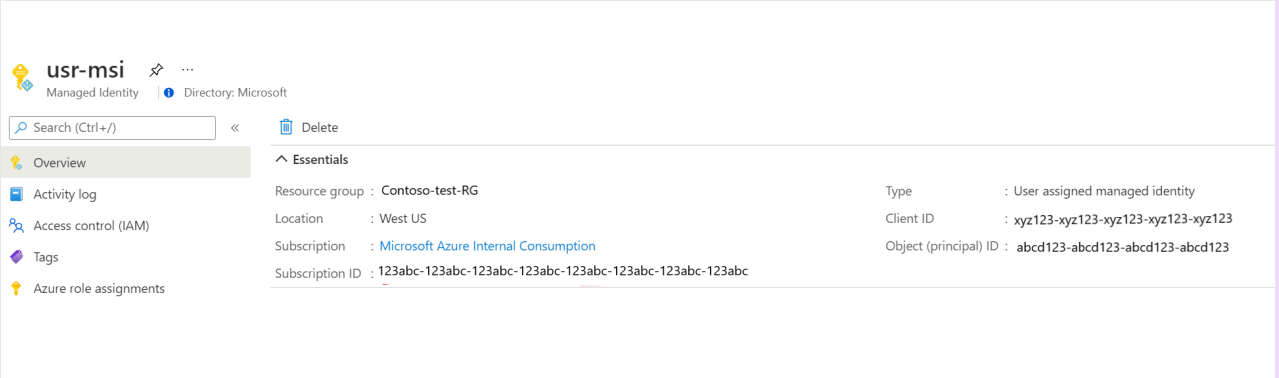
Azure Automation admite tres tipos de control de código fuente:

- GitHub
- Azure DevOps (Git)
- Azure DevOps (TFVC)

Requisitos previos

- Un repositorio de control de código fuente (GitHub o Azure DevOps)
- La cuenta de Automation requiere una identidad administrada asignada por el sistema o asignada por el usuario. Si no configuró una identidad administrada con la cuenta de Automation, consulte Habilitación de identidad administrada asignada por el sistema o Habilitación de identidad administrada asignada por el usuario para crearla.

- Asigne la identidad administrada asignada por el usuario o por el sistema al rol Colaborador en la cuenta de Automation.

> **Nota**
>
> Azure Automation admite tanto la identidad administrada asignada por el sistema como la asignada por el usuario con la integración del control de código fuente. Para usar una identidad administrada asignada por el usuario, cree una variable de automatización `AUTOMATION_SC_USER_ASSIGNED_IDENTITY_ID` con el valor como el id. de cliente de la identidad asignada por el usuario. La `Managed Identity` asignada por el usuario se debe habilitar y debe tener acceso de colaborador a la cuenta de Automation. De manera predeterminada, si esta variable no está creada, usamos la identidad asignada por el sistema.
>
> 
>
>
> Si tiene habilitadas la cuenta de ejecución y la identidad administrada, se le da preferencia a la identidad administrada.

> **Importante**
>
> La cuenta de ejecución de Azure Automation se ha retirado el **30 de septiembre de 2023**. Se recomienda usar identidades administradas.

> **Nota**
>
> Según esta documentación de Azure DevOps, la directiva **Acceso a la aplicación de terceros a través de OAuth** está **desactivada** de forma predeterminada para todas las organizaciones nuevas. Por lo tanto, si intenta configurar el control de código fuente en Azure Automation con **Azure Devops (Git)** como tipo de control de código fuente sin habilitar **Acceso a la aplicación de terceros a través de OAuth** en el icono Directivas de Configuración de la organización en Azure DevOps, es posible que obtenga un error **Token de seguridad SourceControl no válido**. Por lo tanto, para evitar este error,

asegúrese de habilitar primero la opción **Acceso a la aplicación de terceros a través de OAuth** en el icono Directivas de Configuración de la organización en Azure DevOps.

Configurar el control de código fuente

En esta sección se explica cómo configurar el control de código fuente para la cuenta de Automation. Puede usar Azure Portal o PowerShell.

Asignación de una identidad administrada al rol Colaborador

En este ejemplo se usa Azure PowerShell para mostrar cómo asignar el rol Colaborador en la suscripción al recurso de la cuenta de Azure Automation.

1. Abra una consola de PowerShell con privilegios elevados.
2. Inicie sesión en Azure para ejecutar el comando `Connect-AzAccount`.
3. Para asignar la identidad administrada al rol **Colaborador**, ejecute el siguiente comando.

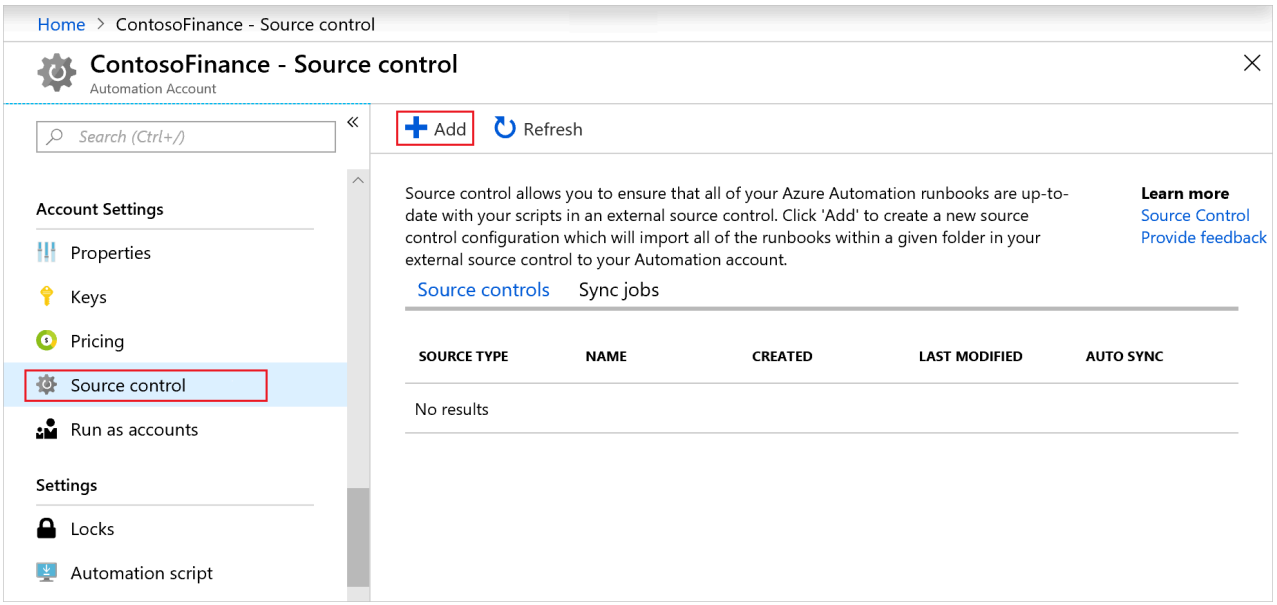
PowerShell

```
New-AzRoleAssignment `
    -ObjectId <automation-Identity-Object(Principal)-Id> `
    -Scope
"/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers
/Microsoft.Automation/automationAccounts/{automationAccountName}" `
    -RoleDefinitionName "Contributor"
```

Configuración del control de código fuente en Azure Portal

Utilice este procedimiento para configurar el control de código fuente mediante Azure Portal.

1. En la cuenta de Automation, seleccione **Control de código fuente** y haga clic en **Agregar**.



2. Elija **Tipo de control de código fuente** y, a continuación, haga clic en **Autenticar**.

3. Se abre una ventana del explorador y se le pide que inicie sesión. Siga las indicaciones para completar la autenticación.

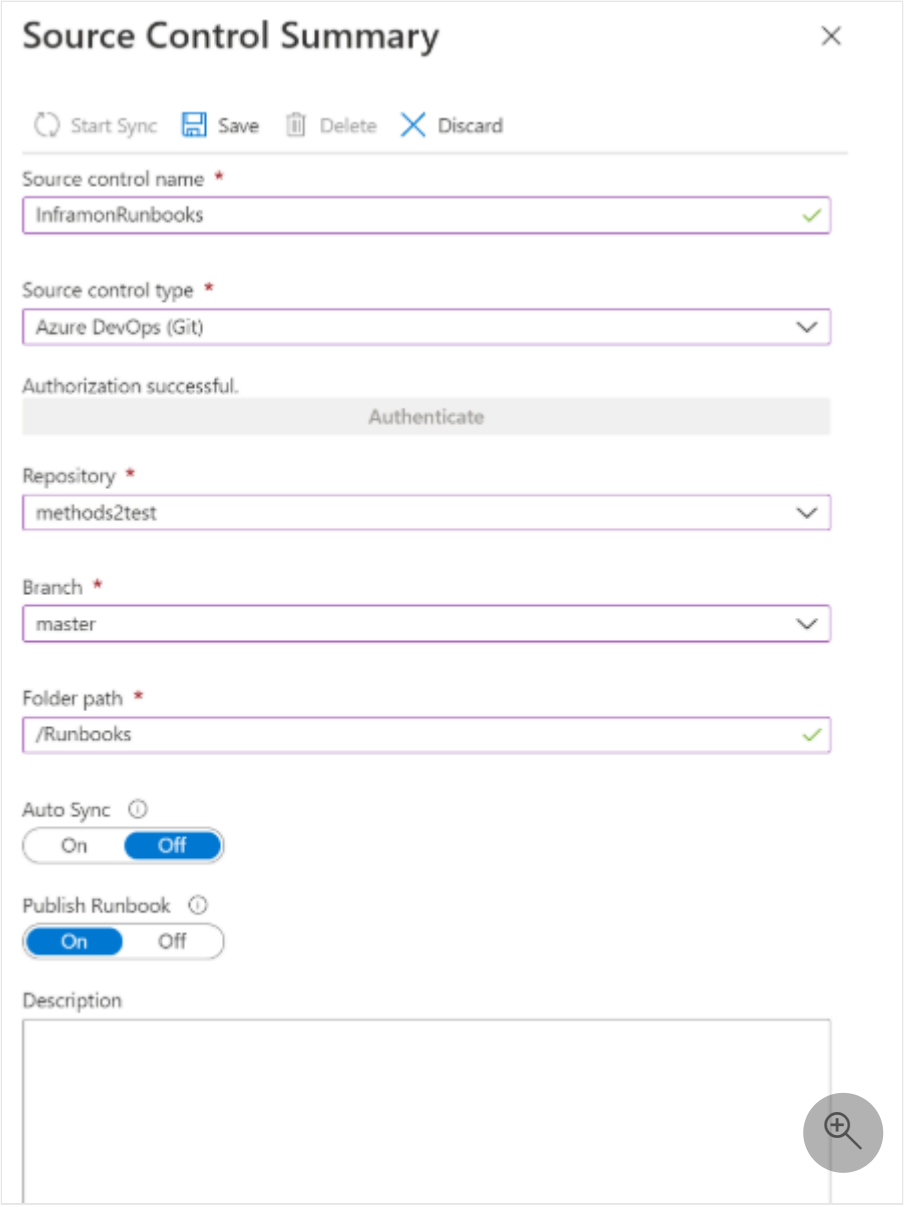
4. En la página Resumen del control de código fuente, utilice los campos para rellenar las propiedades del control de código fuente definidas a continuación. Haga clic en **Guardar** cuando termine.

Expandir tabla

Propiedad	Descripción
Nombre del control de código fuente	Nombre descriptivo del control de código fuente. Este nombre solo puede contener letras y números.
Tipo de control de código fuente	Tipo de mecanismo de control de código fuente. Las opciones disponibles son: * GitHub * Azure DevOps (Git) * Azure DevOps (TFVC)
Repositorio	Nombre del repositorio o proyecto. Se recuperan los primeros 200 repositorios. Para buscar un repositorio, escriba el nombre en el campo y haga clic en **Search on GitHub** (Buscar en GitHub).
Rama	Rama de la que se van a extraer los archivos de código fuente. No se permite especificar una rama para el tipo de control de código fuente TFVC.
Folder path (Ruta de acceso a la carpeta)	Carpeta que contiene los runbooks que se van a sincronizar; por ejemplo, **/Runbooks**. Solo se sincronizan los runbooks de la carpeta especificada. No se admite la recursividad.

Propiedad	Descripción
Sincronización automática[1]	Opción que activa o desactiva la sincronización automática cuando se realiza una confirmación en el repositorio de control de código fuente o en el de GitHub.
Publicar runbook	Se debe establecer en Activado si los runbooks se publican automáticamente después de la sincronización desde el control de código fuente y en Desactivado de otro modo.
Descripción	Texto que especifica información adicional sobre el control de código fuente.

1 Para habilitar la sincronización automática al configurar la integración del control de código fuente con Azure DevOps, debe ser un administrador del proyecto o el propietario del repositorio de GitHub. Los colaboradores solo pueden configurar el control de código fuente sin sincronización automática.La sincronización automática de
no funciona con Private Link de Automation. Si habilita Private Link, se producirá un error en las invocaciones de webhook de control de código fuente, ya que está fuera de la red.



Nota

- El inicio de sesión en el repositorio del control de código fuente podría ser diferente al inicio de sesión en Azure Portal. Al configurar el control de código fuente, asegúrese de que ha iniciado sesión con la cuenta correcta en el repositorio del control de código fuente. Si tiene dudas, abra una nueva pestaña en el explorador, cierre la sesión de **dev.azure.com**, **visualstudio.com** o **github.com**, y vuelva a intentar establecer la conexión con el control de código fuente.
- No se admite la autenticación entre inquilinos.
- La sincronización automática puede producir un error si el control de código fuente se creó hace más de un año, ya que el webhook usado para invocar el control de código fuente expira después de un año. Para resolver este problema, cree un nuevo control de código fuente en la cuenta de Automation con la misma configuración, que generará un nuevo webhook con una fecha de expiración extendida.

Configuración del control de código fuente en PowerShell

También puede usar PowerShell para configurar el control de código fuente en Azure Automation. Para usar cmdlets de PowerShell para esta operación, necesita un token de acceso personal (PAT). Utilice el cmdlet New-AzAutomationSourceControl para crear la conexión al control de código fuente. Este cmdlet requiere una cadena segura para el PAT. Para obtener información sobre cómo crear una cadena segura, consulte ConvertTo-SecureString.

En las subsecciones siguientes se muestra la creación de PowerShell de la conexión del control de código fuente para GitHub, Azure DevOps (Git) y Azure DevOps (TFVC).

Creación de la conexión del control de código fuente para GitHub

PowerShell

```
New-AzAutomationSourceControl -Name SCGitHub -RepoUrl
https://github.com/<accountname>/<reponame>.git -SourceType GitHub -FolderPath
"/MyRunbooks" -Branch main -AccessToken <secureStringofPAT> -ResourceGroupName
<ResourceGroupName> -AutomationAccountName <AutomationAccountName>
```

Creación de la conexión del control de código fuente para Azure DevOps (Git)

> **Nota**
>
> Azure DevOps (Git) usa una dirección URL que accede a **dev.azure.com** en lugar de **visualstudio.com**, que se usaba en formatos anteriores. El formato de dirección URL anterior `https://<accountname>.visualstudio.com/<projectname>/_git/<repositoryname>` está en desuso pero todavía se admite, aunque se prefiere el nuevo formato.

PowerShell

```
New-AzAutomationSourceControl -Name SCReposGit -RepoUrl
https://dev.azure.com/<accountname>/<adoprojectname>/_git/<repositoryname> -
SourceType VsoGit -AccessToken <secureStringofPAT> -Branch main -ResourceGroupName
<ResourceGroupName> -AutomationAccountName <AutomationAccountName> -FolderPath
"/Runbooks"
```

Creación de la conexión del control de código fuente para Azure DevOps (TFVC)

> **Nota**
>
> Azure DevOps (TFVC) usa una dirección URL que accede a **dev.azure.com** en lugar de **visualstudio.com**, que se usaba en formatos anteriores. El formato de dirección URL anterior `https://<accountname>.visualstudio.com/<projectname>/_versionControl` está en desuso pero todavía se admite, aunque se prefiere el nuevo formato.

PowerShell

```
New-AzAutomationSourceControl -Name SCReposTFVC -RepoUrl
https://dev.azure.com/<accountname>/<adoprojectname>/_git/<repositoryname> -
SourceType VsoTfvc -AccessToken <secureStringofPAT> -ResourceGroupName
<ResourceGroupName> -AutomationAccountName <AutomationAccountName> -FolderPath
"/Runbooks"
```

Permisos del token de acceso personal (PAT)

El control de código fuente requiere algunos permisos mínimos para los tokens de acceso personal. Las subsecciones siguientes contienen los permisos mínimos necesarios para GitHub y Azure DevOps.

Permisos mínimos de PAT para GitHub

En la tabla siguiente se definen los permisos mínimos necesarios de PAT para GitHub. Para más información sobre cómo crear un token de acceso personal en GitHub, consulte Creación de un token de acceso personal para la línea de comandos.

Expandir tabla

Ámbito	Descripción
`repo`	
`repo:status`	Acceder al estado de confirmación
`repo_deployment`	Acceder al estado de implementación
`public_repo`	Acceder a los repositorios públicos
`repo:invite`	Acceder a invitaciones de repositorio
`security_events`	Leer y escribir eventos de seguridad
`admin:repo_hook`	

Ámbito	Descripción
`write:repo_hook`	Escribir los enlaces de repositorio
`read:repo_hook`	Leer los enlaces de repositorio

Permisos mínimos de PAT para Azure DevOps

En la tabla siguiente se definen los permisos mínimos necesarios de PAT para Azure DevOps. Para más información sobre cómo crear un token de acceso personal en Azure DevOps, consulte Autenticación del acceso con tokens de acceso personal.

Expandir tabla

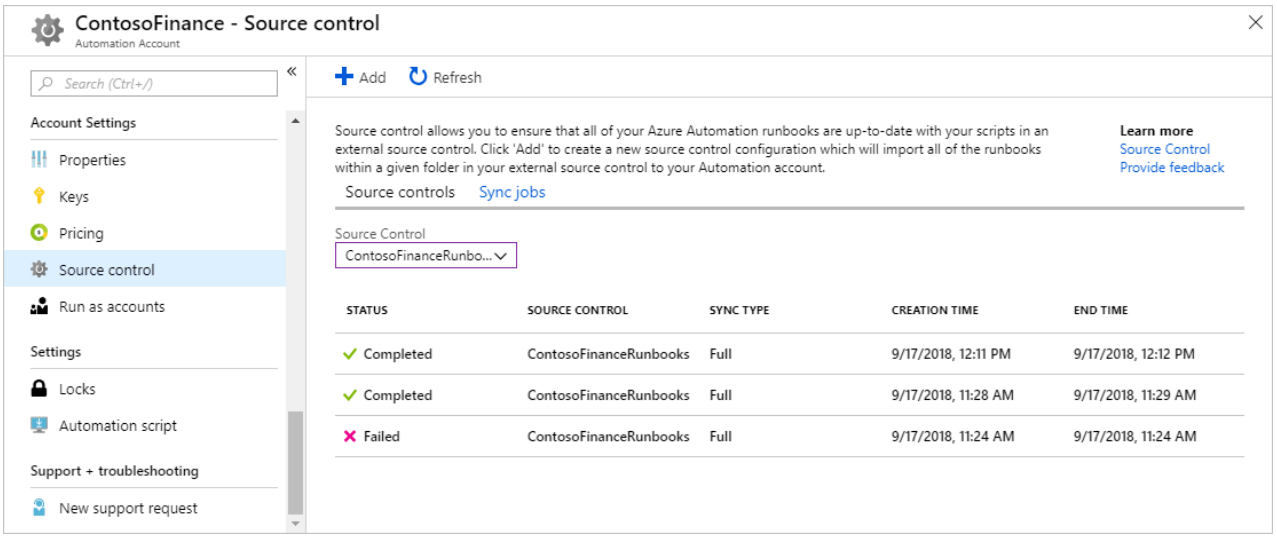
Ámbito	Tipo de acceso
`Code`	Lectura
`Project and team`	Lectura
`Identity`	Lectura
`User profile`	Lectura
`Work items`	Lectura
`Service connections`	Leer, consultar, administrar[1]

[1] El permiso `Service connections` solo es necesario si ha habilitado la sincronización automática.

Sincronización con el control de código fuente

Siga estos pasos para sincronizar con el control de código fuente.

1. Seleccione el código fuente en la tabla de la página Control de código fuente.
2. Haga clic en **Iniciar sincronización** para iniciar el proceso de sincronización.
3. Para ver el estado del trabajo de sincronización actual o de los anteriores, haga clic en la pestaña **Trabajos de sincronización**.
4. En el menú desplegable **Control de código fuente**, seleccione un mecanismo de control de código fuente.



5. Al hacer clic en un trabajo, verá la salida del trabajo. El siguiente es un ejemplo de la salida de un trabajo de sincronización de control de código fuente.

Resultados

```
=================================================================

Azure Automation Source Control.
Supported runbooks to sync: PowerShell Workflow, PowerShell Scripts, DSC
Configurations, Graphical, and Python 2.

Setting AzEnvironment.

Getting AzureRunAsConnection.

Logging in to Azure...

Source control information for syncing:

[Url =
https://ContosoExample.visualstudio.com/ContosoFinanceTFVCExample/_versionCon
trol] [FolderPath = /Runbooks]

Verifying url:
https://ContosoExample.visualstudio.com/ContosoFinanceTFVCExample/_versionCon
trol

Connecting to VSTS...

Source Control Sync Summary:

2 files synced:
 - ExampleRunbook1.ps1
 - ExampleRunbook2.ps1

=================================================================
```

6. Puede encontrar opciones de registro adicionales si selecciona **Todos los registros** en la página Resumen del trabajo de sincronización del control de código fuente. Estas entradas de registro adicionales pueden ayudar a solucionar los problemas que pueden surgir al usar el control de código fuente.

Desconexión del control de código fuente

Para desconectarse de un repositorio de control de código fuente:

1. Abra **Control de código fuente** en **Configuración de la cuenta** de la cuenta de Automation.

2. Seleccione el mecanismo de control de código fuente que desea eliminar.

3. En la página Resumen del trabajo de sincronización del control de código fuente, haga clic en **Eliminar**.

Control de los problemas de codificación

Si varias personas editan runbooks en el repositorio de control de código fuente con diferentes editores, pueden darse problemas de codificación. Para más información sobre esta situación, consulte Causas comunes de problemas de codificación.

Actualización del token de acceso personal

Actualmente, no puede usar Azure Portal para actualizar el token de acceso personal en el control de código fuente. Después de que el token de acceso personal haya expirado o se haya revocado, podrá actualizar el control de código fuente con un nuevo token de acceso de las siguientes maneras:

- Mediante la API REST.
- Mediante el cmdlet Update-AzAutomationSourceControl.

Pasos siguientes

- Para integrar el control de código fuente del runbook con Visual Studio Codespaces, consulte Azure Automation: integración del control de código fuente de runbook mediante Visual Studio Codespaces.

Información general de Azure Automation State Configuration

> **Nota:**
>
> Azure Automation State Configuration se retirará el 30 de septiembre de 2027, realice la transición a Azure Machine Configuration por esa fecha. Para obtener más información, consulte el anuncio del blog. El servicio Azure Machine Configuration combina características de la extensión de DSC, Azure Automation State Configuration y las características más solicitadas de los comentarios de los clientes. Azure Machine Configuration también incluye compatibilidad con máquinas híbridas a través de servidores habilitados para Arc.

> **Importante**
>
> Los **vínculos de navegación Agregar**, **Redactar**y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> Azure Automation DSC para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

Azure Automation State Configuration es un servicio de administración de configuración de Azure que le permite escribir, administrar y compilar configuraciones de PowerShell Desired State Configuration (DSC) para nodo de cualquier centro de datos en la nube local. El servicio también importa recursos de DSC y asigna configuraciones a los nodos de destino, todos en la nube. Puede acceder a Azure Automation State Configuration en el portal de Azure seleccionando **Configuración de estado (DSC)** bajo **Gestión de la configuración**.

Puede usar Azure Automation State Configuration para administrar una variedad de máquinas:

- Azure máquinas virtuales
- máquinas virtuales de Azure (clásicas)

- Máquinas físicas o virtuales de Windows locales o en una nube distinta de Azure (incluidas las instancias de AWS EC2)
- Máquinas físicas o virtuales Linux locales, en Azure o en una nube distinta de Azure

Si no está listo para administrar la configuración de la máquina desde la nube, puede usar Azure Automation State Configuration como un punto de conexión solo para informes. Esta característica permite establecer (insertar) configuraciones a través de DSC y ver los detalles de informes en Azure Automation.

> **Nota:**
>
> La administración de máquinas virtuales de Azure con Azure Automation State Configuration se incluye sin cargo adicional si la versión de la extensión Desired State Configuration instalada en la máquina virtual de Azure es superior a la 2.70. Consulte la página de precios de Automation para más información.

Por qué usar Azure Automation State Configuration

Azure Automation State Configuration proporciona varias ventajas sobre el uso de DSC fuera de Azure. Este servicio permite la escalabilidad en miles de máquinas de forma rápida y sencilla desde una ubicación central y segura. Puede habilitar fácilmente máquinas, asignarles configuraciones declarativas y ver informes que muestran el cumplimiento por parte de cada máquina del estado deseado que se especifique.

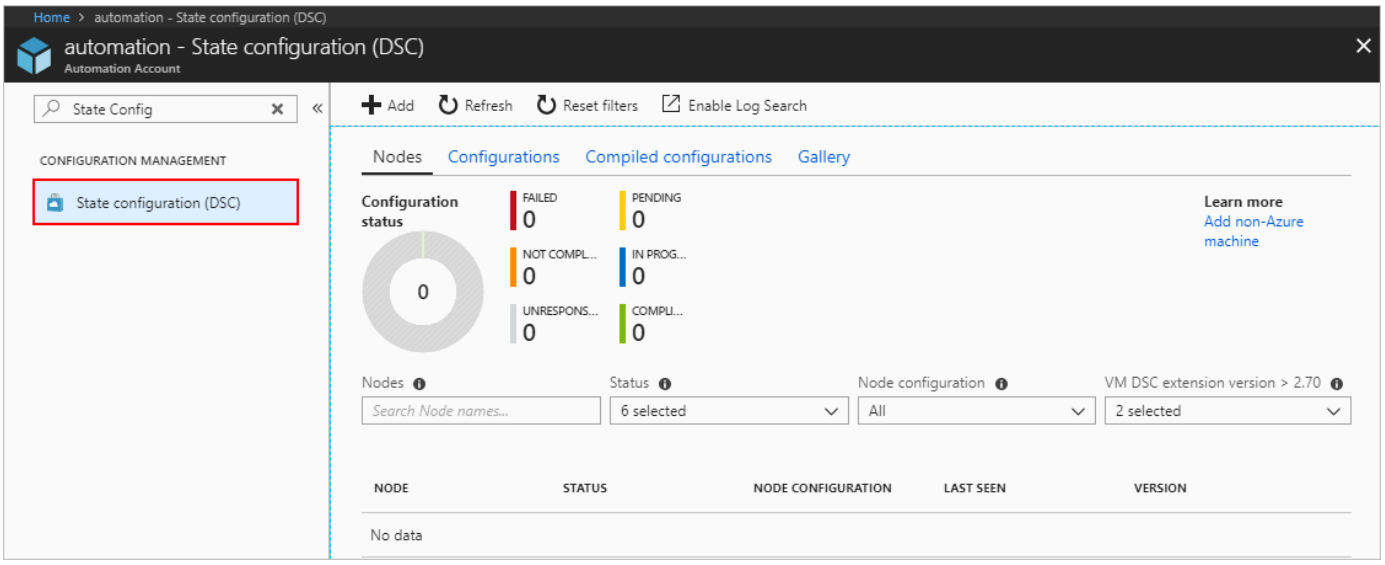
Azure Automation State Configuration es a DSC lo que los runbooks de Azure Automation son al scripting de PowerShell. En otras palabras, de la misma manera que Azure Automation le ayuda a administrar scripts de PowerShell, también le ayuda a administrar las configuraciones de DSC.

Servidor de extracción integrado

Azure Automation State Configuration proporciona un servidor pull de DSC similar a la característica de Windows DSC-Service. Los nodos de destino pueden recibir automáticamente configuraciones, ajustarse al estado deseado y enviar informes de cumplimiento. El servidor de extracción integrado en Azure Automation elimina la necesidad de configurar y mantener su propio servidor de extracción. Azure Automation puede tener como destino máquinas virtuales o físicas Windows o Linux, en la nube o en el entorno local.

Administración de todos los artefactos de DSC

Azure Automation State Configuration aporta la misma capa de gestión a PowerShell Desired State Configuration que ofrece para el scripting en PowerShell. Desde el portal de Azure o desde PowerShell, puede administrar todas las configuraciones, recursos y nodos de destino de DSC.



Importación de datos de informes en registros de Azure Monitor

Los nodos administrados con Azure Automation State Configuration envían datos de estado de informes detallados al servidor de extracción integrado. Puede configurar Azure Automation State Configuration para enviar estos datos al área de trabajo de Log Analytics. Consulte cómo reenviar los datos de informes de la configuración de estado de Azure Automation a los registros de Azure Monitor.

Requisitos previos

Tenga en cuenta los requisitos de esta sección al usar Azure Automation State Configuration.

Requisitos del sistema operativo

> **Nota:**
>
> Windows Server 2008, 2008 R2, 2012 y 2012 R2 alcanzaron el fin del soporte (EOS). Revise el uso y planee las actualizaciones y migraciones del sistema operativo en consecuencia. Para obtener más información, consulte Fin del soporte técnico para:
>
> - Windows Server 2008 y Windows Server 2008 R2
> - Windows Server 2012
> - Windows Server 2012 R2

Actualización local a Windows Server 2016, 2019, 2022 o 2025.

En el caso de los nodos que ejecutan Windows, se admiten las siguientes versiones:

- Windows Server 2022
- Windows Server 2019
- Windows Server 2016
- Windows Server 2012R2
- Windows Server 2012
- Windows Server 2008 R2 SP1
- Windows 10
- Windows 8.1
- Windows 7

> **Nota:**
>
> La SKU de producto independiente Microsoft Hyper-V Server no contiene una implementación de DSC. Por lo tanto, no se puede gestionar mediante DSC de PowerShell o la Configuración de Estado de Azure Automation.

Para los nodos que ejecutan Linux, la extensión DSC de Linux admite todas las distribuciones de Linux que se enumeran en la documentación de DSC de PowerShell.

Requisitos de DSC

Para todos los nodos de Windows que se ejecutan en Azure, se instala WMF 5.1 cuando se habilitan las máquinas. En el caso de los nodos que ejecutan Windows Server 2012 y Windows 7, WinRM está habilitado.

Para todos los nodos de Linux que se ejecutan en Azure, PowerShell DSC para Linux se instala cuando las máquinas están habilitadas.

Configuración de redes privadas

Vea Configuración de red de Azure Automation para obtener información detallada sobre los puertos, las direcciones URL y otros detalles de red necesarios para los nodos de una red privada.

Compatibilidad con proxy

La compatibilidad con proxy para el agente DSC está disponible en Windows versión 1809 y posteriores. Para habilitar esta opción, establezca los valores de las propiedades `ProxyURL` y `ProxyCredential` en el script de metaconfiguraciones que se utiliza para registrar los nodos.

> **Nota:**
>
> Azure Automation State Configuration no proporciona compatibilidad con el proxy DSC para versiones anteriores de Windows.

Para los nodos de Linux, el agente de DSC admite el proxy y utiliza la variable `http_proxy` para determinar la dirección URL. Para obtener más información sobre el soporte con proxy, vea Generación de metaconfiguraciones de DSC.

Registros DNS por región

Se recomienda usar las direcciones que aparecen en la tabla Registros DNS por región al definir excepciones.

Pasos siguientes

- Para empezar, consulte Get started with Azure Automation State Configuration.
- Para obtener información sobre cómo habilitar nodos, consulte Enable Azure Automation State Configuration.
- Para obtener información sobre la compilación de configuraciones de DSC para poder asignarlas a nodos de destino, consulte Compilar configuraciones de DSC en Azure Automation State Configuration.
- Para ver un ejemplo del uso de Azure Automation State Configuration en una canalización de implementación continua, consulte Configuración de la implementación continua con Chocolatey.
- Para obtener información sobre los precios, consulte los precios de Azure Automation State Configuration.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.

Last updated on 05/05/2026

Introducción a Azure Automation State Configuration

> **Nota:**
>
> Azure Automation State Configuration se retirará el 30 de septiembre de 2027, realice la transición a Azure Machine Configuration por esa fecha. Para obtener más información, consulte el anuncio de la entrada de blog. El servicio Azure Machine Configuration combina características de la extensión de DSC, Azure Automation State Configuration y las características más solicitadas de los comentarios de los clientes. Azure Machine Configuration también incluye compatibilidad con máquinas híbridas a través de servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> Azure Automation DSC para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

En este artículo se proporciona una guía paso a paso para realizar las tareas más comunes con Azure Automation State Configuration, como:

- crear, importar y compilar configuraciones
- habilitación de máquinas para administrar
- ver informes

Para obtener una visión general sobre la Configuración del Estado, consulte State Configuration overview. Para obtener la documentación de "Desired State Configuration" (DSC), consulte Windows PowerShell Desired State Configuration Overview.

Si desea un entorno de ejemplo que ya esté configurado sin seguir los pasos descritos en este artículo, puede usar la plantilla de nodo administrado Azure Automation. Esta plantilla configura un entorno completo de State Configuration (DSC), incluida una máquina virtual de Azure administrada por State Configuration (DSC).

Requisitos previos

Para completar los ejemplos de este artículo, se requiere lo siguiente:

- Una cuenta de Azure Automation. Para más información sobre las cuentas de Automation y sus requisitos, consulte Introducción a la autenticación de cuentas de Automation.
- Una máquina virtual Azure Resource Manager (no clásica) que ejecuta un sistema operativo compatible. Para obtener instrucciones sobre cómo crear una máquina virtual, consulte Crear la primera máquina virtual Windows en el portal de Azure

Creación de una configuración de DSC

Puede crear una configuración simple DSC que garantice la presencia o ausencia de la característica **Web-Server** Windows (IIS), en función de cómo asigne nodos.

Los nombres de configuración de Azure Automation deben limitarse a no más de 100 caracteres.

1. Inicie VSCode (o cualquier editor de texto).

2. Escriba el siguiente texto:

PowerShell

```
configuration TestConfig
{
    Node IsWebServer
    {
        WindowsFeature IIS
        {
            Ensure               = 'Present'
            Name                 = 'Web-Server'
            IncludeAllSubFeature = $true
        }
    }

    Node NotWebServer
    {
        WindowsFeature IIS
        {
            Ensure               = 'Absent'
            Name                 = 'Web-Server'
```

```
        }
    }
}
```

3. Guarde el archivo como **TestConfig.ps1**.

Esta configuración llama a un recurso en cada bloque de nodo, el recurso WindowsFeature. Este recurso garantiza la presencia o ausencia de la característica **Web-Server**.

Importación de una configuración en Azure Automation

A continuación, debe importar la configuración en la cuenta de Automation.

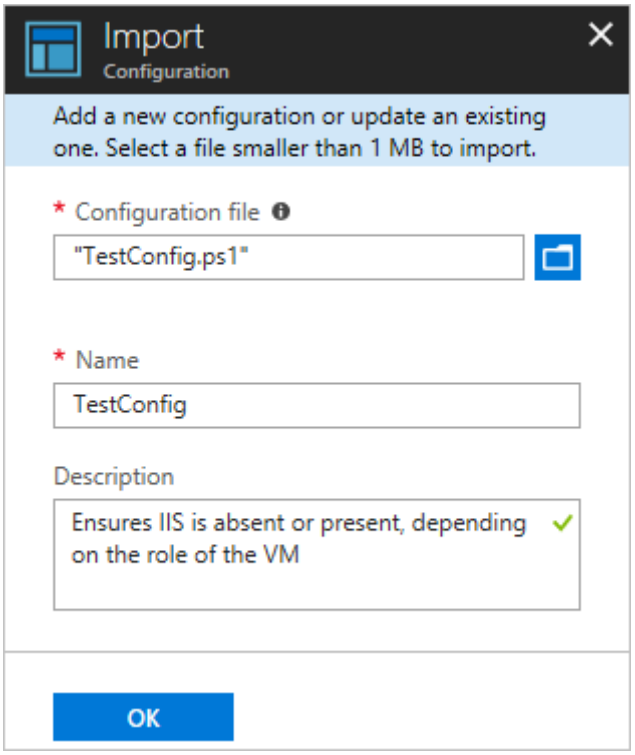
1. Inicie sesión en el portal Azure.

2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.

3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.

4. En la página State Configuration (DSC), seleccione la pestaña **Configuraciones** y, después, seleccione en **Agregar**.

5. En el panel Importar configuración, busque el archivo `TestConfig.ps1` en el equipo.



6. Seleccione **Aceptar**.

Visualización de una configuración en Azure Automation

Después de importar una configuración, puede verlo en el portal de Azure.

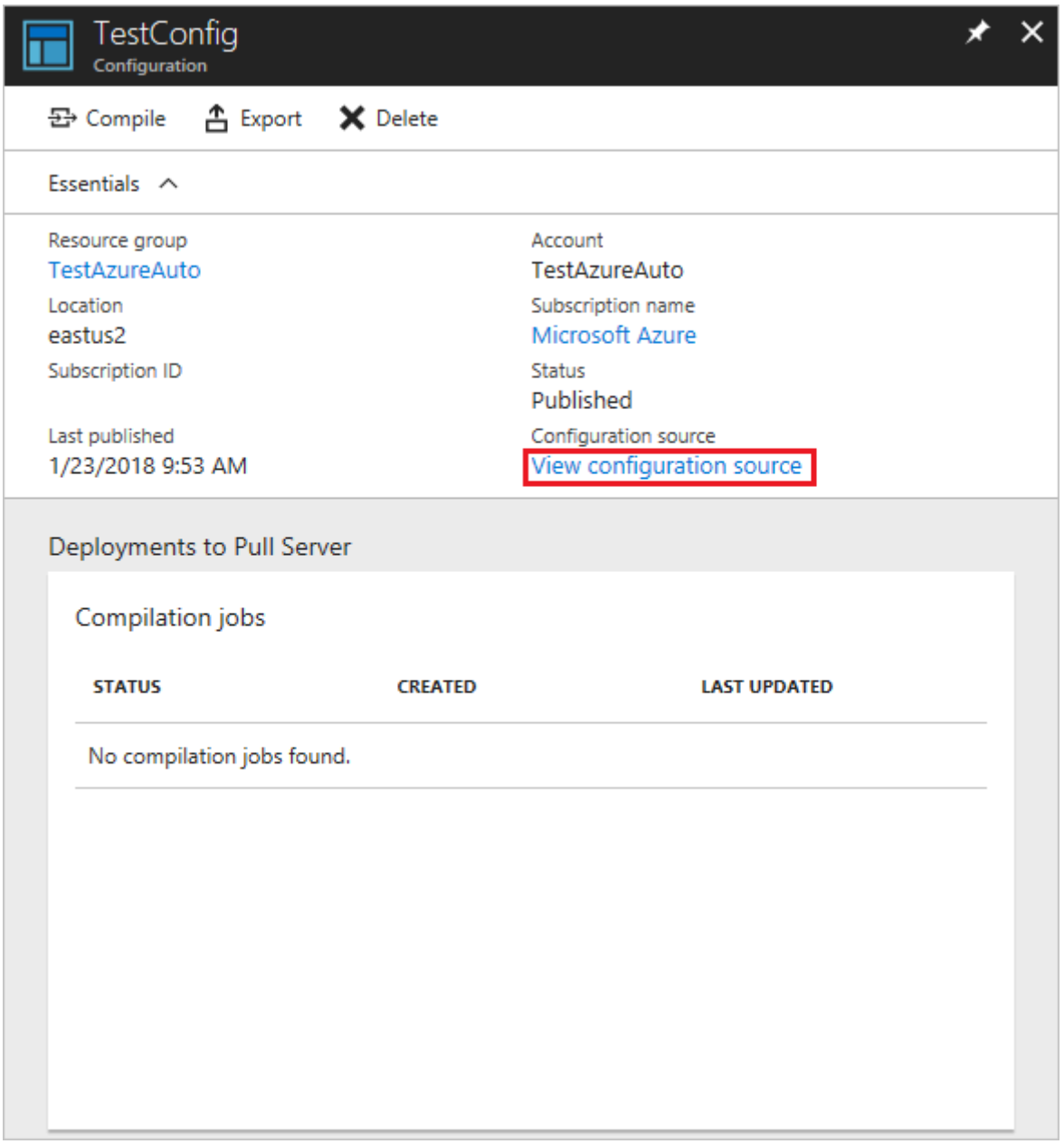
1. Inicie sesión en el portal Azure.

2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.

3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.

4. En la página State Configuration (DSC), seleccione la pestaña **Configuraciones** y, a continuación, seleccione el nombre de la configuración que importó en el procedimiento anterior, **TestConfig**.

5. En el panel Configuración de TestConfig, seleccione **Ver origen de configuración**.



Se abre un panel Origen de configuración de TestConfig, que muestra el código de PowerShell de la configuración.

Compilación de una configuración en Azure Automation

Para poder aplicar un estado deseado a un nodo, se debe compilar una configuración de DSC que defina dicho estado en una o varias configuraciones de nodo (documento de MOF) y se debe colocar en el servidor de extracción de DSC de Automatización. Para obtener una descripción más detallada de la compilación de configuraciones en State Configuration (DSC), consulte Compile configurations in Azure Automation State Configuration. Para más información sobre la compilación de configuraciones, consulte Configuraciones DSC.

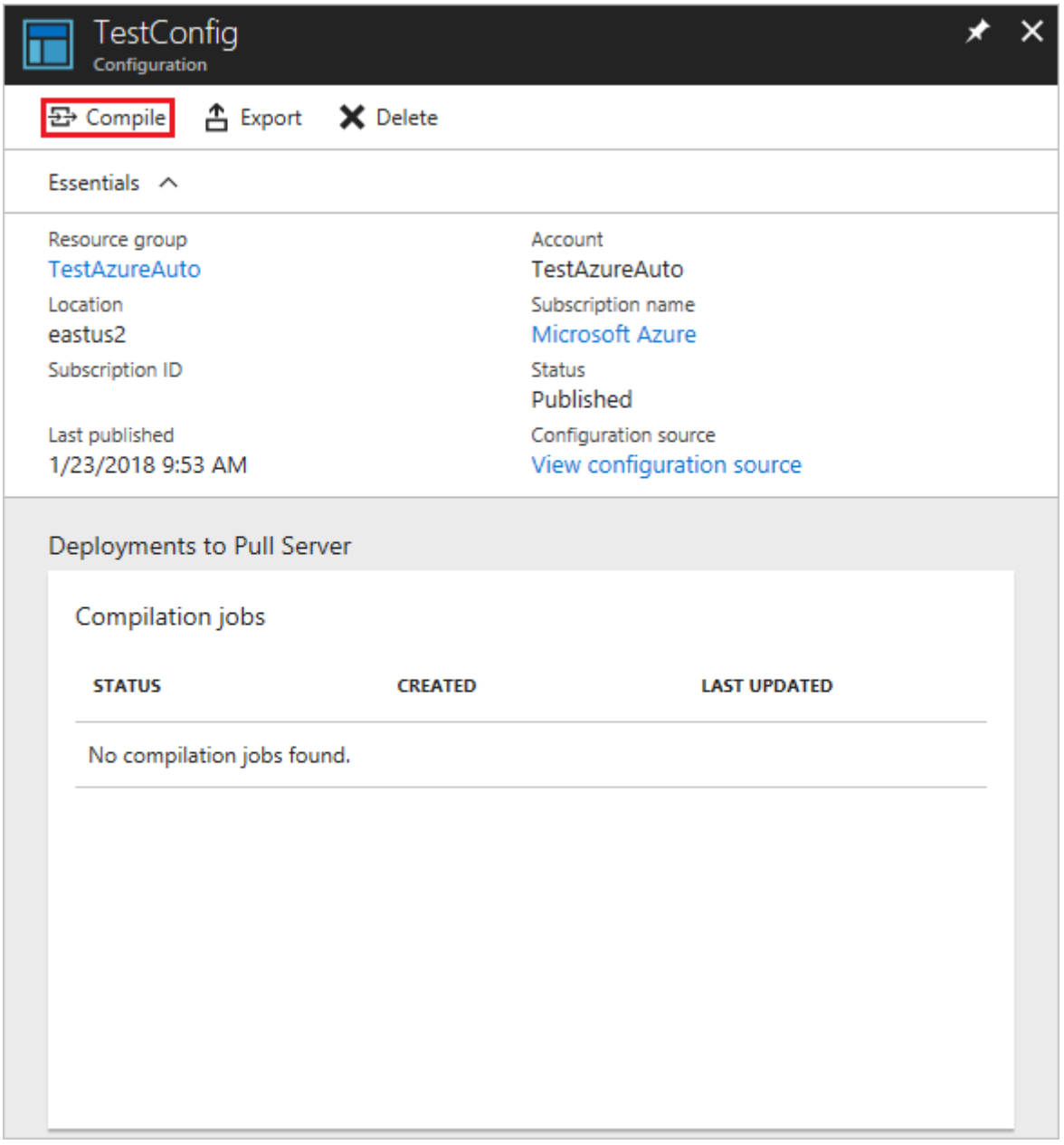
1. Inicie sesión en el portal Azure.

2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.

3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.

4. En la página State configuration (DSC), seleccione la pestaña **Configuraciones** y, a continuación, seleccione el nombre de la configuración importada anteriormente, **TestConfig**.

5. Para iniciar el trabajo de compilación, seleccione **Compilar** en el panel Configuración de TestConfig y, a continuación, seleccione **Sí**.



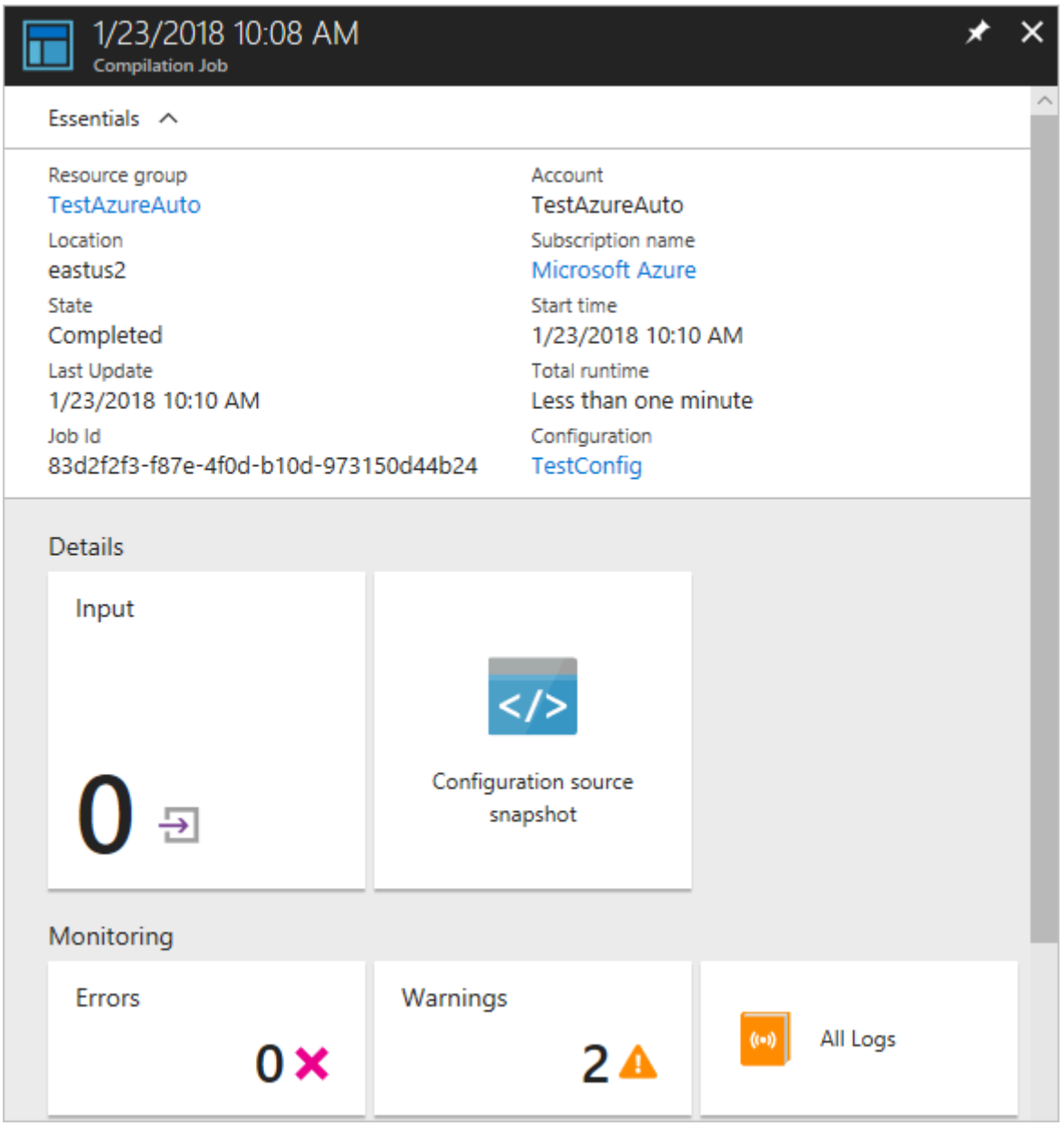
ⓘ **Nota:**

Al compilar una configuración en Azure Automation, implementa automáticamente los archivos MOF creados de configuración de nodo en el servidor pull.

Vista de un trabajo de compilación

Después de iniciar una compilación, puede verla en el icono **Trabajos de compilación** de la página **Configuración**. El icono **Compilation Jobs** (Trabajos de compilación) muestra los trabajos con errores, completados y en ejecución. Cuando abre el panel de un trabajo de compilación, se muestra información acerca del trabajo, incluidos los errores o advertencias encontrados, los parámetros de entrada usados en la configuración y los registros de compilación.

1. Inicie sesión en el portal Azure.
2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.
3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.
4. En la página State configuration (DSC), seleccione la pestaña **Configuraciones** y, a continuación, seleccione el nombre de la configuración importada anteriormente, **TestConfig**.
5. En **Trabajos de compilación**, seleccione el trabajo de compilación que quiere ver. Se abre el panel Trabajo de compilación, etiquetado con la fecha en que se inició el trabajo de compilación.



6. Para ver más detalles sobre el trabajo, seleccione cualquier icono en el panel Trabajo de compilación.

Vista de configuraciones de nodos

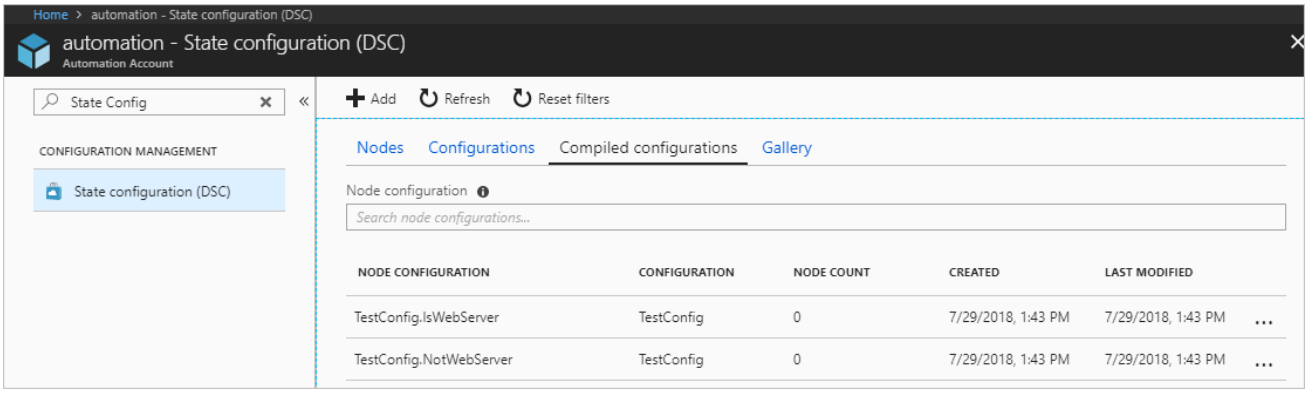
La finalización correcta de un trabajo de compilación crea una o varias configuraciones de nodo nuevas. Una configuración de nodo es un documento MOF que despliegas en el servidor de recopilación. Puede ver las configuraciones de nodos de su cuenta de Automation en la página State configuration (DSC). El formato del nombre de la configuración de un nodo es `ConfigurationName.NodeName`.

1. Inicie sesión en el portal Azure.

2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.

3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.

4. En la página State configuration (DSC), seleccione la pestaña **Configuraciones compiladas**.



Habilitación de una máquina virtual de Azure Resource Manager para la administración con State Configuration

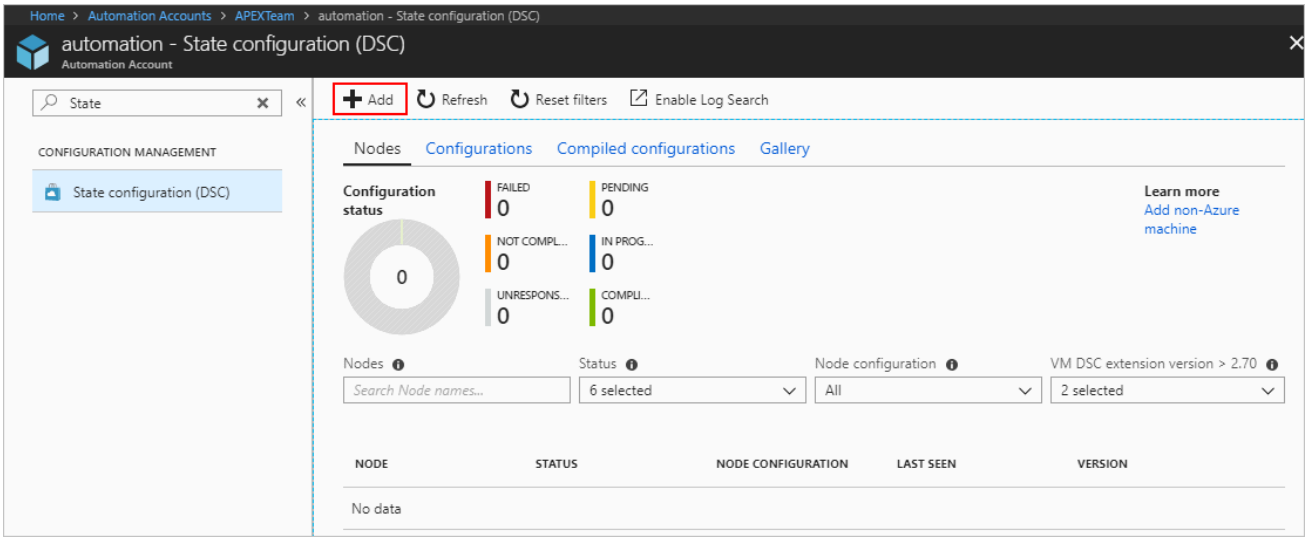
Puede usar State Configuration para administrar máquinas virtuales de Azure (tanto clásicas como Resource Manager), máquinas virtuales locales, máquinas Linux, máquinas virtuales de AWS y máquinas físicas locales. En este artículo, aprenderá a habilitar únicamente máquinas virtuales de Azure Resource Manager. Para obtener información sobre cómo habilitar otros tipos de máquinas, consulte Habilitar máquinas para la gestión por Azure Automation State Configuration.

1. Inicie sesión en el portal Azure.

2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.

3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.

4. En la página State Configuration (DSC), seleccione la pestaña **Nodos** y seleccione **Agregar**.



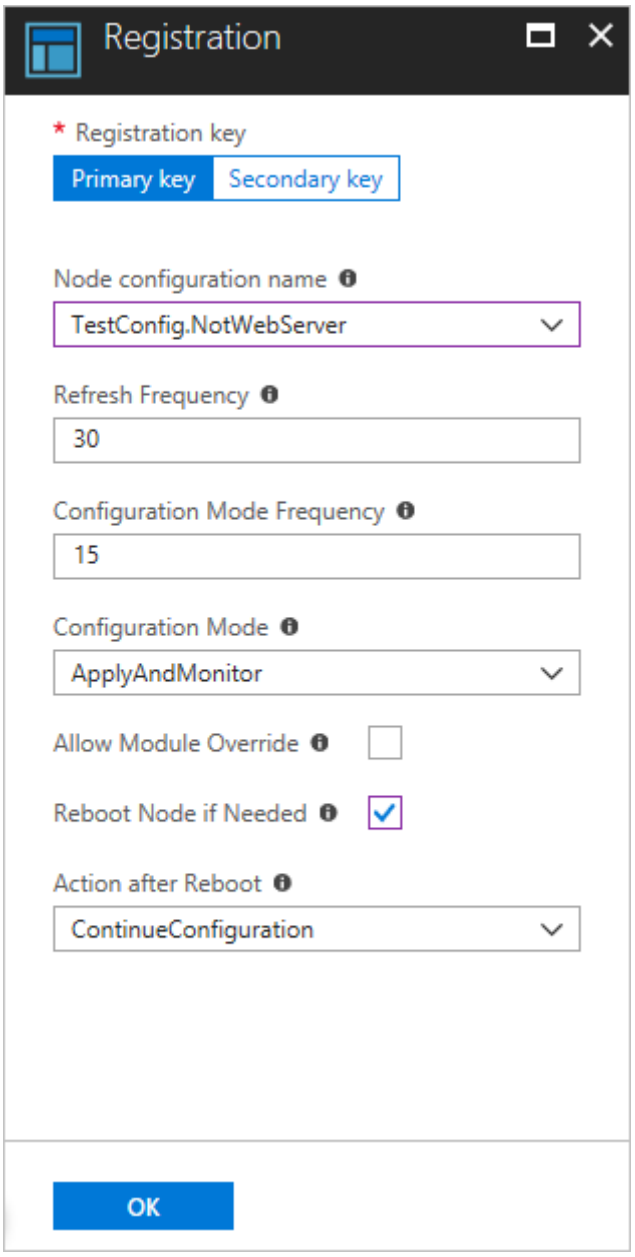
5. En el panel Virtual Machines, seleccione la máquina virtual.

6. En la página de detalles de la máquina virtual, seleccione **Conectar**.

> **Importante**
>
> La máquina virtual debe ser una máquina virtual Azure Resource Manager que ejecute un sistema operativo compatible.

7. En la página Registro, seleccione el nombre de la configuración del nodo que quiere aplicar a la máquina virtual en el campo **Nombre de la configuración del nodo**. Especificar un nombre en este momento es opcional. Puede cambiar la configuración de nodo asignada después de habilitar el nodo.

8. Marque **Reboot Node if Needed** (Reiniciar el nodo si es necesario) y seleccione **Aceptar**.



La configuración de nodo que especificó se aplica a la máquina virtual a intervalos especificados por el valor proporcionado para **Frecuencia del modo de configuración**. La máquina virtual comprueba si hay actualizaciones de la configuración de nodo en los intervalos especificados por el valor **Frecuencia de actualización**. Para obtener más información sobre cómo se usan estos valores, vea Configuring the Local Administrador de configuración.

Azure inicia el proceso de habilitación de la máquina virtual. Cuando finalice, la máquina virtual se muestra en la pestaña **Nodos** de la página State configuration (DSC) de la cuenta de Automation.

Vista de la lista de nodos administrados

La pestaña **Nodos** de la página State Configuration (DSC) contiene una lista de todas las máquinas habilitadas para la administración en la cuenta de Automation.

1. Inicie sesión en el portal Azure.
2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.
3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.
4. En la página State configuration (DSC), seleccione la pestaña **Nodos**.

Valores de estado de nodos de DSC

El nodo de DSC puede adoptar cualquiera de los seis valores siguientes como se indica a continuación:

- **Error**: este estado aparece cuando se produce un error al aplicar una o varias configuraciones en un nodo.
- **No compatible**: este estado aparece cuando se produce un desfase en un nodo y requiere una revisión a fondo.
- **No responde**: este estado aparece cuando no se ha sincronizado un nodo durante más de 24 horas.
- **Pendiente**: este estado se muestra cuando un nodo tiene una nueva configuración para aplicar y el servidor de extracción está esperando a que el nodo se active.
- **En curso**: este estado aparece cuando un nodo aplica la configuración y el servidor de extracción está esperando el estado.
- **Compatible**: este estado aparece cuando un nodo tiene una configuración válida y no se produce ningún desfase actualmente.

> **Nota:**
>
> - **RefreshFrequencyMins**: define la frecuencia de contacto del nodo con el servicio del agente y se puede proporcionar como parte de la incorporación a DSC. Tarda un valor máximo de 10 080 minutos.
> - El nodo se marcará como **No responde** si el nodo no se comunica con el servicio del agente durante 1440 minutos (1 día). Se recomienda usar un valor de **RefreshFrequencyMins** < 1440 minutos; de lo contrario, el nodo aparecerá con un estado **No responde falso**.

Ver informes de nodos administrados

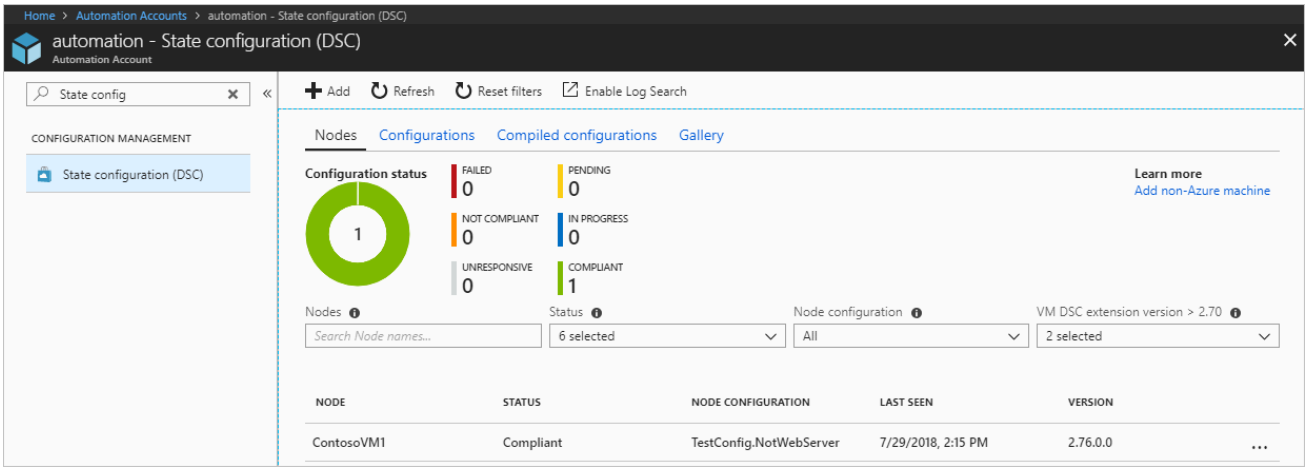
Cada vez que State Configuration realiza una comprobación de coherencia en un nodo administrado, el nodo devuelve un informe de estado al servidor de extracción. Estos informes se pueden ver en la página de dicho nodo.

1. Inicie sesión en el portal Azure.

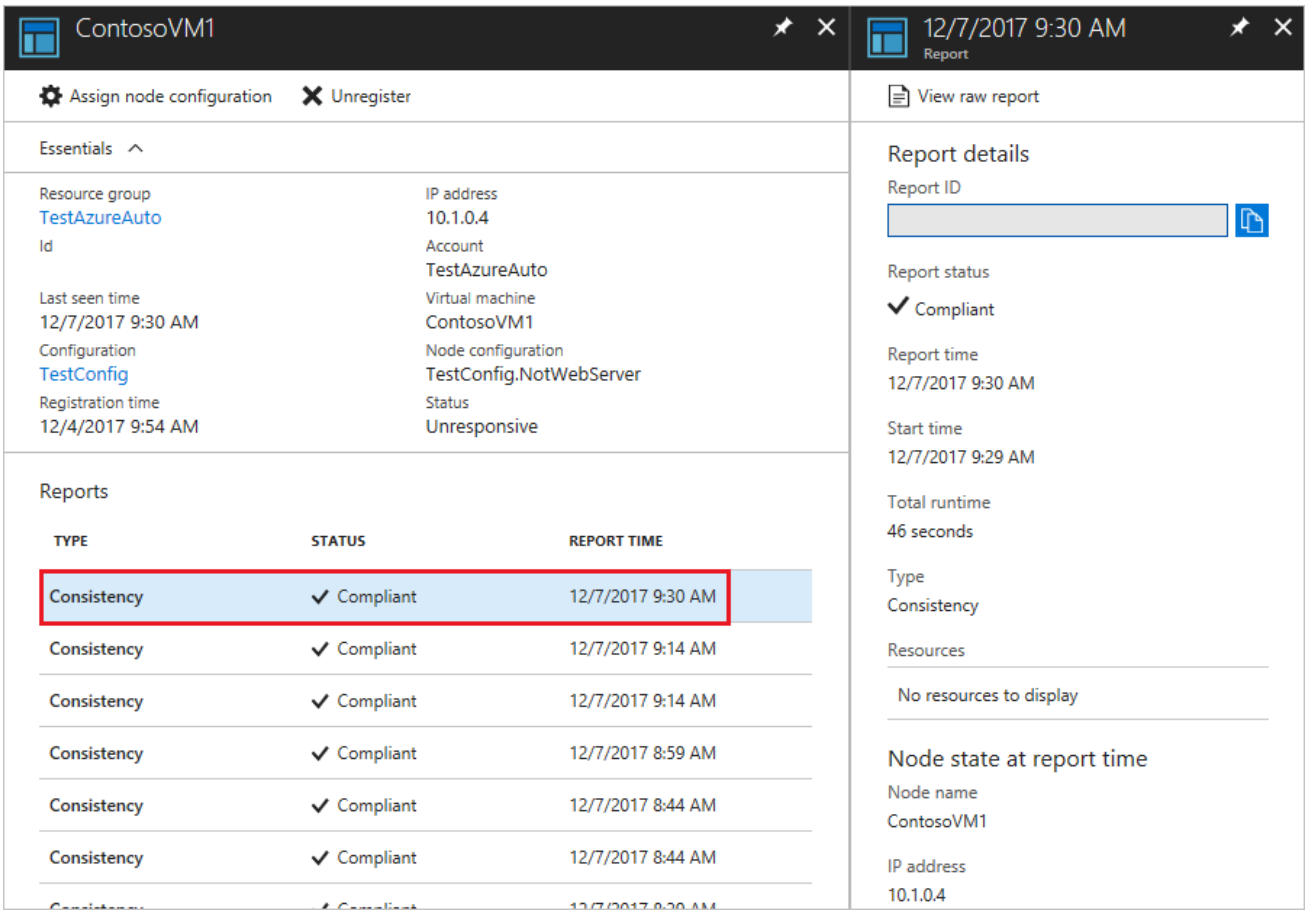
2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.

3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.

4. En la página State configuration (DSC), seleccione la pestaña **Nodos**. Aquí, puede ver la información general del estado de configuración y los detalles de cada nodo.



5. En la pestaña **Nodos**, seleccione el registro del nodo para abrir los informes. Selecciona el informe que quieres visualizar.



Puede ver la siguiente información de estado para la comprobación de coherencia correspondiente:

- Estado del informe. Los valores posibles son:
  - `Compliant`: el nodo es conforme con la comprobación.
  - `Failed`: la configuración no superó la comprobación.
  - `Not Compliant`: el nodo está en modo `ApplyandMonitor` y el equipo no está en el estado deseado.
- La hora de inicio de la comprobación de coherencia.
- El tiempo de ejecución total de la comprobación de coherencia.
- El tipo de comprobación de coherencia.
- Todos los errores, incluidos el código de error y el mensaje de error.
- Los recursos de DSC utilizados en la configuración y el estado de cada recurso (si el nodo está en el estado deseado para dicho recurso). Puede seleccionar cada recurso para obtener información más detallada sobre ese recurso.
- El nombre, la dirección IP y el modo de configuración del nodo.

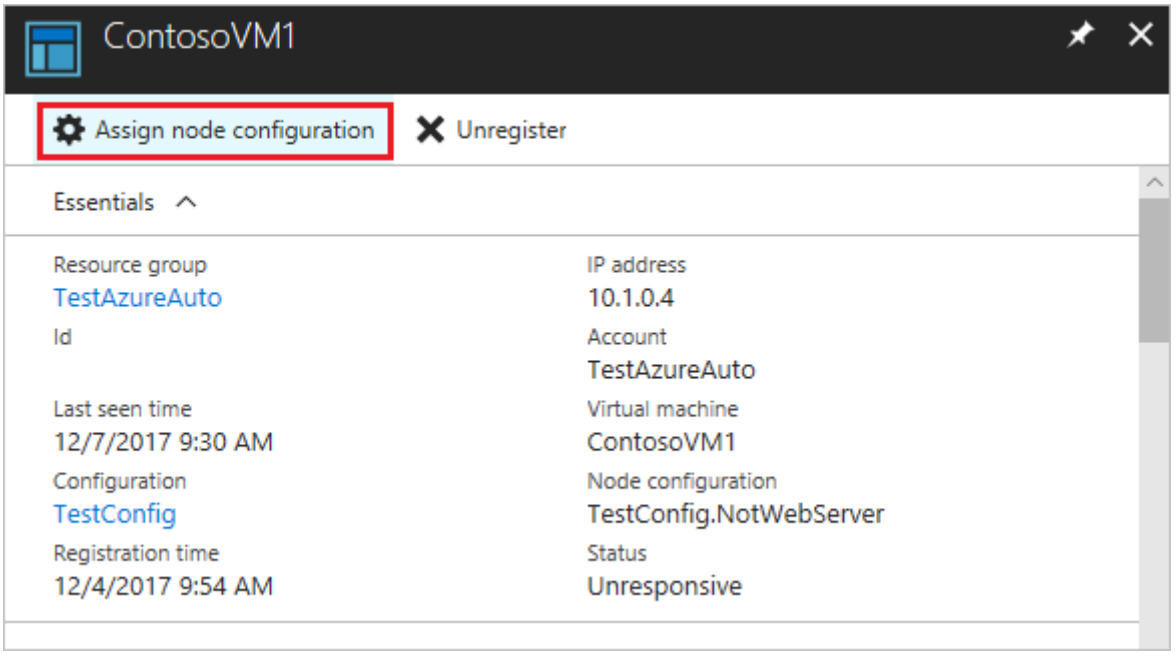
También puede seleccionar **Ver informe sin formato** para ver los datos reales que el nodo envía al servidor. Para más información sobre el uso de esos datos, consulte Uso de un servidor de informes de DSC.

Puede tardar algún tiempo antes de que el primer informe esté disponible, después de habilitar o reiniciar un nodo. Es posible que tenga que esperar hasta 30 minutos después de habilitar un nodo o reiniciar un nodo.

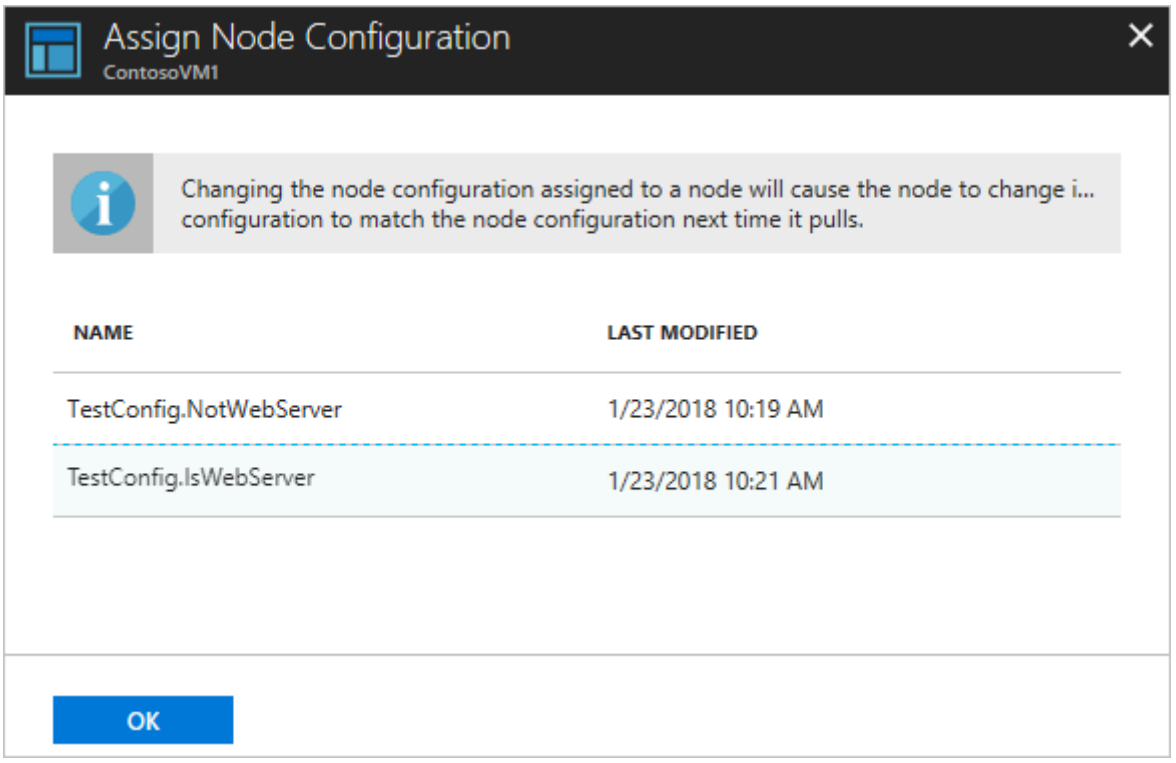
Reasignación de un nodo a una configuración de nodo diferente

Puede asignar un nodo para que use una configuración de nodo diferente a la que le asignó inicialmente.

1. Inicie sesión en el portal Azure.
2. A la izquierda, seleccione **Todos los recursos** y luego el nombre de su cuenta de Automation.
3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.
4. En la página State configuration (DSC), seleccione la pestaña **Nodos**.
5. En la página **Nodos**, seleccione el nombre del nodo que quiere reasignar.
6. En la página de dicho nodo, seleccione **Asignar configuración de nodo**.



7. En la página Asignar configuración de nodo, seleccione la configuración de nodo a la que desea asignar el nodo y luego seleccione **Aceptar**.



Anular el registro de un nodo

Puede anular el registro de un no si ya no desea que State Configuration administrarlo. Consulte Cómo quitar una configuración y un nodo de Automation State Configuration.

Pasos siguientes

- Para obtener información general, consulte Información general de Azure Automation State Configuration.
- Para habilitar la característica para las máquinas virtuales del entorno, consulte Enable Azure Automation State Configuration.
- Para comprender PowerShell DSC, consulte Descripción general de Windows PowerShell Desired State Configuration.
- Para obtener información sobre los precios, consulte precios de Azure Automation State Configuration.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.
- Para solucionar problemas relacionados con Azure Automation State Configuration (DSC), consulte Troubleshoot Azure Automation State Configuration issues.

Last updated on 20/04/2026

Habilitar Azure Automation State Configuration

> **Nota:**
>
> Azure Automation State Configuration se retirará el 30 de septiembre de 2027, realice la transición a Azure Machine Configuration por esa fecha. Para obtener más información, consulte el anuncio de la entrada de blog. El servicio Azure Machine Configuration combina características de la extensión de DSC, Azure Automation State Configuration y las características más solicitadas de los comentarios de los clientes. Azure Machine Configuration también incluye compatibilidad con máquinas híbridas a través de servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> Azure Automation DSC para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

En este artículo se describe cómo configurar las máquinas para la administración con Azure Automation State Configuration. Para obtener más información sobre este servicio, consulte la visión general de Azure Automation State Configuration.

Habilitación de máquinas virtuales de Azure

Azure Automation State Configuration permite habilitar fácilmente Azure máquinas virtuales para la administración de configuraciones, mediante el portal de Azure, las plantillas de Azure Resource Manager o PowerShell. La extensión Azure VM Desired State Configuration registra automáticamente la VM con Azure Automation State Configuration. Dado que la extensión de

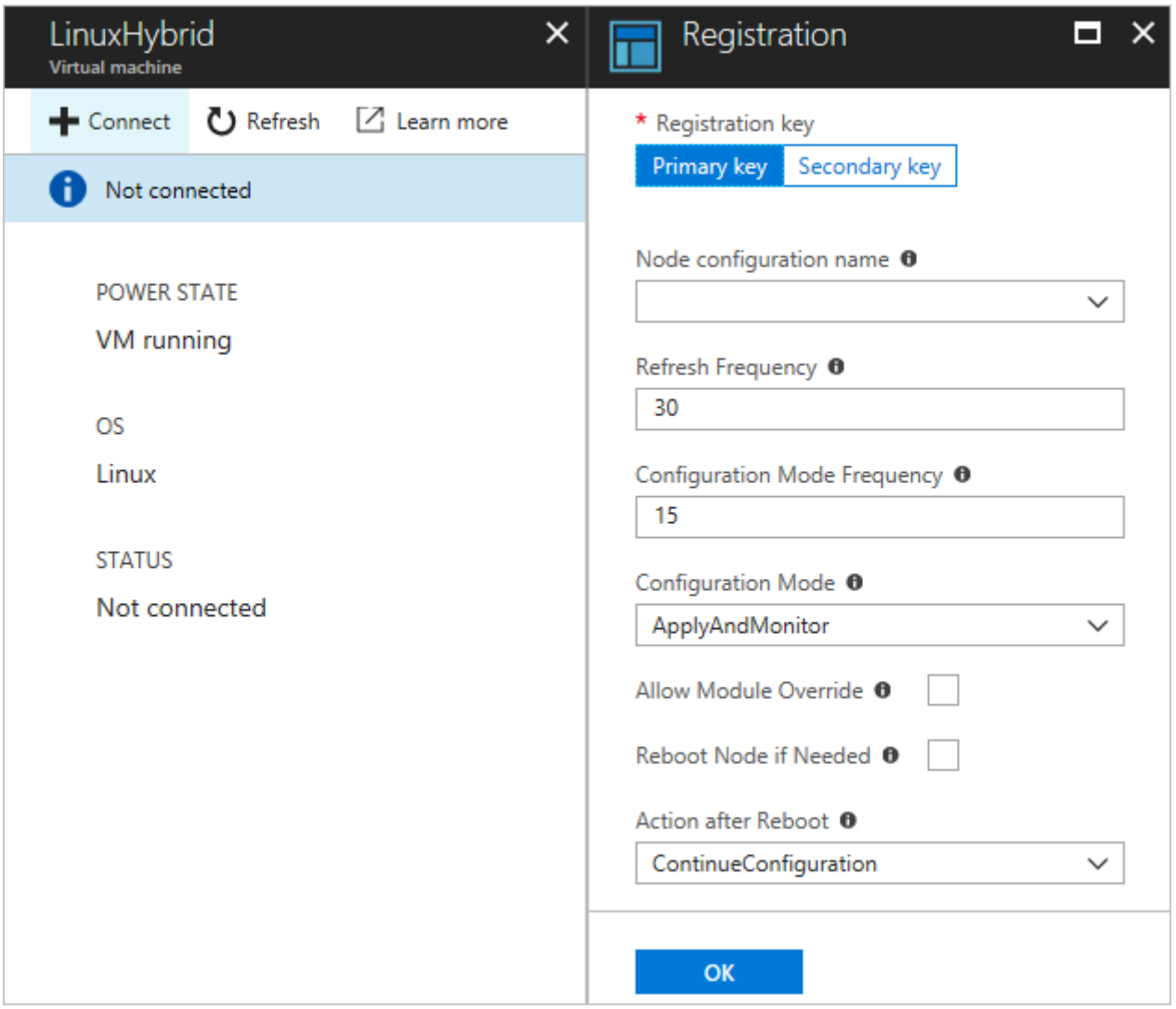
Azure se ejecuta de forma asincrónica, los pasos para realizar un seguimiento de su progreso se proporcionan en Estado de comprobación de la configuración de la máquina virtual.

> **Nota:**
>
> En la implementación de DSC en un nodo de Linux se usa la carpeta **/tmp**. Ciertos módulos, como `nxautomation`, se descargan temporalmente para su comprobación antes de instalarlos en las ubicaciones adecuadas correspondientes. Para asegurarse de que los módulos se instalan correctamente, el agente de Log Analytics para Linux necesita permisos de lectura y escritura en la carpeta **/tmp**. El agente de Log Analytics para Linux se ejecuta como usuario `omsagent`. Para conceder permiso de >escritura al usuario `omsagent`, ejecute el comando `setfacl -m u:omsagent:rwx /tmp`.

Habilitación de una máquina virtual mediante Azure portal

Para habilitar una máquina virtual de Azure para State Configuration a través del portal Azure:

1. Vaya a la cuenta de Azure Automation en la que se habilitarán las máquinas virtuales.
2. En la página State Configuration, seleccione la pestaña **Nodes**, seleccione **Agregar**.
3. Elija la máquina virtual que quiera habilitar.
4. Si la máquina no tiene instalada la extensión de estado deseado de PowerShell y el estado de funcionamiento es "activo", seleccione **Conectar**.
5. En **Registration**, escriba los valores PowerShell DSC Local Administrador de configuración necesarios para su caso de uso. Opcionalmente, se puede especificar una configuración de nodo para asignarla a la máquina virtual.



Habilitación de una máquina virtual mediante plantillas de Azure Resource Manager

Puede instalar y habilitar una máquina virtual para State Configuration mediante plantillas de Azure Resource Manager. Consulte Server administrado por el servicio de Desired State Configuration para obtener una plantilla de ejemplo que habilita una máquina virtual existente para la Configuración de Estado. Si va a administrar un conjunto de escalado de máquinas virtuales, consulte la plantilla de ejemplo en Configuración del conjunto de escalado de máquinas virtuales administrada por Azure Automation.

Habilitación de máquinas con PowerShell

Puede usar el cmdlet Register-AzAutomationDscNode en PowerShell para habilitar máquinas virtuales para State Configuration.

> **Nota:**

El cmdlet `Register-AzAutomationDscNode` solo se implementa actualmente para las máquinas que ejecutan Windows, ya que desencadena solo la extensión Windows.

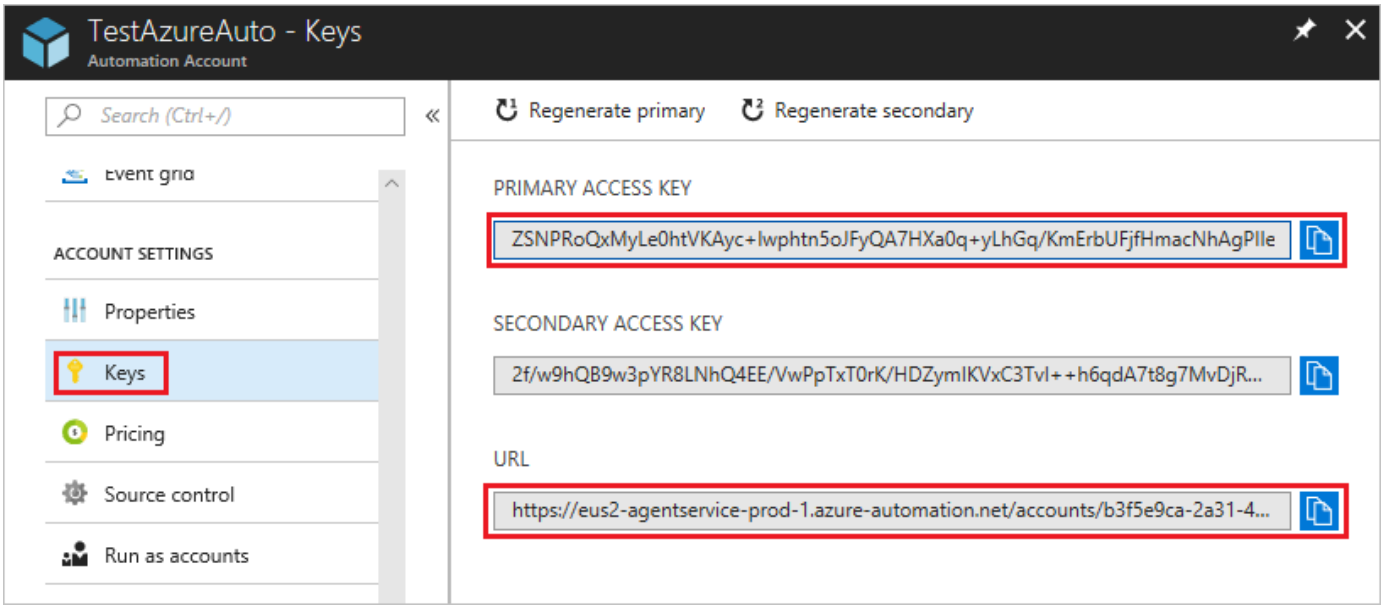
Registro de máquinas virtuales en suscripciones de Azure

La mejor manera de registrar máquinas virtuales de otras suscripciones de Azure es usar la extensión DSC en una plantilla de implementación de Azure Resource Manager. Los ejemplos se proporcionan en Extensión Desired State Configuration con plantillas de Azure Resource Manager.

Uso de la metaconfiguración de DSC para registrar máquinas híbridas

Puede habilitar máquinas de forma segura para una cuenta de Azure Automation a través de la metaconfiguración de DSC. Los protocolos implementados en DSC usan información de la metaconfiguración para autenticarse en Azure Automation State Configuration. El nodo se registra en el servicio en una dirección URL de registro y se autentica usando una clave de registro. Durante el registro, el nodo de DSC y el servicio de DSC negocian un certificado único que este nodo usará para autenticarse en el servidor después del registro. Este proceso impide que los nodos habilitados se suplanten entre sí, por ejemplo, en caso de que un nodo se vea comprometido y se comporte de forma malintencionada. Después del registro, la clave de registro no se vuelve a usar en operaciones de autenticación y se elimina del nodo.

Puede obtener la información necesaria para el protocolo de registro de State Configuration desde **Keys** en **Account Settings** en el portal de Azure.



- La URL de registro es el campo URL en la página Llaves.
- La clave de registro es el valor del campo **Clave de acceso primaria** o del campo **Clave de acceso secundaria** de la página Claves. Puede usarse cualquiera de las teclas.

Para una mayor seguridad, las claves de acceso primaria y secundaria de una cuenta de Automation se pueden volver a generar en cualquier momento en la página Claves. Volver a generar las claves evita que se produzcan futuros registros de nodo en los que se usen las claves anteriores.

Generación de metaconfiguraciones de DSC

Para habilitar cualquier máquina para State Configuration, puede generar una metaconfiguración DSC. Esta configuración indica al agente de DSC que extraiga y/o informe a State Configuration de Azure Automation. Puede generar una metaconfiguración de DSC para Azure Automation State Configuration mediante una configuración de DSC de PowerShell o los cmdlets de Azure Automation PowerShell.

> ⓘ **Nota:**
>
> Las metaconfiguraciones de DSC contienen los secretos necesarios para habilitar una máquina en una cuenta de Automation para su administración. Asegúrese de proteger adecuadamente cualquier metaconfiguración de DSC que cree o elimínelos tras su uso.

El Local Administrador de configuración (LCM) controla la compatibilidad de proxy para metaconfiguraciones. LCM se ejecuta en todos los nodos de destino, y es el responsable de llamar a los recursos de configuración que se incluyen en un script de metaconfiguración de DSC. Para incluir compatibilidad de proxy en una metaconfiguración, incluya definiciones de `ProxyURL` y propiedades `ProxyCredential` según sea necesario en los bloques `ConfigurationRepositoryWeb`, `ResourceRepositoryWeb` y `ReportServerWeb`. Un ejemplo de la configuración de la dirección URL es `ProxyURL = "http://172.16.3.6:3128";`. La propiedad `ProxyCredential` se establece en un objeto `PSCredential`, tal como se describe en Administrar credenciales en Azure Automation.

Generar metaconfiguraciones de DSC utilizando una configuración de DSC

1. Abra un editor de texto, como Visual Studio Code (VS Code), como administrador en un equipo del entorno local. La máquina debe tener instalada la versión más reciente de WMF

5 .

2. Copie el siguiente script localmente. Este script contiene una configuración de DSC de PowerShell para crear metaconfiguraciones y un comando para iniciar la creación de la metaconfiguración.

> **Nota:**
>
> Los nombres de configuración del nodo de State Configuration distinguen mayúsculas de minúsculas en Azure Portal. Si el caso no coincide, el nodo no se mostrará en la pestaña Nodos. Este proceso es para máquinas virtuales que no son de Azure. El registro de una máquina virtual de Azure como nodo híbrido provocará cargos de DSC.

PowerShell

```
# The DSC configuration that will generate metaconfigurations
[DscLocalConfigurationManager()]
Configuration DscMetaConfigs
{
     param
     (
         [Parameter(Mandatory=$True)]
         [String]$RegistrationUrl,

         [Parameter(Mandatory=$True)]
         [String]$RegistrationKey,

         [Parameter(Mandatory=$True)]
         [String[]]$ComputerName,

         [Int]$RefreshFrequencyMins = 30,

         [Int]$ConfigurationModeFrequencyMins = 15,

         [String]$ConfigurationMode = 'ApplyAndMonitor',

         [String]$NodeConfigurationName,

         [Boolean]$RebootNodeIfNeeded= $False,

         [String]$ActionAfterReboot = 'ContinueConfiguration',

         [Boolean]$AllowModuleOverwrite = $False,

         [Boolean]$ReportOnly
     )

     if(!$NodeConfigurationName -or $NodeConfigurationName -eq '')
     {
```

```
        $ConfigurationNames = $null
    }
    else
    {
        $ConfigurationNames = @($NodeConfigurationName)
    }

    if($ReportOnly)
    {
        $RefreshMode = 'PUSH'
    }
    else
    {
        $RefreshMode = 'PULL'
    }

    Node $ComputerName
    {
        Settings
        {
            RefreshFrequencyMins           = $RefreshFrequencyMins
            RefreshMode                    = $RefreshMode
            ConfigurationMode              = $ConfigurationMode
            AllowModuleOverwrite           = $AllowModuleOverwrite
            RebootNodeIfNeeded             = $RebootNodeIfNeeded
            ActionAfterReboot              = $ActionAfterReboot
            ConfigurationModeFrequencyMins = $ConfigurationModeFrequencyMins
        }

        if(!$ReportOnly)
        {
        ConfigurationRepositoryWeb AzureAutomationStateConfiguration
            {
                ServerUrl          = $RegistrationUrl
                RegistrationKey    = $RegistrationKey
                ConfigurationNames = $ConfigurationNames
            }

            ResourceRepositoryWeb AzureAutomationStateConfiguration
            {
                ServerUrl       = $RegistrationUrl
                RegistrationKey = $RegistrationKey
            }
        }

        ReportServerWeb AzureAutomationStateConfiguration
        {
            ServerUrl       = $RegistrationUrl
            RegistrationKey = $RegistrationKey
        }
    }
}

# Create the metaconfigurations
# NOTE: DSC Node Configuration names are case sensitive in the portal.
```

```
# TODO: edit the below as needed for your use case
$Params = @{
     RegistrationUrl = '<fill me in>';
     RegistrationKey = '<fill me in>';
     ComputerName = @('<some VM to onboard>', '<some other VM to onboard>');
     NodeConfigurationName = 'SimpleConfig.webserver';
     RefreshFrequencyMins = 30;
     ConfigurationModeFrequencyMins = 15;
     RebootNodeIfNeeded = $False;
     AllowModuleOverwrite = $False;
     ConfigurationMode = 'ApplyAndMonitor';
     ActionAfterReboot = 'ContinueConfiguration';
     ReportOnly = $False;  # Set to $True to have machines only report to AA DSC
but not pull from it
}

# Use PowerShell splatting to pass parameters to the DSC configuration being
invoked
# For more info about splatting, run: Get-Help -Name about_Splatting
DscMetaConfigs @Params
```

3. Rellene la clave del Registro y la dirección URL para su cuenta de Automation y los nombres de las máquinas que quiere habilitar. Todos los demás parámetros son opcionales. Para buscar la clave de registro y la dirección URL de registro de la cuenta de Automation, consulte Uso de la metaconfiguración de DSC para registrar máquinas híbridas.

4. Si desea que las máquinas notifiquen la información de estado de DSC a Azure Automation State Configuration, pero no deseen obtener la configuración o los módulos de PowerShell, establezca el parámetro `ReportOnly` en true.

5. Si `ReportOnly` no está configurado, las máquinas informan a Azure Automation State Configuration sobre el estado de DSC y extraen las configuraciones o los módulos de PowerShell. Establezca los parámetros según corresponda en los bloques `ConfigurationRepositoryWeb`, `ResourceRepositoryWeb` y `ReportServerWeb`.

6. Ejecute el script. Ahora, debería tener una carpeta directorio llamada **DscMetaConfigs** en funcionamiento en el directorio de trabajo que contenga las metaconfiguraciones de DSC de PowerShell de las máquinas que quiera habilitar (como administrador).

   PowerShell

   ```
   Set-DscLocalConfigurationManager -Path ./DscMetaConfigs
   ```

Generación de metaconfiguraciones de DSC mediante cmdlets de Azure Automation

Puede generar las metaconfiguraciones de DSC mediante los cmdlets de Azure Automation en las condiciones siguientes:

- Los valores predeterminados de LCM coinciden con tu caso de uso
- Quiere habilitar las máquinas para extraer y notificar a State Configuration de Azure Automation

Siga estos pasos para generar las metaconfiguraciones:

1. Abra la consola de PowerShell o VS Code como administrador en una máquina de su entorno local.

2. Conéctese a Azure Resource Manager mediante Connect-AzAccount.

3. Descargue las metaconfiguraciones de DSC de PowerShell de las máquinas que quiera habilitar desde la cuenta de Automation en la que esté configurando los nodos.

   PowerShell

   ```
   # Define the parameters for Get-AzAutomationDscOnboardingMetaconfig using
   PowerShell Splatting
   $Params = @{
       ResourceGroupName = 'ContosoResources' # The Resource Group that contains
   your Azure Automation account
       AutomationAccountName = 'ContosoAutomation'; # The Azure Automation account
   where you want to onboard the node
       ComputerName = @('web01', 'web02', 'sql01'); # The computers to generate the
   metaconfigurations for
       OutputFolder = "$env:UserProfile\Desktop\";
   }
   # Use PowerShell splatting to pass parameters to the Azure Automation cmdlet
   being invoked
   # For more info about splatting, run: Get-Help -Name about_Splatting
   Get-AzAutomationDscOnboardingMetaconfig @Params
   ```

4. Ahora, debería tener la carpeta **DscMetaConfigs** que contiene las metaconfiguraciones de DSC de PowerShell de las máquinas que quiera habilitar (como administrador).

   PowerShell

   ```
   Set-DscLocalConfigurationManager -Path $env:UserProfile\Desktop\DscMetaConfigs
   ```

Habilitación de máquinas de Windows físicas o virtuales

Puede habilitar servidores Windows que se ejecutan de forma local o en otros entornos de la nube (incluidas las instancias de AWS EC2) para Azure Automation State Configuration. Los servidores deben tener acceso outbound a Azure.

1. Asegúrese de que la versión más reciente de WMF 5 esté instalada en las máquinas para habilitar para State Configuration. Además, WMF 5 debe estar instalado en el equipo que esté usando para habilitar las máquinas.

2. Siga las instrucciones de Generación de metaconfiguraciones de DSC para crear una carpeta que contenga las metaconfiguraciones de DSC necesarias.

3. Use el siguiente cmdlet para aplicar las metaconfiguraciones de DSC de PowerShell de forma remota a las máquinas que se van a habilitar.

   PowerShell

   ```
   Set-DscLocalConfigurationManager -Path C:\Users\joe\Desktop\DscMetaConfigs -
   ComputerName MyServer1, MyServer2
   ```

4. Si no puede aplicar las metaconfiguraciones de DSC de PowerShell de forma remota, copie la carpeta de **metaconfiguraciones** en las máquinas que va a habilitar. A continuación, agregue código para llamar a Set-DscLocalConfigurationManager localmente en las máquinas.

5. Con el portal o los cmdlets de Azure, compruebe que las máquinas aparecen como nodos State Configuration registrados en la cuenta de Azure Automation.

Habilitación de máquinas físicas o virtuales con Linux

Puede habilitar servidores Linux que se ejecutan de forma local o en otros entornos en la nube para State Configuration. Los servidores deben tener acceso outbound a Azure.

1. Asegúrese de que la versión más reciente de PowerShell Desired State Configuration para Linux esté instalada en las máquinas para habilitar para State Configuration.

2. Si los Valores predeterminados del Administrador de configuración local de DSC de PowerShell coinciden con su caso de uso, y quiere habilitar máquinas de modo que extraigan información de State Configuration y también envíen informes allí:

   - En cada máquina Linux que quiera habilitar, use `Register.py` para activar la máquina con los valores predeterminados del Administrador de Configuración Local de PowerShell DSC.

`/opt/microsoft/dsc/Scripts/Register.py <Automation account registration key> <Automation account registration URL>`

- Para buscar la clave de registro y la dirección URL de registro de la cuenta de Automation, consulte Uso de la metaconfiguración de DSC para registrar máquinas híbridas.

3. Si los valores predeterminados de Administrador de configuración local (LCM) de DSC de PowerShell no coinciden con el caso de uso o desea habilitar las máquinas que solo informan a Azure Automation State Configuration, siga los pasos 4 a 7. De lo contrario, vaya directamente al paso 7.

4. Siga las instrucciones de la sección Generación de metaconfiguraciones de DSC para generar una carpeta que contenga las metaconfiguraciones de DSC necesarias.

5. Asegúrese de que la versión más reciente de WMF 5 está instalada en el equipo que se usa para habilitar las máquinas para State Configuration.

6. Agregue el siguiente código para aplicar las metaconfiguraciones de DSC de PowerShell de forma remota a las máquinas que quiera habilitar.

PowerShell

```
$SecurePass = ConvertTo-SecureString -String '<root password>' -AsPlainText -
Force
$Cred = New-Object System.Management.Automation.PSCredential 'root', $SecurePass
$Opt = New-CimSessionOption -UseSsl -SkipCACheck -SkipCNCheck -
SkipRevocationCheck
# need a CimSession for each Linux machine to onboard
$Session = New-CimSession -Credential $Cred -ComputerName <your Linux machine> -
Port 5986 -Authentication basic -SessionOption $Opt
Set-DscLocalConfigurationManager -CimSession $Session -Path
C:\Users\joe\Desktop\DscMetaConfigs
```

7. Si no puede aplicar las metaconfiguraciones de DSC de PowerShell de forma remota, copie las metaconfiguraciones correspondientes a las máquinas remotas desde la carpeta descrita en el paso 4 hacia las máquinas Linux.

8. Agregue código para llamar a `Set-DscLocalConfigurationManager.py` localmente en cada máquina Linux que vaya a habilitar State Configuration.

`/opt/microsoft/dsc/Scripts/SetDscLocalConfigurationManager.py -configurationmof <path to metaconfiguration file>`

9. Asegúrese de que las máquinas se muestran como nodos DSC registrados en la cuenta de Azure Automation. Puede usar el portal de Azure o los cmdlets.

Nuevo registro de un nodo

Después de registrar una máquina como un nodo DSC en Azure Automation State Configuration, hay varias razones por las que es posible que tenga que volver a registrar ese nodo en el futuro.

- **Renovación de certificados.** Para las versiones de Windows Server antes de Windows Server 2019, cada nodo negocia automáticamente un certificado único para la autenticación que expira después de un año. Si un certificado expira sin renovación, el nodo no puede comunicarse con Azure Automation y está marcado como `Unresponsive`. En la actualidad, el protocolo de registro de DSC de PowerShell no puede renovar automáticamente los certificados que vayan a expirar, por lo que tendrá que volver a registrar los nodos transcurrido un año. Antes de volver a registrarlos, asegúrese de que cada nodo ejecuta WMF 5 RTM.

  Se genera y utiliza un nuevo certificado si vuelve a registrar 90 días o menos antes de la expiración del certificado o en cualquier momento una vez cumplido el periodo de expiración. Este problema se ha corregido en Windows Server 2019 y versiones posteriores.

- **Cambios en los valores de LCM de DSC.** Puede que haya que cambiar los valores de LCM de DSC de PowerShell establecidos durante el registro inicial del nodo, por ejemplo, `ConfigurationMode`. Actualmente, estos valores de agente DSC solo se pueden cambiar realizando un nuevo registro. La única excepción es el valor de configuración del nodo. Puede cambiar este valor directamente en Azure Automation DSC.

Un nodo se puede volver a registrar del mismo modo en que se registró inicialmente, con cualquiera de los métodos descritos en este documento. No es necesario anular el registro de un nodo de Azure Automation State Configuration antes de volver a registrarlo.

Comprobación del estado de la instalación de máquina virtual

State Configuration permite habilitar fácilmente Azure Windows máquinas virtuales para la administración de la configuración. Internamente, la extensión Configuración de estado deseado de la máquina virtual de Azure se usa para registrar la máquina virtual con Azure Automation State Configuration. Dado que la extensión Desired State Configuration para máquinas virtuales

de Azure se ejecuta de forma asincrónica, el seguimiento de su progreso y la resolución de problemas durante su ejecución pueden ser importantes.

> **Nota:**
>
> Cualquier método para habilitar las máquinas virtuales de Windows en Azure para la Configuración de Estado que utilice la extensión de Desired State Configuration de la máquina virtual de Azure puede tardar hasta una hora para que Azure Automation muestre las máquinas virtuales como registradas. Este retraso se debe a la instalación de WMF 5 en la máquina virtual mediante la extensión Desired State Configuration de máquina virtual de Azure, que es necesaria para habilitar las máquinas virtuales para State Configuration.

Para ver el estado de la extensión Desired State Configuration de máquina virtual de Azure:

1. En el portal de Azure, vaya a la máquina virtual que se está habilitando.
2. Seleccione **Extensiones** en **Configuración**.
3. Ahora, seleccione **DSC** o **DSCForLinux**, según el sistema operativo.
4. Para obtener más detalles, puede seleccionar **Ver estado detallado**.

Pasos siguientes

- Para empezar, consulte Get started with Azure Automation State Configuration.
- Para obtener información sobre la compilación de configuraciones de DSC para poder asignarlas a nodos de destino, consulte Compilar configuraciones de DSC en Azure Automation State Configuration.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.
- Para obtener información sobre los precios, consulte precios de Azure Automation State Configuration.
- Para obtener un ejemplo de uso de Azure Automation State Configuration en una canalización de implementación continua, consulte Configurar la implementación continua con Chocolatey.
- Para solucionar problemas relacionados con la habilitación de Azure Automation State Configuration, consulte Solución de problemas de Configuración de Estado de Azure Automation.

Last updated on 29/04/2026

Configura las máquinas a un estado deseado

> **Nota:**
>
> Azure Automation State Configuration se retirará el 30 de septiembre de 2027, realice la transición a Azure Machine Configuration por esa fecha. Para obtener más información, consulte el anuncio del blog. El servicio Azure Machine Configuration combina características de la extensión de DSC, Azure Automation State Configuration y las características más solicitadas de los comentarios de los clientes. Azure Machine Configuration también incluye compatibilidad con máquinas híbridas a través de servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> Azure Automation DSC para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

Azure Automation State Configuration permite especificar configuraciones para los servidores y asegurarse de que esos servidores están en el estado especificado a lo largo del tiempo.

- ✓ Incorporación de una máquina virtual para ser administrada por Azure Automation DSC
- ✓ Carga de una configuración en Azure Automation
- ✓ Compilar una configuración en una configuración de nodo
- ✓ Asignación de una configuración de nodo a un nodo administrado
- ✓ Comprobación del estado de cumplimiento de un nodo administrado

Para este tutorial, se usa una configuración DSC simple que se asegura de que IIS está instalado en la máquina virtual.

Requisitos previos

> ⓘ **Nota:**
>
> Windows Server 2008 y Windows Server 2008 R2 han alcanzado la finalización del soporte (EOS). Para obtener más información, vea **Fin del soporte técnico para Windows Server 2008 y Windows Server 2008 R2** y **Realizar una actualización local a Windows Server 2016, 2019, 2022 o 2025**. Revise el uso y planee las actualizaciones y migraciones del sistema operativo en consecuencia.

- Una cuenta de Azure Automation. Para más información sobre las cuentas de Automation y sus requisitos, consulte Introducción a la autenticación de cuentas de Automation.
- Una máquina virtual de Azure Resource Manager (no clásica) que ejecuta Windows Server 2008 R2 o posterior. Para obtener instrucciones sobre cómo crear una máquina virtual, consulte Crear la primera máquina virtual Windows en el portal de Azure.
- Azure PowerShell módulo versión 3.6 o posterior. Ejecute `Get-Module -ListAvailable Az` para encontrar la versión. Si necesita actualizar, consulte Install Azure PowerShell module.
- Familiaridad con Desired State Configuration (DSC) (Configuración de Estado Deseado). Para obtener información sobre DSC, consulte Windows PowerShell Desired State Configuration Overview.

Compatibilidad con configuraciones parciales

Azure Automation State Configuration admite el uso de las configuraciones parciales. En este escenario, DSC está configurado para administrar varias configuraciones de forma independiente y cada configuración se recupera de Azure Automation. Sin embargo, solo se puede asignar una configuración a un nodo por cada cuenta de automatización. Esto significa que si usa dos configuraciones para un nodo, necesita dos cuentas de Automation.

Para obtener más información sobre cómo registrar una configuración parcial de un servicio de extracción, consulte la documentación sobre configuraciones parciales.

Para obtener más información acerca de cómo los equipos pueden trabajar juntos para administrar de forma colaborativa los servidores con la configuración como código, consulte Descripción del rol de DSC en una canalización de CI/CD.

Iniciar sesión en Azure

Inicie sesión en la suscripción de Azure con el cmdlet Connect-AzAccount y siga las instrucciones en pantalla.

PowerShell

```
Connect-AzAccount
```

Creación y carga de una configuración en Azure Automation

En un editor de texto, escriba lo siguiente y guárdelo localmente como **TestConfig.ps1**.

PowerShell

```
configuration TestConfig {
   Node WebServer {
      WindowsFeature IIS {
         Ensure               = 'Present'
         Name                 = 'Web-Server'
         IncludeAllSubFeature = $true
      }
   }
}
```

> ⓘ **Nota:**
>
> Los nombres de configuración de Azure Automation deben limitarse a no más de 100 caracteres.
>
> En escenarios más avanzados en los que se requiere la importación de varios módulos que proporcionen recursos de DSC, asegúrese de que cada módulo tenga una línea única `Import-DscResource` en la configuración.

Llame al cmdlet Import-AzAutomationDscConfiguration para cargar una configuración de nodo en la cuenta de Automation.

PowerShell

```
$importAzAutomationDscConfigurationSplat = @{
    SourcePath = 'C:\DscConfigs\TestConfig.ps1'
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
    Published = $true
```

```
}
Import-AzAutomationDscConfiguration @importAzAutomationDscConfigurationSplat
```

Compilar una configuración en una configuración de nodo

Una configuración DSC se debe compilar en una configuración de nodo antes de que se pueda asignar a un nodo. Consulte Configuraciones de DSC.

Ejecuta el cmdlet Start-AzAutomationDscCompilationJob para compilar la configuración `TestConfig` en una configuración de nodo denominada `TestConfig.WebServer` en tu cuenta de Automatización.

PowerShell

```
$startAzAutomationDscCompilationJobSplat = @{
    ConfigurationName = 'TestConfig'
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
}
Start-AzAutomationDscCompilationJob @startAzAutomationDscCompilationJobSplat
```

Registro de una máquina virtual que se va a administrar mediante State Configuration

Puede usar Azure Automation State Configuration para administrar máquinas virtuales de Azure (clásicas y Resource Manager), máquinas virtuales locales, máquinas Linux, máquinas virtuales de AWS y máquinas físicas locales. En este artículo, tratamos cómo registrar solo Azure Resource Manager máquinas virtuales. Para obtener información sobre cómo registrar otros tipos de máquinas, consulte Integración de máquinas para la gestión mediante la Configuración de Estado de Automatización de Azure.

Llame al cmdlet Register-AzAutomationDscNode para registrar la máquina virtual con Azure Automation State Configuration como nodo administrado.

PowerShell

```
$registerAzAutomationDscNodeSplat = @{
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
    AzureVMName = 'DscVm'
```

```
}
Register-AzAutomationDscNode @registerAzAutomationDscNodeSplat
```

Especificación de opciones del modo de configuración

Use el cmdlet Register-AzAutomationDscNode para registrar una máquina virtual como un nodo administrado y especificar las propiedades de configuración. Por ejemplo, puede especificar que el estado de la máquina se aplique solo una vez; para ello, especifique `ApplyOnly` como valor de la propiedad `ConfigurationMode`. State Configuration no intenta aplicar la configuración después de la comprobación inicial.

PowerShell

```
$registerAzAutomationDscNodeSplat = @{
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
    AzureVMName = 'DscVm'
    ConfigurationMode = 'ApplyOnly'
}
Register-AzAutomationDscNode @registerAzAutomationDscNodeSplat```

You can also specify how often DSC checks the configuration state by using the
`ConfigurationModeFrequencyMins` property. For more information about DSC
configuration settings,
see [Configuring the Local Configuration Manager][05].

```powershell
# Run a DSC check every 60 minutes
$registerAzAutomationDscNodeSplat = @{
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
    AzureVMName = 'DscVm'
    ConfigurationModeFrequencyMins = 60
}
Register-AzAutomationDscNode @registerAzAutomationDscNodeSplat```

## Assign a node configuration to a managed node

Now we can assign the compiled node configuration to the VM we want to configure.

```powershell
# Get the ID of the DSC node
$getAzAutomationDscNodeSplat = @{
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
    Name = 'DscVm'
}
$node = Get-AzAutomationDscNode @getAzAutomationDscNodeSplat

# Assign the node configuration to the DSC node
```

```
$setAzAutomationDscNodeSplat = @{
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
    NodeConfigurationName = 'TestConfig.WebServer'
    NodeId = $node.Id
}
Set-AzAutomationDscNode @setAzAutomationDscNodeSplat
```

Esto asigna la configuración de nodo llamada `TestConfig.WebServer` al nodo DSC registrado `DscVm`. De forma predeterminada, se comprueba el cumplimiento de la configuración del nodo DSC cada 30 minutos. Para obtener información sobre cómo cambiar el intervalo de comprobación de cumplimiento, vea Configuring the Local Configuration Manager.

Comprobación del estado de cumplimiento de un nodo administrado

Puede obtener informes sobre el estado de cumplimiento de un nodo administrado mediante el cmdlet Get-AzAutomationDscNodeReport.

PowerShell

```
# Get the ID of the DSC node
$getAzAutomationDscNodeSplat = @{
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
    Name = 'DscVm'
}
$node = Get-AzAutomationDscNode @getAzAutomationDscNodeSplat

# Get an array of status reports for the DSC node
$getAzAutomationDscNodeReportSplat = @{
    ResourceGroupName = 'MyResourceGroup'
    AutomationAccountName = 'myAutomationAccount'
    NodeId = $node.Id
}
$reports = Get-AzAutomationDscNodeReport @getAzAutomationDscNodeReportSplat

# Display the most recent report
$reports[0]
```

Pasos siguientes

- Para empezar, consulte Get started with Azure Automation State Configuration.
- Para obtener información sobre cómo habilitar nodos, consulte Enable Azure Automation State Configuration.

- Para obtener información sobre la compilación de configuraciones de DSC para poder asignarlas a nodos de destino, consulte Compilar configuraciones de DSC en Azure Automation State Configuration.
- Para ver un ejemplo del uso de Azure Automation State Configuration en una canalización de implementación continua, consulte Configurar implementación continua con Chocolatey.
- Para obtener información sobre los precios, consulte los precios de Azure Automation State Configuration.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.
- Para solucionar problemas relacionados con la configuración de máquinas en un estado deseado mediante Azure Automation State Configuration, consulte Solucionar problemas de Azure Automation State Configuration.

Last updated on 24/04/2026

Composición de configuraciones DSC

> **Nota:**
>
> State Configuration de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration para esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> DSC de Automatización de Azure para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

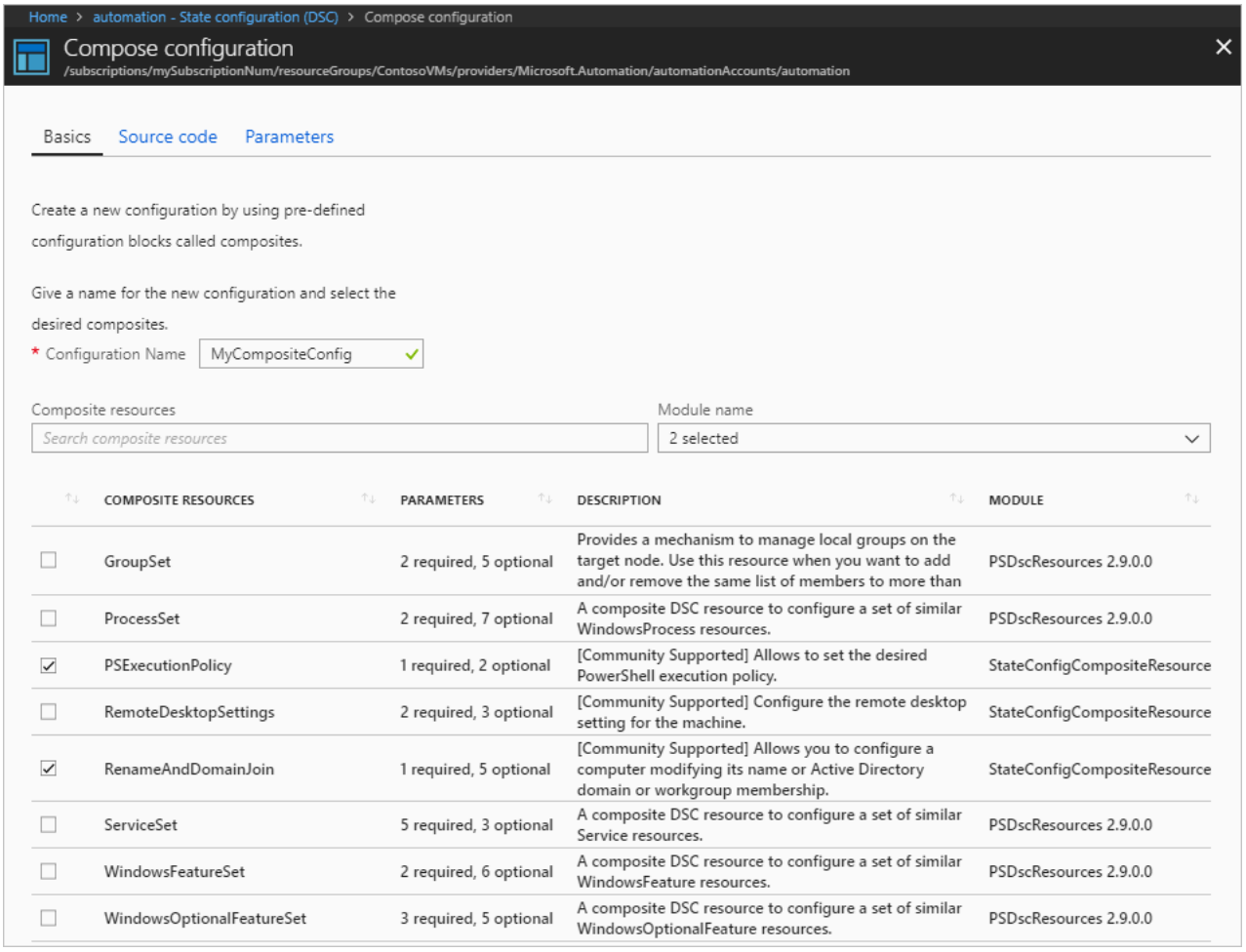
Cuando debe administrar un recurso con más de una configuración de estado deseado (DSC), la mejor opción es usar recursos compuestos. Un recurso compuesto es una configuración anidada y parametrizada que se usa como recurso de DSC dentro de otra configuración. Este uso le permite crear configuraciones complejas, así como administrar y compilar individualmente los recursos compuestos subyacentes.

Azure Automation permite la importación y compilación de recursos compuestos. Una vez que haya importado los recursos compuestos en su cuenta de Automation, podrá usar Azure Automation State Configuration a través de la característica **State Configuration (DSC)** en Azure Portal.

Composición de una configuración

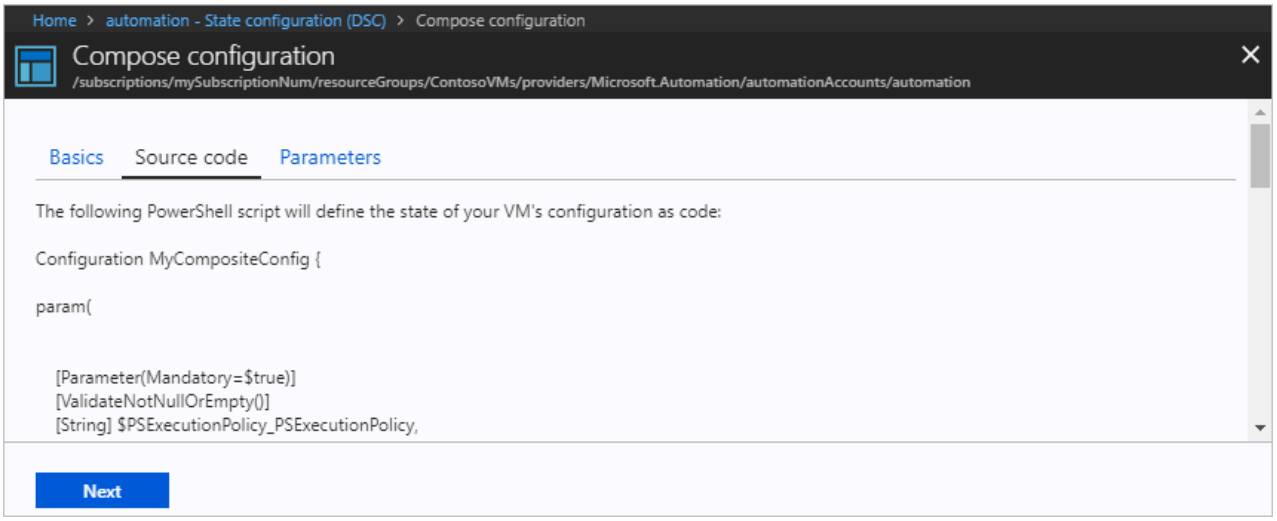
Para poder asignar una configuración realizada desde recursos compuestos en Azure Portal, debe componerla. La composición usa **Configuración de Compose** en la página State Configuration (DSC) mientras se encuentra en la pestaña **Configuraciones** o **Configuraciones compiladas**.

1. Inicie sesión en Azure Portal.
2. A la izquierda, seleccione **Todos los recursos** y, luego, en el nombre de la cuenta de Automation.
3. En la página Cuenta de Automation, seleccione **State configuration (DSC)** en **Administración de configuración**.
4. En la página State Configuration (DSC), seleccione la pestaña **Configuraciones** o **Configuraciones compiladas** y, después, seleccione **Configuración de Compose** en el menú de la parte superior de la página.
5. En el paso **Conceptos básicos**, proporcione el nuevo nombre de configuración (obligatorio) y seleccione cualquier parte en la fila de cada recurso compuesto que quiera incluir en la nueva configuración, luego, seleccione **Siguiente** o seleccione el paso **Código fuente**. En los pasos siguientes, hemos seleccionado `PSExecutionPolicy` y `RenameAndDomainJoin` recursos compuestos.

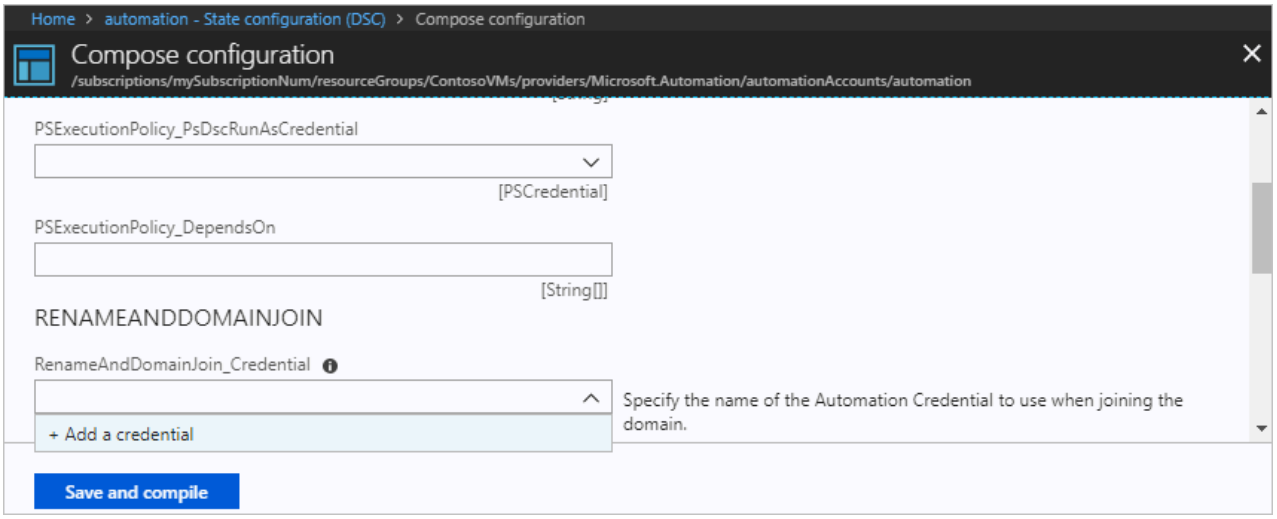


6. El paso **Código fuente** muestra el aspecto de la configuración compuesta de los recursos compuestos seleccionados. Puede ver la combinación de todos los parámetros y cómo se pasan al recurso compuesto. Cuando haya terminado de revisar el código fuente nuevo,

seleccione **Siguiente** o seleccione el paso **Parámetros**.



7. En el paso **Parámetros**, el parámetro de cada recurso compuesto se expone para que puedan proporcionarse valores. Si la descripción del parámetro se muestra junto al campo de parámetro. Si un parámetro es de un tipo `[PSCredential]`, el menú desplegable proporciona una lista de objetos **Credential** en la cuenta de Automation actual. También hay disponible una opción **+ Agregar una credencial**. Proporcione valores para los parámetros necesarios y, después, seleccione **Guardar y compilar**.



Envío de la configuración para su compilación

Envíe la configuración nueva para su compilación. Puede ver el estado del trabajo de compilación igual que con cualquier configuración importada. Para más información, consulte la sección sobre visualización de un trabajo de compilación.

Una vez que la compilación finalice correctamente, la nueva configuración aparece en la pestaña **Configuraciones compiladas**. A continuación, puede asignar la configuración a un nodo administrado, siguiendo los pasos descritos en Reasignación de un nodo a una configuración de nodo diferente.

Pasos siguientes

- Para aprender a habilitar nodos, consulte el artículo sobre habilitación de State Configuration de Azure Automation.
- Para aprender a compilar configuraciones de DSC para poder asignarlas a los nodos de destino, consulte Compilación de configuraciones de DSC en Azure Automation State Configuration.
- Para ver un ejemplo del uso de State Configuration de Azure Automation en una canalización de implementación continua, consulte Configuración de la implementación continua con Chocolatey.
- Para obtener información de precios, consulte Precios de State Configuration de Azure Automation.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.

Last updated on 17/11/2025

Compilación de configuraciones de DSC en Azure Automation State Configuration

> **Nota:**
>
> Azure Automation State Configuration se retirará el 30 de septiembre de 2027, realice la transición a Azure Machine Configuration por esa fecha. Para obtener más información, consulte el anuncio de la entrada de blog. El servicio Azure Machine Configuration combina características de la extensión de DSC, Azure Automation State Configuration y las características más solicitadas de los comentarios de los clientes. Azure Machine Configuration también incluye compatibilidad con máquinas híbridas a través de servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> Azure Automation DSC para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

Puede compilar configuraciones de Desired State Configuration (DSC) de Azure Automation State Configuration de las siguientes maneras:

- servicio de compilación de Azure State Configuration
  - Método para principiantes, con una interfaz de usuario interactiva
  - Seguimiento sencillo del estado del trabajo
- Windows PowerShell
  - Llamada desde Windows PowerShell en una estación de trabajo local o servicio de compilación
  - Integrar con el canal de pruebas de desarrollo
  - Proporcione valores de parámetros complejos

- Trabajo con datos de nodo y datos que no son de nodo a escala
- Mejora significativa del rendimiento

También puede usar plantillas de Azure Resource Manager con Azure Desired State Configuration extensión (DSC) para insertar configuraciones en las máquinas virtuales de Azure. La extensión DSC de Azure usa el marco del agente de máquina virtual de Azure para entregar, aplicar e informar sobre las configuraciones de DSC que se ejecutan en máquinas virtuales de Azure. Para obtener más información sobre la compilación mediante plantillas de Azure Resource Manager, consulte Desired State Configuration extensión con plantillas de Azure Resource Manager.

Compilación de una configuración de DSC en Azure State Configuration

Portal

1. En su cuenta de Automation, seleccione **State Configuration (DSC)**.
2. Seleccione la pestaña **Configuraciones** y, a continuación, seleccione el nombre de la configuración que se va a compilar.
3. Seleccione **Compilar**.
4. Si la configuración no tiene parámetros, se le pedirá que confirme que desea compilarla. Si la configuración tiene parámetros, se abrirá el panel **Compilar configuración** para que pueda proporcionar valores de parámetros.
5. Se abre la página Trabajo de compilación para que pueda realizar el seguimiento del estado del trabajo de compilación. También puede usar esta página para realizar un seguimiento de las configuraciones de nodo (documentos de configuración MOF) colocadas en el servidor de extracción de Azure Automation State Configuration.

Azure PowerShell

Puede usar Start-AzAutomationDscCompilationJob para empezar a compilar con Windows PowerShell. El código de ejemplo siguiente inicia la compilación de una configuración de DSC llamada **SampleConfig**.

PowerShell

```
Start-AzAutomationDscCompilationJob -ResourceGroupName 'MyResourceGroup' -
AutomationAccountName 'MyAutomationAccount' -ConfigurationName 'SampleConfig'
```

`Start-AzAutomationDscCompilationJob` devuelve un objeto de trabajo de compilación que puede usar para realizar el seguimiento del estado del trabajo. Después, puede utilizar este objeto de trabajo de compilación con Get-AzAutomationDscCompilationJob para determinar el estado del trabajo de compilación y Get-AzAutomationDscCompilationJobOutput para ver sus flujos (salida). El ejemplo siguiente inicia la compilación de la configuración SampleConfig, espera hasta que finalice y, a continuación, muestra sus flujos.

PowerShell

```
$CompilationJob = Start-AzAutomationDscCompilationJob -ResourceGroupName
'MyResourceGroup' -AutomationAccountName 'MyAutomationAccount' -ConfigurationName
'SampleConfig'

while($null -eq $CompilationJob.EndTime -and $null -eq $CompilationJob.Exception)
{
    $CompilationJob = $CompilationJob | Get-AzAutomationDscCompilationJob
    Start-Sleep -Seconds 3
}

$CompilationJob | Get-AzAutomationDscCompilationJobOutput –Stream Any
```

Declaración de los parámetros básicos

La declaración de parámetros en configuraciones de DSC, incluidos los tipos de parámetros y las propiedades, funciona igual que en Azure Automation runbooks. Para obtener más información sobre los parámetros de runbook, consulte Iniciar un runbook en Azure Automation.

En el ejemplo siguiente se usan los parámetros `FeatureName` y `IsPresent` para determinar los valores de las propiedades de la configuración del nodo **ParametersExample.sample**, que se han generado durante la compilación.

PowerShell

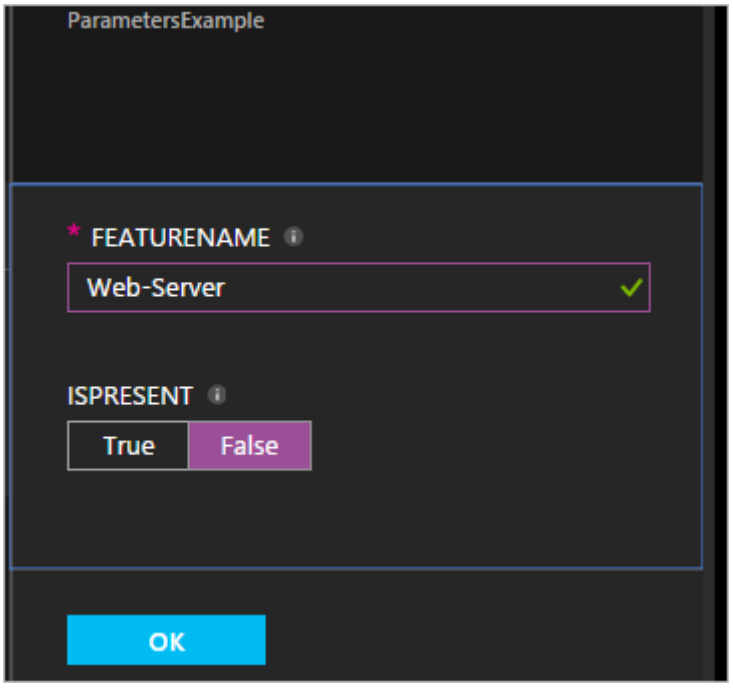
```
Configuration ParametersExample
{
    param(
        [Parameter(Mandatory=$true)]
        [string] $FeatureName,

        [Parameter(Mandatory=$true)]
        [boolean] $IsPresent
    )

    $EnsureString = 'Present'
    if($IsPresent -eq $false)
    {
        $EnsureString = 'Absent'
```

```
    }

    Node 'sample'
    {
        WindowsFeature ($FeatureName + 'Feature')
        {
            Ensure = $EnsureString
            Name   = $FeatureName
        }
    }
}
```

Puede compilar configuraciones de DSC que usen parámetros básicos en el portal de Azure Automation State Configuration o con Azure PowerShell.

Portal

En el portal, puede especificar valores de parámetro después de hacer clic **Compilar**.



Azure PowerShell

PowerShell requiere parámetros de un hashtable donde la clave coincide con el nombre del parámetro y el valor es igual al valor de parámetro.

PowerShell

```
$Parameters = @{
    'FeatureName' = 'Web-Server'
    'IsPresent' = $False
}
```

```
Start-AzAutomationDscCompilationJob -ResourceGroupName 'MyResourceGroup' -
AutomationAccountName 'MyAutomationAccount' -ConfigurationName 'ParametersExample' -
Parameters $Parameters
```

Para obtener información acerca de cómo pasar objetos `PSCredential` como parámetros, consulte Recursos de credenciales.

Compile configuraciones que contengan recursos compuestos en Azure Automation

La característica **Recursos compuestos** le permite utilizar las configuraciones de DSC como recursos anidados dentro de una configuración. Esta característica permite la aplicación de varias configuraciones a un solo recurso. Consulte Recursos compuestos: uso de una configuración de DSC como un recurso para más información acerca de los recursos compuestos.

> **Nota:**
>
> Para que las configuraciones que contienen recursos compuestos se compilen correctamente, primero debe importar en Azure Automation los recursos de DSC en los que se basan los compuestos. Agregar un recurso compuesto de DSC no es diferente de agregar ningún módulo de PowerShell a Azure Automation. Este proceso se documenta en Manage Modules in Azure Automation.

Administrar ConfigurationData al compilar configuraciones en Azure Automation

`ConfigurationData` es un parámetro de DSC integrado que le permite separar la configuración estructural de cualquier configuración específica del entorno al usar DSC de PowerShell. Para obtener más información, consulte Diferenciación de "Qué" y "Dónde" en DSC de PowerShell.

> **Nota:**
>
> Al compilar en Azure Automation State Configuration, puede usar `ConfigurationData` en Azure PowerShell, pero no en el portal de Azure.

En el siguiente ejemplo de configuración de DSC se usa `ConfigurationData` mediante las palabras clave `$ConfigurationData` y `$AllNodes`. En este ejemplo, también necesitará el módulo xWebAdministration.

PowerShell

```
Configuration ConfigurationDataSample
{
    Import-DscResource -ModuleName xWebAdministration -Name MSFT_xWebsite

    Write-Verbose $ConfigurationData.NonNodeData.SomeMessage

    Node $AllNodes.Where{$_.Role -eq 'WebServer'}.NodeName
    {
        xWebsite Site
        {
            Name         = $Node.SiteName
            PhysicalPath = $Node.SiteContents
            Ensure       = 'Present'
        }
    }
}
```

Puede compilar la configuración de DSC anterior con Windows PowerShell. El siguiente script agrega dos configuraciones de nodo al servicio de extracción de Azure Automation State Configuration: **ConfigurationDataSample.MyVM1** y **ConfigurationDataSample.MyVM3**.

PowerShell

```
$ConfigData = @{
    AllNodes = @(
        @{
            NodeName = 'MyVM1'
            Role = 'WebServer'
        },
        @{
            NodeName = 'MyVM2'
            Role = 'SQLServer'
        },
        @{
            NodeName = 'MyVM3'
            Role = 'WebServer'
        }
    )

    NonNodeData = @{
        SomeMessage = 'I love Azure Automation State Configuration and DSC!'
    }
}

Start-AzAutomationDscCompilationJob -ResourceGroupName 'MyResourceGroup' -
AutomationAccountName 'MyAutomationAccount' -ConfigurationName
'ConfigurationDataSample' -ConfigurationData $ConfigData
```

Trabajar con recursos en Azure Automation durante la compilación

Las referencias de recursos son las mismas en Azure Automation State Configuration y runbooks. Vea los siguientes artículos para más información:

- Certificados
- Conexiones
- Credenciales
- Variables

Activos de credenciales

Si una configuración tiene un parámetro que especifica un objeto `PSCredential`, use `Get-AutomationPSCredential` pasando el nombre de un recurso de credencial de Azure Automation al cmdlet para recuperar la credencial. Azure Automation pasa la credencial a la configuración.

Para mantener las credenciales seguras en una configuración de nodo, cifre las credenciales en el archivo MOF de la configuración de nodo. Debe otorgar permiso al DSC de PowerShell para mostrar credenciales en texto plano durante la generación del MOF de configuración del nodo. DSC de PowerShell no sabe que Azure Automation cifra todo el archivo MOF después de ser generado por un trabajo de compilación.

Puede indicar a DSC de PowerShell que está bien que las credenciales se muestren como texto sin formato dentro de los MOFs de configuración del nodo generados mediante datos de configuración. Debe pasar `PSDscAllowPlainTextPassword = $true` a través de `ConfigurationData` para el nombre de cada bloque de nodo que se incluya en la configuración de DSC y que utilice credenciales.

En el ejemplo siguiente se muestra una configuración de DSC que usa un recurso de la credencial de Automation.

PowerShell

```
Configuration CredentialSample
{
    Import-DscResource -ModuleName PSDesiredStateConfiguration
    $Cred = Get-AutomationPSCredential 'SomeCredentialAsset'

    Node $AllNodes.NodeName
    {
        File ExampleFile
        {
```

```
            SourcePath      = '\\Server\share\path\file.ext'
            DestinationPath = 'C:\destinationPath'
            Credential      = $Cred
        }
    }
}
```

Puede compilar la configuración de DSC anterior con PowerShell. El código de PowerShell siguiente agrega dos configuraciones de nodo al servidor de extracción de Azure Automation State Configuration: **CredentialSample.MyVM1** y **CredentialSample.MyVM2**.

PowerShell

```
$ConfigData = @{
    AllNodes = @(
        @{
            NodeName = '*'
            PSDscAllowPlainTextPassword = $True
        },
        @{
            NodeName = 'MyVM1'
        },
        @{
            NodeName = 'MyVM2'
        }
    )
}

Start-AzAutomationDscCompilationJob -ResourceGroupName 'MyResourceGroup' -
AutomationAccountName 'MyAutomationAccount' -ConfigurationName 'CredentialSample' -
ConfigurationData $ConfigData
```

> **Nota:**
>
> Cuando se completa la compilación, puede recibir el mensaje de error `The 'Microsoft.PowerShell.Management' module was not imported because the 'Microsoft.PowerShell.Management' snap-in was already imported.` Puede omitir este mensaje de forma segura.

Compilación de la configuración de DSC en Windows PowerShell

El proceso para compilar configuraciones de DSC en Windows PowerShell se incluye en la documentación de DSC de PowerShell Write, Compile y Aplicar una configuración. Puede ejecutar

este proceso desde una estación de trabajo de desarrollador o dentro de un servicio de compilación, como Azure DevOps. A continuación, puede importar los archivos MOF generados mediante la compilación de la configuración en el servicio Azure State Configuration.

La compilación en Windows PowerShell también proporciona la opción de firmar contenido de configuración. El agente de DSC comprueba una configuración de nodo firmada localmente en un nodo administrado. La comprobación garantiza que la configuración que se aplica al nodo proviene de un origen autorizado.

También puede importar configuraciones de nodo compiladas fuera de Azure. La importación incluye la compilación desde una estación de trabajo de desarrollador o en un servicio como Azure DevOps. Este enfoque tiene múltiples ventajas, como el rendimiento y la confiabilidad.

> **Nota:**
>
> Un archivo de configuración de nodo no debe tener más de 1 MB para permitir que Azure Automation importarlo.

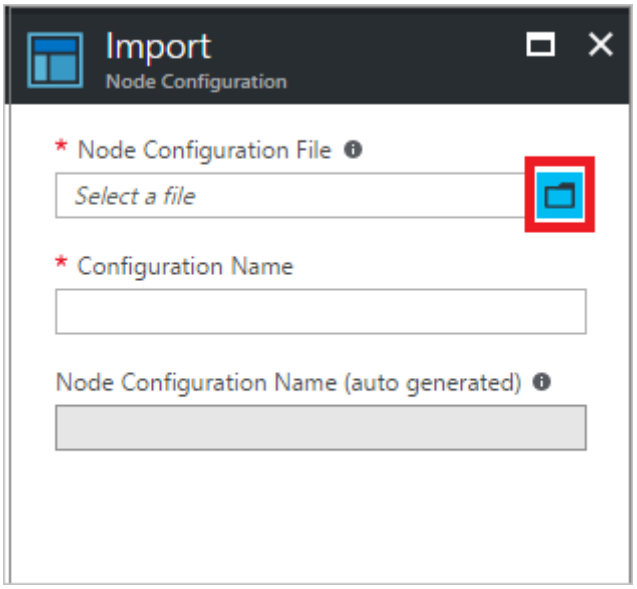
Para más información sobre la firma de configuraciones de nodo, consulte Mejoras en WMF 5.1: firma de la configuración y del módulo.

Importación de una configuración de nodo en el portal de Azure

1. En la cuenta de Automation, seleccione **State Configuration (DSC)** en **Administración de configuración**.

2. En la página State Configuration (DSC), seleccione la pestaña **Configuraciones** y, después, seleccione en **Agregar**.

3. En la página Importar, seleccione el icono de carpeta situado junto al cuadro de texto **Archivo de configuración de nodo** para buscar un archivo MOF de configuración de nodo en el equipo local.



4. Escriba un nombre en el campo **Nombre de la configuración**. Este nombre debe coincidir con el de la configuración desde la que se compiló la configuración del nodo.

5. Seleccione **Aceptar**.

Importación de una configuración de nodo con Azure PowerShell

Puede usar el cmdlet Import-AzAutomationDscNodeConfiguration para importar una configuración de nodo en la cuenta de Automation.

PowerShell

```
$importAzAutomationDscNodeConfigurationSplat = @{
    AutomationAccountName = 'MyAutomationAccount'
    ResourceGroupName = 'MyResourceGroup'
    ConfigurationName = 'MyNodeConfiguration'
    Path = 'C:\MyConfigurations\TestVM1.mof'
}
Import-AzAutomationDscNodeConfiguration @importAzAutomationDscNodeConfigurationSplat
```

Pasos siguientes

- Para empezar, consulte Get started with Azure Automation State Configuration.
- Para obtener información sobre la compilación de configuraciones de DSC para poder asignarlas a nodos de destino, consulte Compilar configuraciones de DSC en Azure Automation State Configuration.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.

- Para obtener información sobre los precios, consulte precios de Azure Automation State Configuration.
- Para obtener un ejemplo del uso de State Configuration en una canalización de implementación continua, consulte Configuración continua con Chocolatey.
- Para solucionar problemas relacionados con la compilación de configuraciones de DSC en Azure Automation State Configuration, consulte Problemas de solución de problemas de Azure Automation State Configuration.

Last updated on 20/04/2026

Corrección de servidores de Azure Automation State Configuration no conformes

> **Nota:**
>
> La Configuración de Estado de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration antes de esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> Azure Automation DSC para Linux ha sido retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

Cuando los servidores se registran con Azure Automation State Configuration, el modo de configuración se establece en `ApplyOnly`, `ApplyAndMonitor` o `ApplyAndAutoCorrect`. Si el modo no está establecido en `ApplyAndAutoCorrect`, los servidores que dejan de ser conformes permanecen en ese estado de no conformidad hasta que se corrigen manualmente.

Azure Compute ofrece una característica denominada **Ejecutar comando** que permite a los clientes ejecutar scripts dentro de las máquinas virtuales. En este documento se proporcionan scripts de ejemplo de esta característica que se pueden aplicar para corregir manualmente la variación de la configuración.

Corrección de variación en máquinas virtuales Windows con PowerShell

Puede corregir el desfase de las máquinas virtuales de Windows mediante la función de comando **Run**. Vea Ejecución de scripts de PowerShell en la máquina virtual Windows con el comando Ejecutar.

Para hacer que un nodo de Azure Automation State Configuration descargue la configuración más reciente y la aplique, use el cmdlet Update-DscConfiguration.

PowerShell

```
Update-DscConfiguration -Wait -Verbose
```

Corrección de variación en máquinas virtuales Linux

En el caso de las máquinas virtuales Linux, no tiene la opción de usar el comando **Ejecutar**. Solo puede corregir el desfase de estas máquinas repitiendo el proceso de registro.

En el caso de los nodos de Azure, puede corregir la desviación desde el portal de Azure o mediante los cmdlets del módulo Az. Los detalles sobre este proceso se documentan en Habilitar una máquina virtual mediante Azure Portal.

En el caso de los nodos híbridos, puede corregir el desfase con scripts de Python. Vea Realizar operaciones de DSC desde el equipo Linux.

Pasos siguientes

- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.
- Para ver un ejemplo del uso de la Configuración de estado de Azure Automation en una canalización de implementación continua, consulte Configurar la implementación continua con Chocolatey.

Last updated on 30/03/2026

Cómo quitar una configuración y un nodo de Automation State Configuration

> **Nota:**
>
> State Configuration de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration para esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> DSC de Automatización de Azure para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

En este artículo se explica cómo anular el registro de un nodo administrado por Automation State Configuration y quitar de forma segura una configuración de Desired State Configuration (DSC) de PowerShell de los nodos administrados. Para los nodos de Windows y Linux, debe anular el registro del nodo y eliminar la configuración del nodo. Solo para los nodos de Linux, también puede eliminar los paquetes DSC desde los nodos. Consulte Eliminación del paquete DSC de un nodo de Linux.

Anular el registro de un nodo

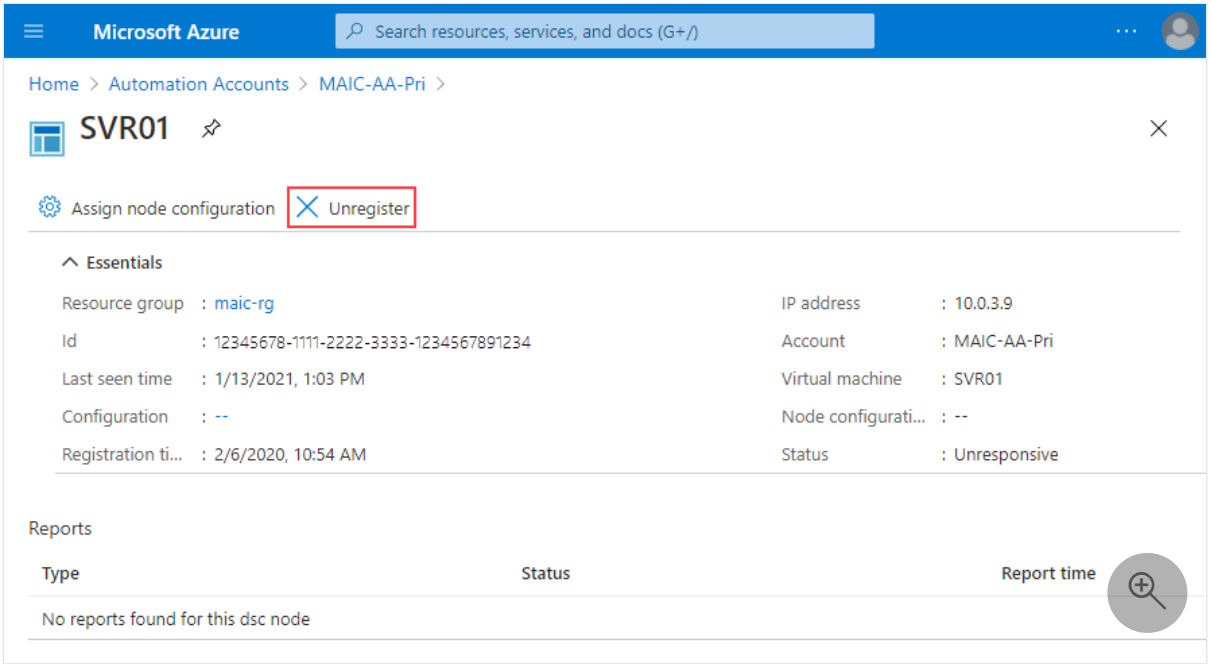
> **Nota:**
>
> Cuando se anula el registro de un nodo en el servicio, solo establece la configuración del Administrador de configuración local de forma que el nodo deja de conectarse al servicio. Esto no afecta a la configuración que se aplica actualmente al nodo y deja los archivos

relacionados en su lugar en el nodo. Después de anular el registro o eliminar el nodo, para volver a registrarlo, borre los archivos de configuración existentes. Vea Eliminación de una configuración del nodo.

Si ya no desea administrar un nodo mediante DSC, puede anular el registro desde Azure Portal o con Azure PowerShell mediante los pasos siguientes.

Azure Portal

1. Inicie sesión en Azure Portal.
2. Busque y seleccione **Cuentas de Automation**.
3. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
4. En la cuenta de Automation, seleccione **State Configuration (DSC)** en **Administración de configuración**.
5. En la página **State configuration (DSC)**, seleccione la pestaña **Nodos**.
6. En la pestaña **Nodos**, seleccione el nombre del nodo cuyo registro quiere anular.
7. En el panel de dicho nodo, seleccione **Anular registro**.



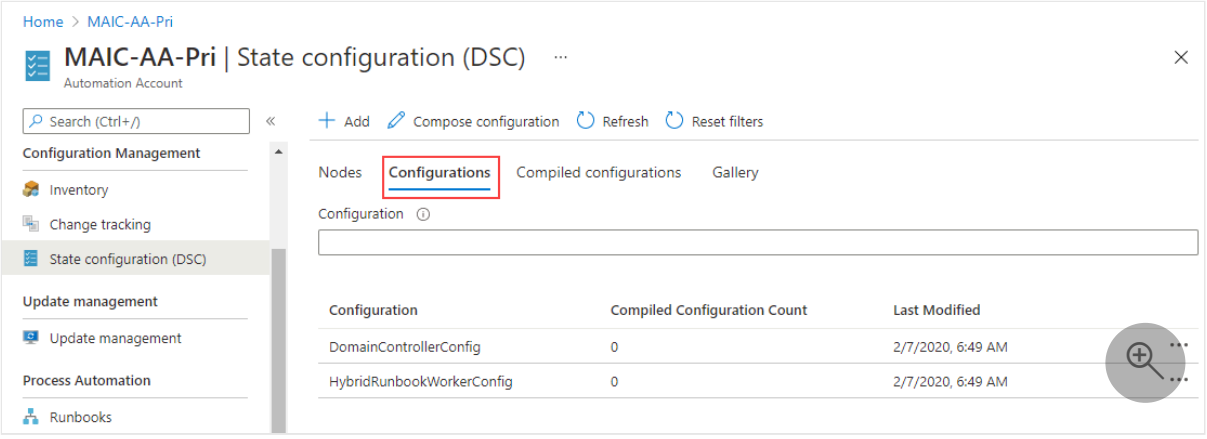
Eliminación de una configuración del nodo

Use cualquiera de los pasos siguientes para quitar un documento de configuración de DSC importado (archivo `.mof`) asignado a uno o varios nodos.

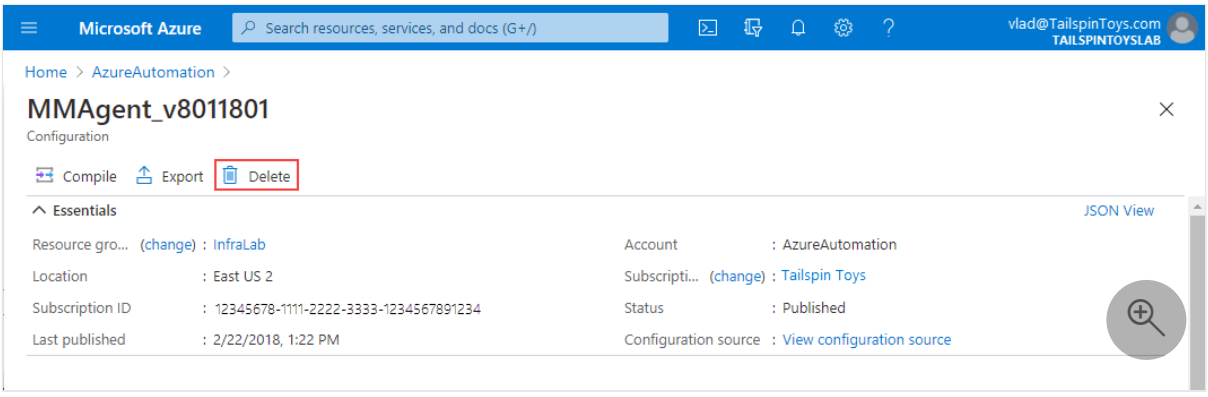
Azure Portal

Puede eliminar las configuraciones de los nodos de Windows y Linux desde Azure Portal.

1. Inicie sesión en Azure Portal.
2. Busque y seleccione **Cuentas de Automation**.
3. En la página **Cuentas de Automation**, seleccione su cuenta de Automation en la lista.
4. En la cuenta de Automation, seleccione **State Configuration (DSC)** en **Administración de configuración**.
5. En la página **State Configuration (DSC)**, seleccione la pestaña **Configuraciones** y seleccione el nombre de la configuración que quiera eliminar.



6. En la página de detalles de la configuración, seleccione **Eliminar** para quitar la configuración.



Nuevo registro de un nodo

Un nodo se puede volver a registrar del mismo modo en que se ha registrado inicialmente, con cualquiera de los métodos descritos en Habilitación de State Configuration de Azure Automation

Eliminación del paquete DSC de un nodo de Linux

Este paso es opcional. Al anular el registro de un nodo de Linux de State Configuration (DSC) no se quitan los paquetes DSC y OMI de la máquina. Use los siguientes comandos para quitar los paquetes, así como todos los registros y datos relacionados.

Para buscar los nombres de paquetes y otros detalles pertinentes, consulte el repositorio de GitHub Desired State Configuration de PowerShell para Linux.

Sistemas basados en RPM

Bash

```
rpm -e <package name>
```

Sistemas basados en DPKG

Bash

```
dpkg -P <package name>
```

Pasos siguientes

- Si quiere volver a registrar el nodo, o registrar uno nuevo, consulte Registro de una máquina virtual para su administración mediante State Configuration.
- Si quiere volver a agregar la configuración y volver a compilarla, consulte Compilación de configuraciones de DSC en Azure Automation State Configuration.

Last updated on 17/11/2025

Configuración de la implementación continua con Chocolatey

> **Nota:**
>
> State Configuration de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration para esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

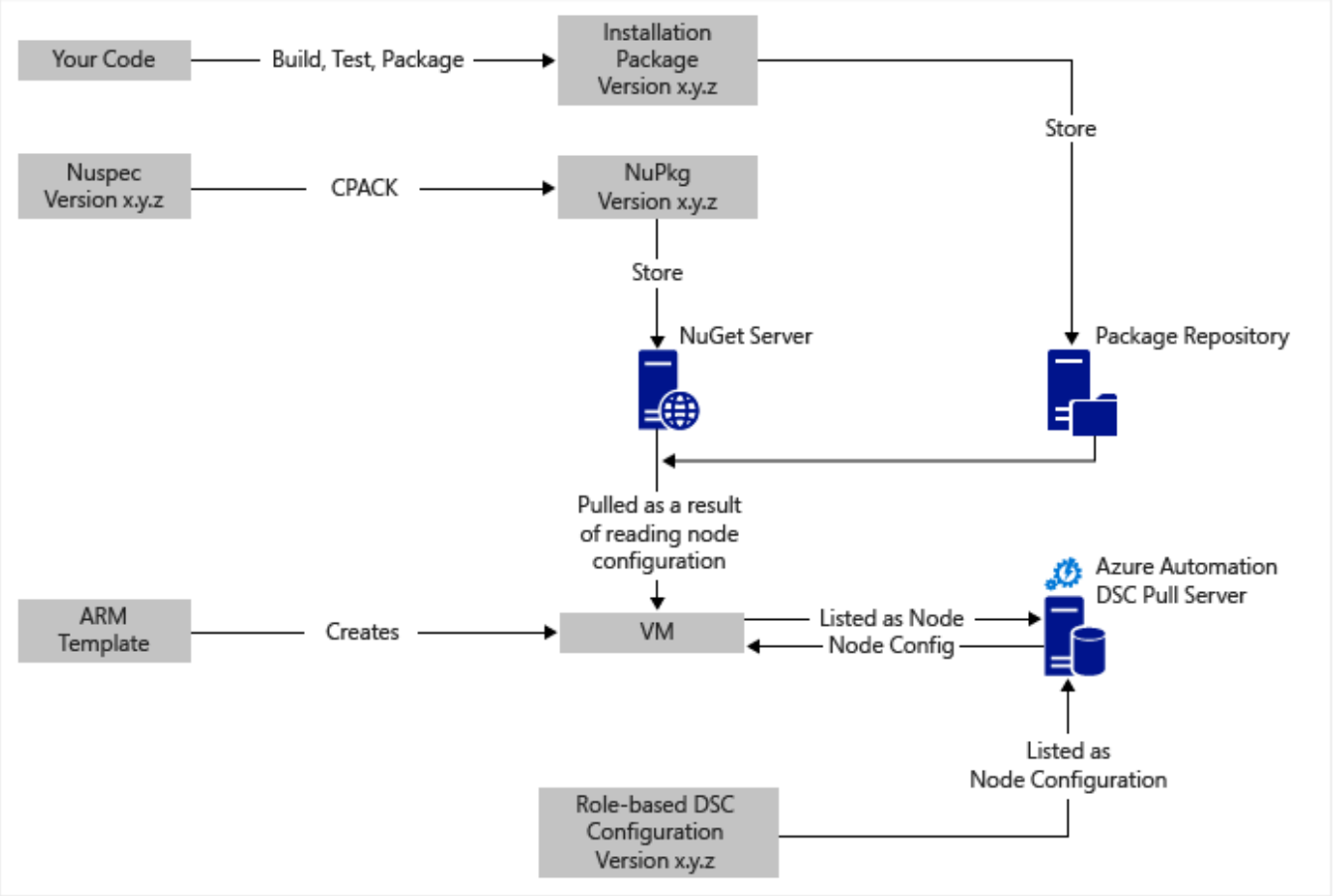
> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> DSC de Automatización de Azure para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

En el mundo de DevOps, existen muchas herramientas que ayudan en diversos puntos de la canalización de integración continua. Azure Automation State Configuration es una opción nueva, y bienvenida, que puede ser de utilidad para los equipos de DevOps.

Azure Automation es un servicio administrado de Microsoft Azure que le permite automatizar diversas tareas con runbooks, nodos y recursos compartidos, como credenciales, programaciones y variables globales. Azure Automation State Configuration amplía esta funcionalidad de automatización para incluir las herramientas Desired State Configuration (DSC) de PowerShell. Aquí hay una introducción muy buena.

En este artículo, se muestra cómo configurar la implementación continua (CD) para un equipo Windows. Puede ampliar fácilmente la técnica para incluir tantos equipos Windows como sean necesarios en el rol, por ejemplo, un sitio web y, a partir de ahí, a roles adicionales.



A nivel general

Aunque todo esto pueda parecer bastante complicado, afortunadamente puede dividirse en dos procesos principales:

- Escribir código y probarlo, y después crear y publicar paquetes de instalación para las versiones principales y secundarias del sistema;
- Crear y administrar las máquinas virtuales que instalen y ejecuten el código en los paquetes.

Una vez que estos dos procesos principales se hayan puesto en práctica, es fácil actualizar automáticamente el paquete en las máquinas virtuales a medida que se crean e implementan las nuevas versiones.

Información general sobre los componentes

Los administradores de paquetes como apt-get son muy conocidos en el mundo de Linux, pero no tanto en el de Windows. Chocolatey es un administrador de paquetes para Windows. La entrada de blog de Scott Hanselman sobre Chocolatey es una excelente introducción. Chocolatey le permite usar la línea de comandos para instalar paquetes desde un repositorio central en un sistema operativo Windows. Puede crear y administrar su propio

repositorio y Chocolatey puede instalar paquetes desde cualquier número de repositorios que designe.

DSC de PowerShell es una herramienta de PowerShell que permite declarar la configuración que quiera para un equipo. Por ejemplo, si quiere que se instale Chocolatey, se instale IIS, se abra el puerto 80 y se instale la versión 1.0.0 de su sitio web, Local Configuration Manager (LCM) de DSC implementa esa configuración. Un servidor de extracción de DSC contiene un repositorio de configuraciones para los equipos. El LCM de cada equipo comprueba periódicamente para ver si su configuración coincide con la almacenada. Después, notifica el estado o intenta equiparar la configuración del equipo a la almacenada. Puede editar la configuración almacenada en el servidor de extracción para hacer que un equipo o un conjunto de equipos se adapten a la configuración modificada.

Un recurso de DSC es un módulo de código con capacidades específicas, como la administración de redes, Active Directory o SQL Server. El recurso DSC de Chocolatey sabe cómo acceder a un servidor NuGet descargar paquetes, instalarlos y realizar otras tareas. Existen muchos otros recursos de DSC en la Galería de PowerShell. Estos módulos se instalan en el servidor de extracción de Azure Automation State Configuration para que las configuraciones puedan utilizarlas.

Las plantillas de Resource Manager proporcionan una manera declarativa de generar recursos para la infraestructura, como los siguientes:

- redes y subredes
- seguridad de red
- enrutamiento,
- equilibradores de carga,
- NIC, máquinas virtuales y otros

Para obtener una comparación entre el modelo de implementación de Resource Manager (declarativo) y el modelo de implementación clásico de Azure (imperativo), vea Diferencias entre Azure Resource Manager y la implementación clásica. En este artículo se incluye una explicación de los proveedores de recursos principales: proceso, almacenamiento y red.

Una característica clave de una plantilla de Resource Manager es su capacidad para instalar una extensión de máquina virtual durante el aprovisionamiento de la máquina virtual. Una extensión de máquina virtual posee capacidades específicas, como ejecutar un script personalizado, instalar software antivirus y ejecutar un script de configuración de DSC. Existen muchos otros tipos de extensiones de máquina virtual.

Vista rápida al diagrama

Empezando desde arriba, se escribe el código, se compila, se prueba y después se crea un paquete de instalación. Chocolatey admite diversos tipos de paquetes de instalación, como MSI, MSU o ZIP. Además, cuenta con toda la funcionalidad de PowerShell para realizar la instalación en sí si las capacidades nativas de Chocolatey no son suficientes. Coloque el paquete en un lugar accesible: un repositorio de paquetes. En este ejemplo de uso se emplea una carpeta pública en una cuenta de almacenamiento de blobs de Azure, pero vale cualquier lugar. Chocolatey funciona de forma nativa con los servidores NuGet y otros para administrar los metadatos de los paquetes. este artículo se describen las opciones. En este ejemplo de uso se utiliza NuGet. Nuspec son los metadatos de los paquetes. La información de Nuspec se compila en un NuPkg y se almacena en un servidor NuGet. Cuando la configuración solicita un paquete por su nombre y hace referencia a un servidor NuGet, el recurso de DSC de Chocolatey en la máquina virtual toma el paquete y lo instala. También se puede solicitar una versión específica de un paquete.

En la parte inferior izquierda de la imagen, se ve una plantilla de Azure Resource Manager. En este ejemplo de uso, la extensión de máquina virtual registra la máquina virtual con el servidor de extracción de Azure Automation State Configuration como nodo. La configuración se almacena en el servidor de extracción dos veces: una vez como texto sin formato y una vez compilada como un archivo MOF. En Azure Portal, MOF representa una configuración de nodo, en lugar de una configuración simple.

Es relativamente sencillo crear Nuspec, compilarlo y almacenarlo en un servidor NuGet. Para el siguiente paso de la implementación continua se necesitan las siguientes tareas únicas:

- Configurar el servidor de extracción
- Registrar los nodos con el servidor
- Crear la configuración inicial en el servidor

Solo tiene que actualizar la configuración y la configuración de nodos en el servidor de extracción al actualizar e implementar paquetes en el repositorio.

Si no empieza con una plantilla de Resource Manager, hay comandos de PowerShell que le ayudarán a registrar las máquinas virtuales con el servidor de extracción. Para más información, consulte Incorporación de máquinas para administrarlas con State Configuration de Azure Automation.

Acerca del ejemplo de uso

En este ejemplo de uso de este artículo se comienza con una máquina virtual de una imagen genérica de Windows Server 2012 R2 procedente de Azure Gallery. Puede iniciarla desde cualquier imagen almacenada y después ajustarla con la configuración de DSC. Sin embargo,

cambiar la configuración que se incluye en una imagen es mucho más difícil que actualizar dinámicamente la configuración con DSC.

No es necesario usar una plantilla de Resource Manager ni la extensión de máquina virtual para emplear esta técnica con sus máquinas virtuales. Además, sus máquinas virtuales tampoco tienen que estar en Azure para administrarlas mediante CD. Solo tiene que instalar Chocolatey y configurar el LCM en la máquina virtual para que sepa dónde está el servidor de extracción.

Cuando se actualiza un paquete en una máquina virtual que se encuentra en producción, debe sacar esa máquina virtual de la rotación mientras se instala la actualización. Existen métodos muy diversos de hacerlo. Por ejemplo, con una máquina virtual tras un equilibrador de carga de Azure, puede agregar un sondeo personalizado. Al actualizar la máquina virtual, haga que el extremo de sondeo devuelva 400. El ajuste necesario para ocasionar este cambio puede estar dentro de la configuración, al igual que el ajuste para devolver de nuevo 200 una vez completada la actualización.

El código fuente completo de este ejemplo de uso se encuentra en este proyecto de Visual Studio en GitHub.

Paso 1: Configurar la cuenta de Automation y el servidor de extracción

Ejecute los comandos siguientes En una sesión de PowerShell (`Connect-AzAccount`) con privilegios elevados:

Azure PowerShell

```
New-AzResourceGroup -Name MY-AUTOMATION-RG -Location MY-RG-LOCATION-IN-QUOTES
$newAzAutomationAccountSplat = @{
    ResourceGroupName = 'MY-AUTOMATION-RG'
    Location = 'MY-RG-LOCATION-IN-QUOTES'
    Name = 'MY-AUTOMATION-ACCOUNT'
}
New-AzAutomationAccount @newAzAutomationAccountSplat
```

Este paso tarda unos minutos mientras se configura el servidor de extracción.

Puede crear la cuenta de Automation en cualquiera de las siguientes regiones de Azure:

- Este de EE. UU. 2
- Centro-sur de EE. UU.
- US Gov - Virginia
- Oeste de Europa
- Sudeste de Asia

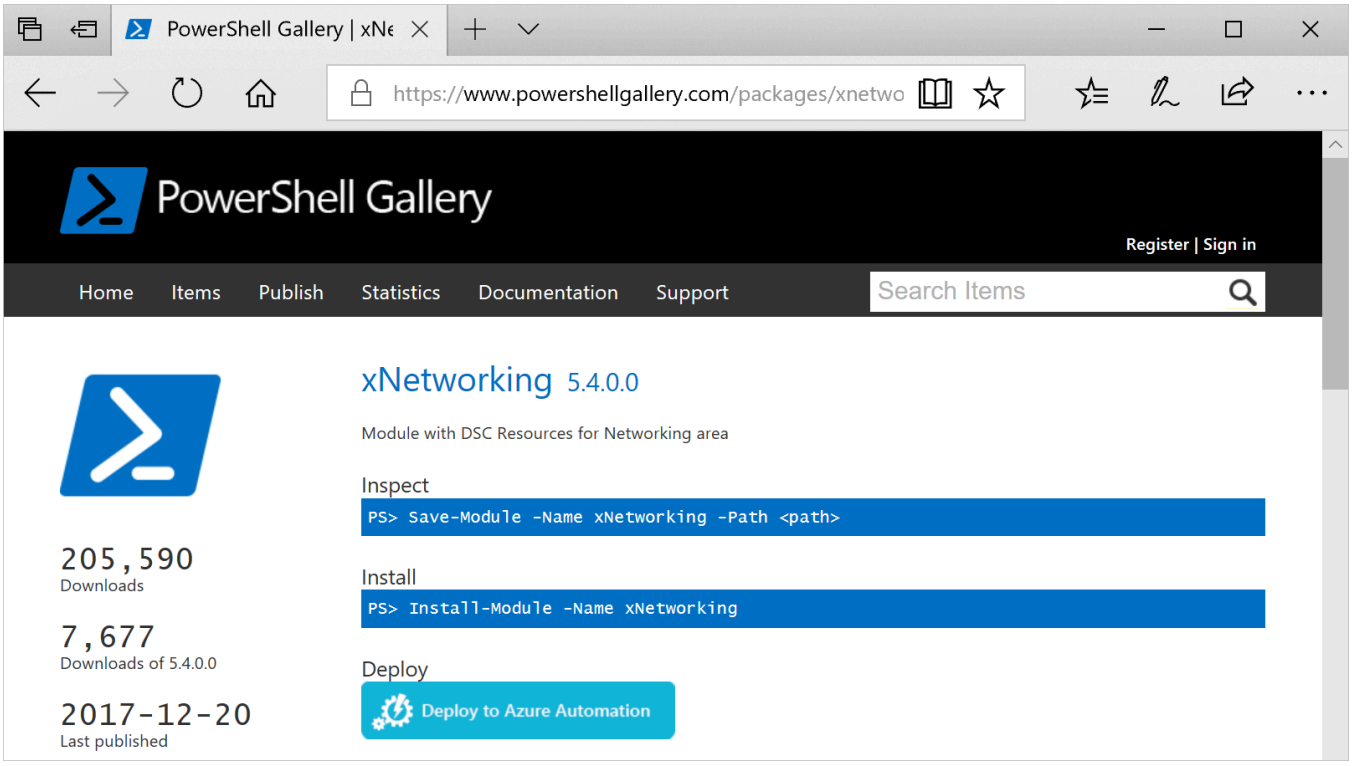
- Japón Oriental
- Centro de la India
- Sudeste de Australia
- Centro de Canadá
- Norte de Europa

Paso 2: Realizar ajustes de la extensión de máquina virtual en la plantilla de Resource Manager

Se proporcionan detalles sobre el registro de máquina virtual (con la extensión de máquina virtual de DSC de PowerShell) en esta plantilla de inicio rápido de Azure. En este paso, se registra la nueva máquina virtual con el servidor de extracción en la lista de nodos de State Configuration. Parte de este registro es especificar la configuración del nodo que se aplica al nodo. No es necesario que esta configuración de nodos exista todavía en el servidor de extracción, pero debe elegir el nombre del nodo y el nombre de la configuración. En este ejemplo, el nodo es `isvbox` y el nombre de la configuración es `ISVBoxConfig`. El nombre de configuración del nodo que especifique en `DeploymentTemplate.json` es `ISVBoxConfig.isvbox`.

Paso 3: Agregar recursos de DSC necesarios al servidor de extracción

La Galería de PowerShell puede instalar recursos de DSC en la cuenta de Azure Automation. Navegue hasta el recurso que quiera y seleccione **Implementar en Azure Automation**.



Otra técnica agregada hace poco a Azure Portal permite extraer módulos nuevos o actualizar los existentes. Después, seleccione el icono **Examinar galería** para ver la lista de módulos de la galería, profundizar en los detalles e importarlos a la cuenta de Automation. Puede usar este proceso para mantener actualizados los módulos. Además, la característica de importación comprueba las dependencias con otros módulos para asegurarse de que nada se queda sin sincronizar.

También hay un enfoque manual, que se usa solo una vez por recurso, a menos que desee actualizarlo más adelante. Para más información sobre la creación de módulos de integración de PowerShell, consulte Creación de módulos de integración para Azure Automation.

> **Nota:**
>
> La estructura de carpetas de un módulo de integración de PowerShell para un equipo Windows es algo diferente de la estructura de carpetas que espera Azure Automation.

1. Instale Windows Management Framework v5 (no es necesario para Windows 10).
2. Instale el módulo de integración.

   Azure PowerShell

   ```
   Install-Module -Name MODULE-NAME`    <—grabs the module from the PowerShell Gallery
   ```

3. Copie la carpeta del módulo de `C:\Program Files\WindowsPowerShell\Modules\MODULE-NAME` a una carpeta temporal.
4. Elimine los ejemplos y la documentación de la carpeta principal.
5. Comprima la carpeta principal y asigne al archivo comprimido el nombre de la carpeta.
6. Coloque el archivo comprimido en una ubicación HTTP accesible, como un almacenamiento de blobs en una cuenta de Azure Storage.
7. Ejecute el siguiente comando:

   Azure PowerShell

   ```
   $newAzAutomationModuleSplat = @{
       ResourceGroupName = 'MY-AUTOMATION-RG'
       AutomationAccountName = 'MY-AUTOMATION-ACCOUNT'
       Name = 'MODULE-NAME'
       ContentLinkUri = 'https://STORAGE-URI/CONTAINERNAME/MODULE-NAME.zip'
   ```

```
}
New-AzAutomationModule @newAzAutomationModuleSplat
```

El ejemplo incluido implementa estos pasos para cChoco y xNetworking.

Paso 4: Agregar la configuración de nodo al servidor de extracción

No hay nada especial que mencionar acerca de la primera vez que se importa la configuración al servidor de extracción y se compila. Todas las importaciones o compilaciones posteriores de la misma configuración tendrán exactamente el mismo aspecto. Cada vez que actualice el paquete y deba enviarlo a producción, realice este paso después de asegurarse de que el archivo de configuración sea correcto, incluida la nueva versión del paquete. Este es el archivo de configuración `ISVBoxConfig.ps1`:

PowerShell

```
Configuration ISVBoxConfig
{
    Import-DscResource -ModuleName cChoco
    Import-DscResource -ModuleName xNetworking

    Node 'isvbox' {

        cChocoInstaller installChoco
        {
            InstallDir = 'C:\choco'
        }

        WindowsFeature installIIS
        {
            Ensure = 'Present'
            Name   = 'Web-Server'
        }

        xFirewall WebFirewallRule
        {
            Direction    = 'Inbound'
            Name         = 'Web-Server-TCP-In'
            DisplayName  = 'Web Server (TCP-In)'
            Description  = 'IIS allow incoming web site traffic.'
            Enabled       = 'True'
            Action       = 'Allow'
            Protocol     = 'TCP'
            LocalPort    = '80'
            Ensure       = 'Present'
        }

        cChocoPackageInstaller trivialWeb
```

```
        {
            Name      = 'trivialweb'
            Version   = '1.0.0'
            Source    = 'MY-NUGET-V2-SERVER-ADDRESS'
            DependsOn = '[cChocoInstaller]installChoco','[WindowsFeature]installIIS'
        }
    }
}
```

El siguiente script de `New-ConfigurationScript.ps1` se ha modificado para usar el módulo Az PowerShell:

PowerShell

```
$importAzAutomationDscConfigurationSplat = @{
    ResourceGroupName = 'MY-AUTOMATION-RG'
    AutomationAccountName = 'MY-AUTOMATION-ACCOUNT'
    SourcePath = 'C:\temp\AzureAutomationDsc\ISVBoxConfig.ps1'
    Published = -Published
    Force = -Force
}
Import-AzAutomationDscConfiguration @importAzAutomationDscConfigurationSplat

$startAzAutomationDscCompilationJobSplat = @{
    ResourceGroupName = 'MY-AUTOMATION-RG'
    AutomationAccountName = 'MY-AUTOMATION-ACCOUNT'
    ConfigurationName = 'ISVBoxConfig'
}
$jobData = Start-AzAutomationDscCompilationJob
@startAzAutomationDscCompilationJobSplat

$compilationJobId = $jobData.Id

$getAzAutomationDscCompilationJobSplat = @{
    ResourceGroupName = 'MY-AUTOMATION-RG'
    AutomationAccountName = 'MY-AUTOMATION-ACCOUNT'
    Id = $compilationJobId
}
Get-AzAutomationDscCompilationJob @getAzAutomationDscCompilationJobSplat
```

Paso 5: Crear y mantener los metadatos de paquetes

Cada paquete que incluya en el repositorio de paquetes necesita un nuspec que lo describa. Se debe compilar y almacenar en el servidor NuGet. Para más información, vea [Creación de un paquete NuGet mediante la CLI de nuget.exe].

Puede usar **MyGet.org** como servidor NuGet. Puede comprar este servicio, pero existe una SKU de inicio gratuita. Para obtener instrucciones sobre cómo instalar un servidor NuGet propio

para paquetes privados, vea la documentación en Nuget.org.

Paso 6: Unirlo todo

Cada vez que una versión pasa un control de calidad y se aprueba para implementación, se crea el paquete, se actualizan nuspec y nupkg, y se implementan en el servidor NuGet. Debe actualizar la configuración (paso 4) con el nuevo número de versión. Después, envíela al servidor de extracción y compárela.

Desde ese momento, corresponderá a las máquinas virtuales que dependen de esa configuración extraer la actualización e instalarla. Cada una de estas actualizaciones es sencilla: solo una línea o dos de PowerShell. En el caso de Azure DevOps, algunas se encapsulan en tareas de compilación que se pueden encadenar de manera conjunta en una compilación. En este artículo se ofrecen más detalles. En este repositorio de GitHub se detallan las tareas de compilación disponibles.

Artículos relacionados

- Información general de DSC de Azure Automation
- Incorporación de máquinas para administrarlas con DSC de Azure Automation

Pasos siguientes

- Para obtener información general, consulte Información general de Azure Automation State Configuration.
- Para empezar a usar la característica, vea Introducción a Azure Automation State Configuration.
- Para aprender a compilar configuraciones de DSC para poder asignarlas a los nodos de destino, consulte Compilación de configuraciones de DSC en Azure Automation State Configuration.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.
- Para obtener información de precios, consulte Precios de State Configuration de Azure Automation.

Last updated on 17/11/2025

Integración de Azure Automation State Configuration con los registros de Azure Monitor

> **Nota:**
>
> State Configuration de Azure Automation se retirará el 30 de septiembre de 2027. Realice la transición a Azure Machine Configuration para esa fecha. Para obtener más información, vea la entrada de blog del anuncio. El servicio Azure Machine Configuration combina características de la extensión DSC, Azure Automation State Configuration y las características que más solicitan los clientes en sus comentarios. Azure Machine Configuration también incluye compatibilidad con las máquinas híbridas mediante servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> DSC de Automatización de Azure para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

Azure Automation State Configuration conserva los datos de estado del nodo durante 30 días. Puede enviar los datos de estado del nodo a los registros de Azure Monitor si prefiere conservar esos datos durante un período más largo. El estado de cumplimiento se puede ver en Azure Portal, o con PowerShell, tanto para los nodos como para los recursos de DSC individuales en las configuraciones de los nodos.

Los registros de Azure Monitor proporcionan mayor visibilidad operativa para los datos de Automation State Configuration y pueden ayudar a resolver incidentes con mayor rapidez. Con los registros de Azure Monitor, puede:

- Obtener información de cumplimiento de nodos administrados y recursos individuales.
- Desencadenar un correo electrónico o una alerta basados en el estado de cumplimiento.
- Escribir consultas avanzadas en los nodos administrados.
- Correlacionar el estado de cumplimiento en las cuentas de Automation.

- Usar vistas personalizadas y consultas de búsqueda para visualizar los resultados de runbook, el estado del trabajo de runbook y otras métricas o indicadores clave relacionados.

> **Nota:**
>
> Este artículo se ha actualizado recientemente para usar el término registros de Azure Monitor en lugar de Log Analytics. Los datos de registro siguen almacenándose en un área de trabajo de Log Analytics y siguen recopilándose y analizándose por el mismo servicio de Log Analytics. Estamos actualizando la terminología para reflejar mejor el rol de los registros de Azure Monitor. Consulte Azure Monitor terminology changes (Cambios en la terminología de Azure Monitor) para obtener más información.

Requisitos previos

Para empezar a enviar los informes de Automation State Configuration a los registros de Azure Monitor, necesita:

- El módulo de Az de PowerShell. Asegúrese de disponer de la versión más reciente. Si es necesario, ejecute `Update-Module -Name Az`.
- Una cuenta de Azure Automation Para más información, consulte Introducción a Azure Automation.
- Un área de trabajo de Log Analytics. Para más información, consulte Introducción a los registros de Azure Monitor.
- Al menos un nodo de Azure Automation State Configuration. Para más información, consulte Incorporación de máquinas para administrarlas con State Configuration de Azure Automation.
- Módulo xDscDiagnostics, versión 2.7.0.0 o posterior. Para conocer los pasos necesarios para la instalación, consulte Solución de problemas de Desired State Configuration (DSC) de Azure Automation.

Configuración de la integración con registros de Azure Monitor

Para comenzar a importar datos de Azure Automation State Configuration en los registros de Azure Monitor, haga lo siguiente. Para los pasos a realizar en el portal, consulte Reenvío de datos de un trabajo de Azure Automation a registros de Azure Monitor.

1. En la máquina, inicie sesión en la suscripción a Azure con el cmdlet Connect-AzAccount de PowerShell y siga las instrucciones de la pantalla.

   PowerShell

   ```
   # Sign in to your Azure subscription
   $sub = Get-AzSubscription -ErrorAction SilentlyContinue
   if (-not($sub)) {
       Connect-AzAccount
   }
   # If you have multiple subscriptions, set the one to use
   # Select-AzSubscription -SubscriptionId "<SUBSCRIPTIONID>"
   ```

2. Proporcione un valor adecuado para las variables `automationAccount` con el nombre real de la cuenta de Automation y `workspaceName` con el nombre real del área de trabajo de Log Analytics. A continuación, ejecute el script.

   PowerShell

   ```
   $automationAccount = "automationAccount"
   $law = "workspaceName"
   ```

3. Para obtener el identificador de recurso de la cuenta de Automation, ejecute los siguientes comandos de PowerShell.

   PowerShell

   ```
   # Find the ResourceId for the Automation account
   $AutomationResourceId = (Get-AzResource -ResourceType
   'Microsoft.Automation/automationAccounts' |
      Where-Object {$_.Name -eq $automationAccount}).ResourceId
   ```

4. Para obtener el identificador de recurso del área de trabajo de Log Analytics, ejecute los siguientes comandos de PowerShell.

   PowerShell

   ```
   # Find the ResourceId for the Log Analytics workspace
   $WorkspaceResourceId = (Get-AzResource `
       -ResourceType 'Microsoft.OperationalInsights/workspaces' |
       WHERE {$_.Name -eq $law}).ResourceId
   ```

5. Para configurar las opciones de diagnóstico en la cuenta de Automation para reenviar los datos de registro de estado del nodo de DSC a los registros de Azure Monitor, el siguiente cmdlet de PowerShell crea una configuración de diagnóstico con ese destino.

PowerShell

```
$setAzDiagnosticSettingSplat = @{
    ResourceId = $AutomationResourceId
    WorkspaceId = $WorkspaceResourceId
    Enabled = $true
    Category = 'DscNodeStatus'
}
Set-AzDiagnosticSetting @setAzDiagnosticSettingSplat
```

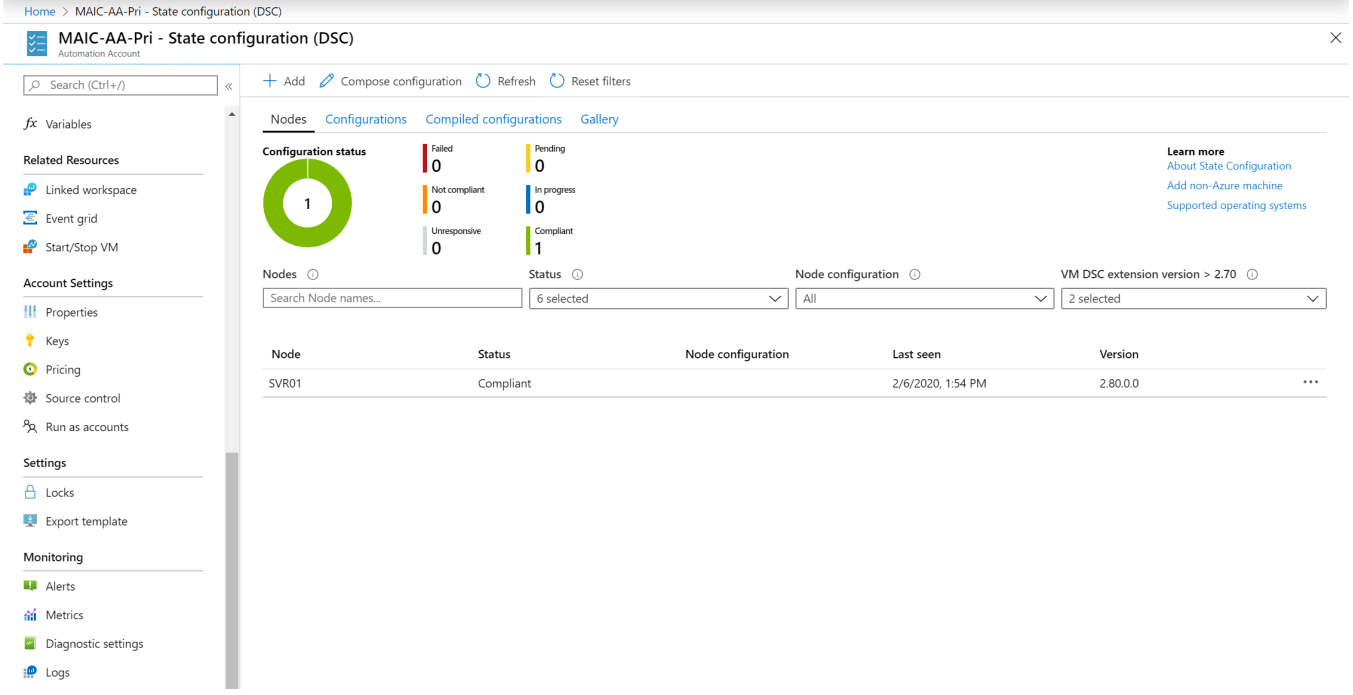
Si desea detener el reenvío de datos de registro de Azure Automation State Configuration a los registros de Azure Monitor, ejecute el siguiente cmdlet de PowerShell.

PowerShell

```
$setAzDiagnosticSettingSplat = @{
    ResourceId = $AutomationResourceId
    WorkspaceId = $WorkspaceResourceId
    Enabled = $false
    Category = 'DscNodeStatus'
}
Set-AzDiagnosticSetting @setAzDiagnosticSettingSplat
```

Visualización de los registros de State Configuration

Puede buscar las operaciones de DSC en los registros de State Configuration buscando en los registros de Azure Monitor. Después de configurar la integración con los registros de Azure Monitor para los datos de Azure Automation State Configuration, vaya a la cuenta de Automation en Azure Portal. En **Supervisión**, seleccione **Registros**.



Cierre el cuadro de diálogo **Consultas**. Se abre el panel Búsqueda de registros con una región de consulta cuyo ámbito es el recurso de la cuenta de Automation. Los registros de las operaciones de DSC se almacenan en la tabla `AzureDiagnostics`. Para buscar los nodos que no son compatibles, escriba la siguiente consulta.

Kusto

```
AzureDiagnostics
| where Category == "DscNodeStatus"
| where OperationName contains "DSCNodeStatusData"
| where ResultType != "Compliant"
```

Detalles de filtrado:

- Filtre por `DscNodeStatusData` para devolver las operaciones de cada nodo de State Configuration.
- Filtre por `DscResourceStatusData` para devolver las operaciones de cada recurso de DSC llamado en la configuración de nodo aplicada a ese recurso.
- Filtre por `DscResourceStatusData` para devolver información de errores de los recursos de DSC con errores.

Para más información sobre la creación de consultas de registros para buscar datos, consulte Introducción a las consultas de registro en Azure Monitor.

Envío de un correo electrónico cuando se produce un error en cualquier comprobación de cumplimiento de State Configuration

1. Vuelva a la consulta creada anteriormente.

2. Haga clic en el botón **+ Nueva regla de alertas** para iniciar el flujo de creación de alertas.

3. En la consulta siguiente, reemplace `NODENAME` por el nombre real del nodo administrado y pegue la consulta revisada en el cuadro de texto **Consulta de búsqueda**:

   Kusto

   ```
   AzureDiagnostics
   | where Category == "DscNodeStatus"
   | where NodeName_s == "NODENAME"
   | where OperationName == "DscNodeStatusData"
   | where ResultType == "Failed"
   ```

Si configura registros para más de una cuenta o suscripción de Automation, puede agrupar las alertas por suscripción y cuenta de Automation. Derive el nombre de la cuenta de Automation de la propiedad `Resource` en los resultados de la búsqueda de registros de `DscNodeStatusData`.

4. Consulte Creación, visualización y administración de alertas de métricas mediante Azure Monitor para completar los pasos restantes.

Búsqueda de recursos de DSC con errores en todos los nodos

Una ventaja de usar los registros de Azure Monitor es que puede buscar comprobaciones con errores en todos los nodos. Para buscar todas las instancias de errores, use la consulta siguiente:

Kusto

```
AzureDiagnostics
| where Category == "DscNodeStatus"
| where OperationName == "DscResourceStatusData"
| where ResultType == "Failed"
```

Visualización del historial de estado del nodo de DSC

Para visualizar el historial de estado del nodo de DSC con el paso del tiempo, puede usar esta consulta:

Kusto

```
AzureDiagnostics
| where ResourceProvider == "MICROSOFT.AUTOMATION"
| where Category == "DscNodeStatus"
| where ResultType != "started"
| summarize count() by ResultType
```

Esta consulta muestra un gráfico del estado del nodo con el paso del tiempo.

Registros de Azure Monitor

Diagnósticos de Azure Automation crea dos categorías de registros en los registros de Azure Monitor:

- Datos de estado del nodo (`DscNodeStatusData`)
- Datos de estado del recurso (`DscResourceStatusData`)

DscNodeStatusData

Expandir tabla

Propiedad	Descripción
TimeGenerated	Fecha y hora en que se ejecutó la comprobación de cumplimiento.
OperationName	`DscNodeStatusData`.
ResultType	Valor que indica si el nodo es conforme.
NodeName_s	El nombre del nodo administrado.
NodeComplianceStatus_s	Valor de estado que especifica si el nodo es conforme.
DscReportStatus	Valor de estado que indica si la comprobación de cumplimiento se ejecutó correctamente.
ConfigurationMode	El modo que se usa para aplicar la configuración al nodo. Los valores posibles son: • `ApplyOnly`: DSC aplica la configuración y no hace nada más. Después de la aplicación inicial de una nueva configuración, DSC no comprueba si se ha producido una desviación desde un estado configurado previamente. DSC intenta aplicar la configuración hasta que sea la correcta antes de que el valor de `ApplyOnly` surta efecto. • `ApplyAndMonitor`: este es el valor predeterminado. El LCM aplica las nuevas configuraciones. Después de la aplicación inicial de una nueva configuración, si el nodo de destino se desvía del estado deseado, DSC notifica la discrepancia en los registros. DSC intenta aplicar la configuración hasta que sea la correcta antes de que el valor de `ApplyAndMonitor` surta efecto. • `ApplyAndAutoCorrect`: DSC aplica las nuevas configuraciones. Después de la aplicación inicial de una nueva configuración, si el nodo de destino se desvía del estado deseado, DSC notifica la discrepancia en los registros y vuelve a aplicar la configuración actual.
HostName_s	El nombre del nodo administrado.
IPAddress	La dirección IPv4 del nodo administrado.
Category	`DscNodeStatus`.
Resource	El nombre de la cuenta de Azure Automation.
Tenant_g	El GUID que identifica al inquilino del autor de la llamada.
NodeId_g	GUID que identifica el nodo administrado.
DscReportId_g	GUID que identifica el informe.

Propiedad	Descripción
LastSeenTime_t	Fecha y hora en que el informe se vio por última vez.
ReportStartTime_t	Fecha y hora en que el informe se inició.
ReportEndTime_t	Fecha y hora en que el informe se completó.
NumberOfResources_d	El número de recursos de DSC que se llama en la configuración que se aplica al nodo.
SourceSystem	El sistema de origen que identifica cómo se han recopilado los datos de los registros de Azure Monitor. Siempre `Azure` para Azure Diagnostics.
ResourceId	El identificador de recurso de la cuenta de Azure Automation.
ResultDescription	La descripción del recurso de esta operación.
SubscriptionId	Identificador de la suscripción de Azure (GUID) para la cuenta de Automation.
Grupo de recursos	nombre del grupo de recursos de la cuenta de Automation.
ResourceProvider	MICROSOFT.AUTOMATION.
ResourceType	AUTOMATIONACCOUNTS.
CorrelationId	Identificador de correlación del informe de cumplimiento.

DscResourceStatusData

Expandir tabla

Propiedad	Descripción
TimeGenerated	Fecha y hora en que se ejecutó la comprobación de cumplimiento.
OperationName	`DscResourceStatusData`.
ResultType	Si el recurso es compatible.
NodeName_s	El nombre del nodo administrado.
Category	DscNodeStatus.
Resource	El nombre de la cuenta de Azure Automation.
Tenant_g	El GUID que identifica al inquilino del autor de la llamada.
NodeId_g	GUID que identifica el nodo administrado.
DscReportId_g	GUID que identifica el informe.

Propiedad	Descripción
DscResourceId_s	El nombre de la instancia del recurso de DSC.
DscResourceName_s	El nombre del recurso de DSC.
DscResourceStatus_s	Si el recurso de DSC es compatible.
DscModuleName_s	El nombre del módulo de PowerShell que contiene el recurso de DSC.
DscModuleVersion_s	La versión del módulo de PowerShell que contiene el recurso de DSC.
DscConfigurationName_s	El nombre de la configuración aplicada al nodo.
ErrorCode_s	El código de error si se produjo un error en el recurso.
ErrorMessage_s	El mensaje de error si se produjo un error en el recurso.
DscResourceDuration_d	El tiempo, en segundos, durante el que se ejecutó el recurso de DSC.
SourceSystem	Cómo los registros de Azure Monitor recopilan los datos. Siempre `Azure` para Azure Diagnostics.
ResourceId	El identificador de la cuenta de Azure Automation.
ResultDescription	La descripción de esta operación.
SubscriptionId	Identificador de la suscripción de Azure (GUID) para la cuenta de Automation.
Grupo de recursos	nombre del grupo de recursos de la cuenta de Automation.
ResourceProvider	MICROSOFT.AUTOMATION.
ResourceType	AUTOMATIONACCOUNTS.
CorrelationId	Identificador de correlación del informe de cumplimiento.

Pasos siguientes

- Para obtener información general, consulte Información general de Azure Automation State Configuration.
- Para dar los primeros pasos, consulte Introducción a Azure Automation State Configuration.
- Para aprender a compilar configuraciones de DSC para poder asignarlas a los nodos de destino, consulte Compilación de configuraciones de DSC en Azure Automation State Configuration.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.
- Para obtener información de precios, consulte Precios de State Configuration de Azure Automation.

- Para ver un ejemplo del uso de State Configuration de Azure Automation en una canalización de implementación continua, consulte Configuración de la implementación continua con Chocolatey.
- Para aprender a crear diferentes consultas de búsqueda y a revisar los registros de Automation State Configuration con los registros de Azure Monitor, consulte Búsquedas de registros en los registros de Azure Monitor.
- Para más información sobre los registros de Azure Monitor y los orígenes de recopilación de datos, consulte Introducción a la recopilación de datos de almacenamiento en los registros de Azure Monitor.

Last updated on 17/11/2025

Trabajar con el historial de versiones de la extensión de Azure Desired State Configuration

> **Nota:**
>
> Azure Automation State Configuration se retirará el 30 de septiembre de 2027, realice la transición a Azure Machine Configuration por esa fecha. Para obtener más información, consulte el anuncio del blog. El servicio Azure Machine Configuration combina características de la extensión de DSC, Azure Automation State Configuration y las características más solicitadas de los comentarios de los clientes. Azure Machine Configuration también incluye compatibilidad con máquinas híbridas a través de servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose** y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> Azure Automation DSC para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

La máquina virtual de Azure Desired State Configuration (DSC) extension se actualiza según sea necesario para admitir mejoras y nuevas funcionalidades proporcionadas por Azure, Windows Server y el marco de administración de Windows (WMF) que incluye Windows PowerShell.

En este artículo se proporciona información sobre cada versión de la extensión de máquina virtual de DSC de Azure, qué entornos admite, y comentarios y comentarios sobre las nuevas características o cambios.

La versión más reciente

ⓘ **Nota:**

Windows Server 2008, 2008 R2, 2012 y 2012 R2 alcanzaron el fin del soporte (EOS). Revise el uso y planee las actualizaciones y migraciones del sistema operativo en consecuencia. Para obtener más información, consulte Fin del soporte técnico para:

- Windows Server 2008 y Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2

Actualización local a Windows Server 2016, 2019, 2022 o 2025.

Versión 2.83

- **Fecha de publicación:**
  - Febrero de 2021
- **Compatibilidad con SO**:
  - Windows Server 2019
  - Windows Server 2016
  - Windows Server 2012 R2
  - Windows Server 2012
  - Windows Server 2008 R2 SP1
  - Windows Cliente 7/8.1/10
  - Nano Server
- **Compatibilidad con WMF:**
  - WMF 5.1
  - WMF 5.0 RTM
  - Actualización de WMF 4.0
  - WMF 4.0
- **Entorno:**
  - Azure
  - Microsoft Azure operado por 21Vianet
  - Azure para el Gobierno
- **Remarks:** Esta versión incluye una corrección para los archivos binarios sin firmar con la extensión de máquina virtual de Windows.

Versión 2.80

- **Fecha de publicación:**
  - 26 de septiembre de 2019 (Azure) | 6 de julio de 2020 (Microsoft Azure operado por 21Vianet) | 20 de julio de 2020 (Azure Government)
- **Compatibilidad con SO**:
  - Windows Server 2019
  - Windows Server 2016
  - Windows Server 2012 R2
  - Windows Server 2012
  - Windows Server 2008 R2 SP1
  - Windows Cliente 7/8.1/10
  - Nano Server
- **Compatibilidad con WMF:**
  - WMF 5.1
  - WMF 5.0 RTM
  - Actualización de WMF 4.0
  - WMF 4.0
- **Entorno:**
  - Azure
  - Microsoft Azure operado por 21Vianet
  - Azure para el Gobierno
- **Comentarios:** En esta versión, no se incluyen nuevas características.

Versión 2.76

- **Fecha de publicación:**
  - 9 de mayo de 2018 (Azure) | 21 de junio de 2018 (Microsoft Azure operado por 21Vianet, Azure Government)
- **Compatibilidad con SO**:
  - Windows Server 2016
  - Windows Server 2012 R2
  - Windows Server 2012
  - Windows Server 2008 R2 SP1
  - Windows Cliente 7/8.1/10
  - Nano Server
- **Compatibilidad con WMF:**
  - WMF 5.1
  - WMF 5.0 RTM

  - Actualización de WMF 4.0
  - WMF 4.0
- **Entorno:**
  - Azure
  - Microsoft Azure operado por 21Vianet
  - Azure para el Gobierno
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.1 (la instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.
- **Nuevas características:**
  - Se han mejorado los metadatos de extensión de los subestados y se han aplicado otras correcciones de errores.

Versiones compatibles

> ⚠ **Advertencia**
>
> De la versión 2.4 a la 2.13 se usa WMF 5.0 Public Preview, cuyos certificados de firma expiraron en agosto de 2016. Para más información sobre este escenario, consulte el siguiente **artículo de blog**.

Versión 2.75

- **Fecha de publicación:** 5 de marzo de 2018
- **Sistemas operativos admitidos:** Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 SP1, Windows Client 7/8.1/10 y Nano Server
- **Compatibilidad con WMF:** WMF 5.1, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.1 (la instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.
- **Nuevas características:**
  - Después de la aplicación de TLS 1.2 por parte de GitHub, no puedes agregar una máquina virtual a Azure Automation State Configuration utilizando plantillas DIY de

Resource Manager disponibles en Azure Marketplace, ni usar la extensión DSC para recuperar configuraciones hospedadas en GitHub. Se devuelve un error similar al siguiente durante la implementación de la extensión:

JSON

```
{
    "code": "DeploymentFailed",
    "message": "At least one resource deployment operation failed. Please list deployment operations for details. Please see https://aka.ms/arm-debug for usage details.",
    "details": [{
        "code": "Conflict",
        "message": "{
            \"status\": \"Failed\",
            \"error\": {
                \"code\": \"ResourceDeploymentFailure\",
                \"message\": \"The resource operation completed with terminal provisioning state 'Failed'.\",
                \"details\": [ {
                    \"code\": \"VMExtensionProvisioningError\",
                    \"message\": \"VM has reported a failure when processing extension 'Microsoft.Powershell.DSC'.
                    Error message: \\\"The DSC Extension failed to execute: Error downloading
                    https://github.com/Azure/azure-quickstart-templates/raw/master/dsc-extension-azure-automation-pullserver/UpdateLCMforAAPull.zip
                    after 29 attempts: The request was aborted: Could not create SSL/TLS secure channel..\\nMore information about the failure can
                    be found in the logs located under 'C:\\\\WindowsAzure\\\\Logs\\\\Plugins\\\\Microsoft.Powershell.DSC\\\\2.74.0.0' on the VM.\\\".\"
                } ]
            }
        }"
    }]
}
```

- En la nueva versión de la extensión se usa obligatoriamente TLS 1.2. Al implementar la extensión, si ya especificó `AutoUpgradeMinorVersion = true` en la plantilla de Resource Manager, la extensión se cambia automáticamente a 2.75. Para las actualizaciones manuales, especifique `TypeHandlerVersion = 2.75` en la plantilla de Resource Manager.

Versiones 2.70 - 2.72

- **Fecha de publicación:** 13 de noviembre de 2017

- **Sistemas operativos admitidos:** Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 SP1, Windows Client 7/8.1/10 y Nano Server
- **Compatibilidad con WMF:** WMF 5.1, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.1 (la instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.
- **Nuevas características:**
  - Correcciones de errores y mejoras que simplifican el uso de Azure Automation State Configuration en el portal y con una plantilla de Resource Manager. Para más información, consulte Script de configuración predeterminada en la documentación de la extensión DSC.

Versión 2.26

- **Fecha de publicación:** 9 de junio de 2017
- **Sistemas operativos admitidos:** Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 SP1, Windows Client 7/8.1/10 y Nano Server
- **Compatibilidad con WMF:** WMF 5.1, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.1 (la instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.
- **Nuevas características:**
  - Se han realizado mejoras de telemetría.

Versión 2.25

- **Fecha de publicación:** 2 de junio de 2017
- **Sistemas operativos admitidos:** Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 SP1, Windows Client 7/8.1/10 y Nano Server
- **Compatibilidad con WMF:** WMF 5.1, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.1 (la

instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.

- **Nuevas características:**
  - Se han corregido varios errores y se han agregado mejoras menores.

Versión 2.24

- **Fecha de publicación:** 13 de abril de 2017
- **Sistemas operativos admitidos:** Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 SP1 y Nano Server
- **Compatibilidad con WMF:** WMF 5.1, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.1 (la instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.
- **Nuevas características:**
  - Expone el id. de agente de DSC y el UUID de la máquina virtual como metadatos de la extensión. Se han agregado otras mejoras menores.

Versión 2.23

- **Fecha de publicación:** 15 de marzo de 2017
- **Sistemas operativos admitidos:** Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 SP1 y Nano Server
- **Compatibilidad con WMF:** WMF 5.1, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.1 (la instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.
- **Nuevas características:**
  - Se han agregado otras mejoras y correcciones de errores.

Versión 2.22

- **Fecha de publicación:** 8 de febrero de 2017

- **Sistemas operativos admitidos:** Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 SP1 y Nano Server
- **Compatibilidad con WMF:** WMF 5.1, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.1 (la instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.
- **Nuevas características:**
  - La extensión DSC ahora es compatible con WMF 5.1
  - Se han agregado otras mejoras menores.

Versión 2.21

- **Fecha de publicación:** 2 de diciembre de 2016
- **Sistemas operativos admitidos:** Windows Server 2016, Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2 SP1 y Nano Server
- **Compatibilidad con WMF:** WMF 5.1 Preview, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.0 RTM (la instalación de WMF requiere un reinicio). En el caso de Nano Server, el rol de DSC se instala en la máquina virtual.
- **Nuevas características:**
  - La extensión DSC ahora está disponible en Nano Server. Esta versión contiene principalmente cambios de código para ejecutar la extensión en Nano Server.
  - Se han agregado otras mejoras menores.

Versión 2.20

- **Fecha de publicación:** 2 de agosto de 2016
- **Sistemas operativos admitidos:** Windows Server 2016 Technical Preview, Windows Server 2012 R2, Windows Server 2012 y Windows Server 2008 R2 SP1
- **Compatibilidad con WMF:** WMF 5.1 Preview, WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure

- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016 Technical Preview; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.0 RTM (la instalación de WMF requiere un reinicio).
- **Nuevas características:**
  - Se ha agregado la compatibilidad con WMF 5.1 Preview. Cuando se publicó por primera vez, esta versión era una actualización opcional y tenía que especificar Wmfversion = '5.1PP' en Resource Manager plantillas para instalar la versión preliminar de WMF 5.1. Wmfversion = 'latest' todavía instala WMF 5.0 RTM. Para obtener más información sobre la versión preliminar de WMF 5.1, vea este blog.
  - Se han agregado correcciones y otras mejoras menores.

Versión 2.19

- **Fecha de publicación:** 3 de junio de 2016
- **Sistemas operativos admitidos:** Windows Server 2016 Technical Preview, Windows Server 2012 R2, Windows Server 2012 y Windows Server 2008 R2 SP1
- **Compatibilidad con WMF:** WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Environment:** Azure, Microsoft Azure operado por 21Vianet, Azure Government
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016 Technical Preview; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.0 RTM (la instalación de WMF requiere un reinicio).
- **Nuevas características:**
  - La extensión DSC ahora está disponible en Microsoft Azure operado por 21Vianet. Esta versión contiene correcciones para ejecutar la extensión en Microsoft Azure operada por 21Vianet.

Versión 2.18

- **Fecha de publicación:** 3 de junio de 2016
- **Sistemas operativos admitidos:** Windows Server 2016 Technical Preview, Windows Server 2012 R2, Windows Server 2012 y Windows Server 2008 R2 SP1
- **Compatibilidad con WMF:** WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016 Technical Preview; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.0 RTM (la instalación de WMF requiere un reinicio).
- **Nuevas características:**

- Haz que la telemetría no bloquee cuando ocurra un error durante la descarga del hotfix de telemetría (problema conocido de Azure DNS) o durante la instalación.
- Se ha corregido el problema intermitente que producía que la extensión dejase de procesar la configuración después de un reinicio. Esto provocaba que la extensión DSC se mantuviese en estado de "transición".
- Se han agregado correcciones y otras mejoras menores.

Versión 2.17

- **Fecha de publicación:** 26 de abril de 2016
- **Sistemas operativos admitidos:** Windows Server 2016 Technical Preview, Windows Server 2012 R2, Windows Server 2012 y Windows Server 2008 R2 SP1
- **Compatibilidad con WMF:** WMF 5.0 RTM, actualización de WMF 4.0 y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016 Technical Preview; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.0 RTM (la instalación de WMF requiere un reinicio).
- **Nuevas características:**
  - Compatibilidad con la actualización de WMF 4.0. Para obtener más información sobre la actualización de WMF 4.0, vea este blog.
  - Lógica de reintentos cuando ocurren errores durante la instalación de la extensión DSC o al aplicar una configuración de DSC después de la instalación de la extensión. Como parte de este cambio, la extensión volverá a intentar la instalación en caso de errores en una instalación anterior o volverá a establecer una configuración de DSC si se produjo un error con esta, durante un máximo de tres veces hasta alcanzar el estado de finalización (correcto o error) o si se recibe una solicitud nueva. Si se produce un error en la extensión debido a una entrada de usuario o a configuración de usuario no válidas, no se producirá el reintento. En este caso, será necesario volver a invocar la extensión con una solicitud y configuración de usuario correcta nuevas.

> **Nota:**

La extensión DSC depende del agente de máquinas virtuales de Azure para los reintentos. El agente de máquina virtual de Azure invoca la extensión con la última solicitud fallida hasta que llegue a un estado de éxito o error.

Versión 2.16

- **Fecha de publicación:** 21 de abril de 2016
- **Sistemas operativos admitidos:** Windows Server 2016 Technical Preview, Windows Server 2012 R2, Windows Server 2012 y Windows Server 2008 R2 SP1
- **Compatibilidad con WMF:** WMF 5.0 RTM y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016 Technical Preview; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.0 RTM (la instalación de WMF requiere un reinicio).
- **Nuevas características:**
  - Se ha mejorado el control de errores y se han agregado otras correcciones de errores menores.
  - Se ha agregado una nueva propiedad de configuración de la extensión DSC. Se ha añadido `ForcePullAndApply` en AdvancedOptions para permitir que la extensión DSC genere configuraciones de DSC cuando el modo de actualización es Extracción, es decir, el modo contrario al modo Inserción. Para obtener más información sobre la configuración de la extensión DSC, consulte este blog.

Versión 2.15

- **Fecha de publicación:** 14 de marzo de 2016
- **Sistemas operativos admitidos:** Windows Server 2016 Technical Preview, Windows Server 2012 R2, Windows Server 2012 y Windows Server 2008 R2 SP1
- **Compatibilidad con WMF:** WMF 5.0 RTM y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016 Technical Preview; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.0 RTM (la instalación de WMF requiere un reinicio).
- **Nuevas características:**
  - En la versión 2.14 de la extensión se incluyen cambios para instalar WMF RTM. Durante la actualización de la versión 2.13.2.0 a la 2.14.0.0 de la extensión, puede que observe que se producen errores en algunos cmdlets de DSC o en la configuración con el siguiente

error: "No se ha encontrado ninguna instancia con los valores de propiedad especificados". Para obtener más información, vea las notas de la versión de DSC. Las soluciones alternativas para estos problemas se han agregado en la versión 2.15.
  - Si ha instalado la versión 2.14 y se encuentra con alguno de los dos problemas anteriores, tendrá que realizar los pasos manualmente. En una sesión de PowerShell con privilegios elevados, ejecute los comandos siguientes:
    - `Remove-Item -Path $env:SystemRoot\system32\Configuration\DSCEngineCache.mof`
    - `mofcomp $env:windir\system32\wbem\DscCoreConfProv.mof`

Versión 2.14

- **Fecha de publicación:** 25 de febrero de 2016
- **Sistemas operativos admitidos:** Windows Server 2016 Technical Preview, Windows Server 2012 R2, Windows Server 2012 y Windows Server 2008 R2 SP1
- **Compatibilidad con WMF:** WMF 5.0 RTM y WMF 4.0
- **Entorno:** Azure
- **Remarks:** Esta versión usa DSC como se incluye en Windows Server 2016 Technical Preview; para otros sistemas operativos de Windows, instala el Windows Management Framework 5.0 RTM (la instalación de WMF requiere un reinicio).
- **Nuevas características:**
  - Usa WMF RTM.
  - Habilita la recopilación de datos con el fin de mejorar la calidad de la extensión DSC. Para más información, consulte este artículo del blog.
  - Proporciona un formato de configuración actualizado para la extensión en una plantilla de Resource Manager. Para más información, consulte este artículo del blog.
  - Se han aplicado correcciones de errores y otras mejoras.

Pasos siguientes

- Para obtener más información sobre DSC de PowerShell, vea el Centro de documentación de PowerShell.
- Examine la plantilla Resource Manager para la extensión DSC.
- Para ver más funcionalidades y recursos que puede administrar con DSC de PowerShell, examine la Galería de PowerShell.
- Para obtener detalles sobre cómo introducir parámetros confidenciales en las configuraciones, consulte Administrar credenciales de forma segura con el controlador de extensiones DSC.

Last updated on 05/05/2026

¿Qué es el Seguimiento de Cambios e Inventario de Azure?

Azure Seguimiento de cambios e inventario (CTI) es un servicio de Azure que mejora la auditoría y la gobernanza de las operaciones en invitado supervisando los cambios de configuración y proporcionando registros detallados de inventario entre servidores en Azure, locales y otros entornos en la nube. Le ayuda a detectar el desfase de configuración y a mantener la visibilidad de los recursos del sistema.

> ⓘ **Importante**
>
> Utilice el seguimiento de cambios y el inventario con la extensión de seguimiento de cambios versión 2.20.0.0 o posterior.

El servicio proporciona dos funcionalidades principales: seguimiento de cambios e inventario.

Change Tracking

- Supervisa los cambios de configuración, incluidas las modificaciones de archivos, las claves del Registro, las instalaciones de software y los servicios de Windows o los demonios de Linux.
- Proporciona registros detallados de lo que cambió y cuándo detectar rápidamente el desfase de configuración o cambios no autorizados.
- Los metadatos de seguimiento de cambios se incorporan en la tabla `ConfigurationChange` del área de trabajo de Log Analytics conectada. Para obtener más información, vea ConfigurationChange.

> ⓘ **Nota:**
>
> Seguimiento de cambios e inventario registra los datos de las aplicaciones de nivel de sistema y de usuario. Los datos de nivel de sistema siempre se registran, pero las aplicaciones de nivel de usuario solo aparecen cuando un usuario inicia sesión en una máquina. Cuando el usuario cierra la sesión, esas aplicaciones se marcan como **Quitadas**.

Inventario

- Recopila y mantiene un inventario actualizado de software instalado, detalles del sistema operativo y otros datos de configuración de la máquina en áreas de trabajo de Log Analytics conectadas.
- Proporciona una visión general de los recursos del sistema para facilitar el cumplimiento, las auditorías y el mantenimiento proactivo.
- Ingiere metadatos de inventario en la `ConfigurationData` tabla del área de trabajo de Log Analytics conectada. Para obtener más información, vea ConfigurationData.

Para obtener más información sobre la experiencia mejorada de detección e incorporación en el Seguimiento de cambios e inventario para la administración de acciones en el sistema invitado, consulta el blog de Microsoft Community Hub .

Principales ventajas de Azure Seguimiento de Cambios e Inventario

- **Compatibilidad con Azure Monitor Agent (AMA):** es compatible con el agente de Azure Monitor (AMA), que mejora la seguridad y la confiabilidad y facilita una experiencia de hospedaje múltiple para el almacenamiento de datos.
- **Compatibilidad con la extensión Change Tracking**: es compatible con la extensión Change Tracking implementada a través de Azure Policy en la máquina virtual (VM) del cliente. Después de cambiar a AMA, la extensión Change Tracking envía datos de software, archivos y registro a AMA.
- **Compatibilidad con multihoming con Azure Monitor Logs**: estandariza la administración desde un único espacio de trabajo central. Puede migrar de Registros de Azure Monitor a AMA para que todas las máquinas virtuales usen un único área de trabajo para la recopilación y el mantenimiento de los datos.
- **Administración de reglas de recopilación de datos (DCR):** usa reglas de recopilación de datos (DCR) para configurar o personalizar varios aspectos de la recopilación de datos. Un DCR es una configuración de Azure Monitor que define qué datos de supervisión se recopilan, desde qué máquinas y dónde se envía. Por ejemplo, puede cambiar la frecuencia de la colección de archivos.

Para obtener información sobre los sistemas operativos compatibles, consulte Matriz de compatibilidad y regiones para el seguimiento de cambios e inventario.

Habilitación de Azure Seguimiento de cambios e Inventario

Puede habilitar Seguimiento de cambios e inventario de las siguientes maneras:

- **Servidores habilitados para Azure Arc (máquinas que no son de Azure):** en el portal de Azure, en el **Centro de seguimiento de cambios e inventario | Máquinas**, seleccione **Directiva**, >**Tipo de definición**, >**Categoría**, > Seguimiento de cambios e inventario . En **Iniciativa**, seleccione **Activar el seguimiento de cambios y el inventario para máquinas virtuales con Arc habilitado**. Para habilitar una sola máquina, consulte Habilitación de Seguimiento de cambios e inventario para una sola máquina virtual habilitada para Azure Arc mediante el portal. Para habilitar varias máquinas, consulte Habilitación de Seguimiento de cambios e inventario para varias máquinas virtuales habilitadas para Azure Arc mediante la CLI. Para habilitarlo a gran escala, utilice el enfoque basado en directivas deploy-if-not-exists (DINE).

- **Máquina virtual de Azure única**: en el portal de Azure, seleccione la máquina virtual en el **panel Máquinas virtuales** y, a continuación, habilite Seguimiento de cambios e inventario para esa máquina virtual. Este escenario admite máquinas virtuales Linux y Windows. Para más información, consulte Habilitación de Seguimiento de cambios e inventario para una sola máquina virtual de Azure mediante el portal.

- **Varias máquinas virtuales Azure**: en el portal de Azure, seleccione varias máquinas virtuales en el **panel Máquinas** virtuales para habilitar Seguimiento de cambios e inventario para esas máquinas. Para más información, consulte Habilitación de Seguimiento de cambios e inventario para varias máquinas virtuales de Azure mediante el portal.

Qué supervisa el seguimiento de cambios y el inventario

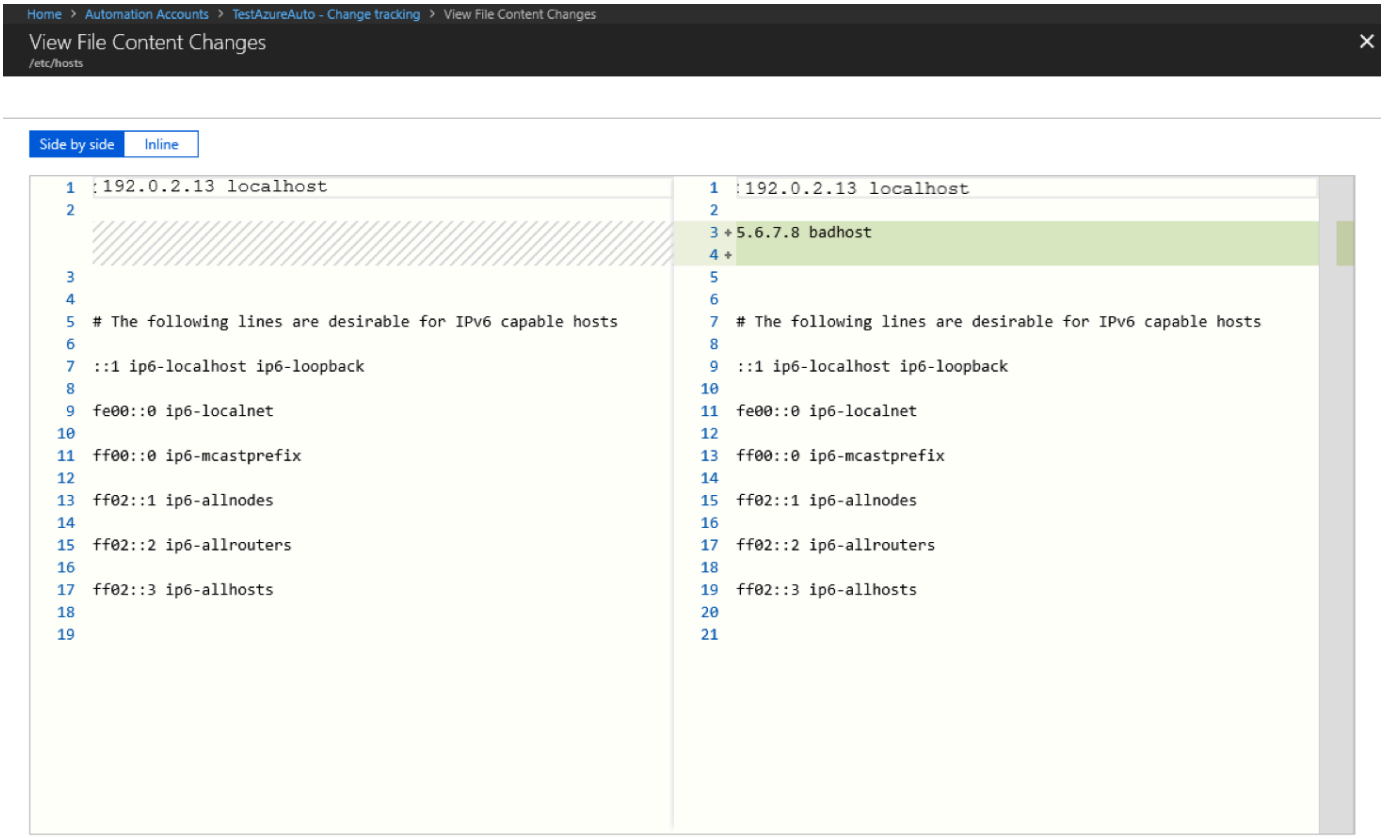
El seguimiento de cambios y el inventario de Azure supervisa los cambios en archivos, el contenido de los archivos, las claves del Registro, el software, los servicios de Windows y los demonios de Linux.

Seguimiento de los cambios de archivos

Para realizar un seguimiento de los cambios de archivos en Windows y Linux, Seguimiento de cambios e inventario usa hash SHA256 de los archivos y los compara con los hash recopilados durante el examen de inventario anterior para detectar cambios en los archivos.

Seguimiento de los cambios en el contenido del archivo

Mediante Seguimiento de cambios e inventario, puede ver el contenido de los archivos Windows y Linux. Para cada cambio de archivo, Seguimiento de cambios e inventario almacena el contenido del archivo en una cuenta de Azure Storage. Al realizar un seguimiento de un archivo, puede ver y comparar el contenido del archivo antes y después de un cambio. Puede ver el contenido en línea o en paralelo. Para obtener más información, consulte Tutorial: Cambio de un área de trabajo y configuración de la regla de recopilación de datos.



Seguimiento de las claves del Registro

Seguimiento de cambios e inventario supervisa los cambios en las claves del Registro de Windows. La supervisión de claves del Registro puede ayudar a identificar los cambios de configuración y los puntos de extensibilidad en los que el código no Microsoft puede activarse. En la tabla siguiente se enumeran las claves del Registro preconfiguradas (pero no habilitadas). Para realizar un seguimiento de estas claves, debe habilitar cada clave.

> **Nota:**
>
> Una clave del Registro es un contenedor del Registro de Windows que funciona como una carpeta en un sistema de archivos. Organiza los valores de configuración y los datos de hardware, software y usuarios. Las claves contienen valores del Registro, como archivos y subclaves.

Expandir tabla

Clave del Registro	Propósito
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Startup`	Supervisa los scripts que se ejecutan al arrancar.
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Group Policy\Scripts\Shutdown`	Supervisa los scripts que se ejecutan al apagar el equipo.
`HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run`	Supervisa las claves que se cargan antes de que el usuario inicie sesión en la cuenta de Windows. La clave se usa para aplicaciones de 32 bits que se ejecutan en equipos de 64 bits.
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Active Setup\Installed Components`	Supervisa los cambios en la configuración de la aplicación.
`HKEY_LOCAL_MACHINE\Software\Classes\Directory\ShellEx\ContextMenuHandlers`	Supervisa los controladores de menú contextual que enlazan directamente con el Explorador de Windows y normalmente se ejecutan en el mismo proceso con `explorer.exe`.
`HKEY_LOCAL_MACHINE\Software\Classes\Directory\Shellex\CopyHookHandlers`	Supervisa los controladores de copia que enlazan directamente con el Explorador de Windows y se suelen ejecutar en el proceso con `explorer.exe`.
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\ShellIconOverlayIdentifiers`	Supervisa el registro del controlador de superposición de iconos.
`HKEY_LOCAL_MACHINE\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Explorer\ShellIconOverlayIdentifiers`	Supervisa el registro del controlador de superposición de iconos para aplicaciones de 32 bits que se ejecutan en equipos de 64 bits.
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects`	Supervisa si hay nuevos complementos de objetos auxiliares de explorador para Internet Explorer. Se usa para acceder al Modelo de objetos de documento (DOM) del panel actual y para controlar la navegación.
`HKEY_LOCAL_MACHINE\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects`	Supervisa si hay nuevos complementos de objetos auxiliares de explorador para Internet Explorer. Se usa para acceder al DOM del panel actual y para controlar la navegación de aplicaciones de 32 bits que se ejecutan en equipos de 64 bits.
`HKEY_LOCAL_MACHINE\Software\Microsoft\Internet Explorer\Extensions`	Supervisa si hay nuevas extensiones de Internet Explorer, como menús de la herramienta personalizada y botones de la barra de herramientas personalizada.

Clave del Registro	Propósito
`HKEY_LOCAL_MACHINE\Software\Wow6432Node\Microsoft\Internet Explorer\Extensions`	Supervisa si hay nuevas extensiones de Internet Explorer, como menús de la herramienta personalizada y botones de la barra de herramientas personalizada para aplicaciones de 32 bits que se ejecutan en equipos de 64 bits.
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Drivers32`	Supervisa los controladores de 32 bits asociados a wavemapper, wave1 y wave2, msacm.imaadpcm, .msadpcm, .msgsm610 y vidc. Similar a la `[drivers]` sección del `system.ini` archivo.
`HKEY_LOCAL_MACHINE\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Drivers32`	Supervisa los controladores de 32 bits asociados a wavemapper, wave1 y wave2, msacm.imaadpcm, .msadpcm, .msgsm610 y vidc para aplicaciones de 32 bits que se ejecutan en equipos de 64 bits. Similar a la `[drivers]` sección del `system.ini` archivo.
`HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\KnownDlls`	Supervisa la lista de DDL del sistema conocidas o utilizadas habitualmente. La supervisión impide que las personas se aprovechen de permisos de directorio de aplicaciones débiles colocando versiones de troyanos en archivos DLL del sistema.
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify`	Supervisa la lista de paquetes que pueden recibir notificaciones de eventos de `winlogon.exe`, que es el modelo de soporte para el inicio de sesión interactivo en Windows.

Seguimiento de los cambios de software

Seguimiento de cambios e inventario realiza un seguimiento del software instalado en máquinas Windows y Linux mediante la recopilación de inventario de software y la comparación de cada ciclo de recopilación con el estado anterior. Esta funcionalidad le ayuda a detectar las instalaciones de software, las eliminaciones y los cambios de versión en las máquinas y mantener la visibilidad del inventario de software en su entorno. Para obtener más información sobre los límites y frecuencias de recopilación, consulte Matriz de compatibilidad y regiones.

Supervisar servicios de Windows y demonios de Linux

El módulo Seguimiento de cambios e inventario realiza un seguimiento de los cambios en los servicios de Windows y los demonios de Linux, incluidos los cambios relacionados con el estado y la configuración, y registra dichos cambios en eventos de modificación de configuración para su análisis y para la emisión de alertas. El seguimiento de estos cambios le ayuda a identificar los cambios relacionados con el servicio en las máquinas.

Contenido relacionado

- Para habilitar Azure Change Tracking e Inventario desde Azure Portal, consulte Inicio rápido: Habilitación de Azure Change Tracking e Inventario.
- Revise la matriz de compatibilidad y regiones de Seguimiento de cambios e inventario.
- Para deshabilitar Seguimiento de cambios e inventario mediante AMA, consulte Deshabilitar Azure Seguimiento de cambios e inventario mediante el agente de Azure Monitor.

Last updated on 08/07/2026

Matriz de compatibilidad y regiones para el Seguimiento de cambios e inventario de Azure

Azure Change Tracking e Inventario supervisa los cambios y proporciona registros de inventario para servidores en Azure, locales y otros entornos en la nube. En este artículo se resumen las configuraciones de soporte y las limitaciones al habilitar Change Tracking e Inventario, y al realizar un seguimiento de los cambios. También proporciona información sobre las regiones admitidas y las asignaciones para Seguimiento de cambios e inventario mediante el Agente de Azure Monitor (AMA).

Matriz de soporte

Expandir tabla

Componente	Se aplica a
Sistemas operativos	Windows Linux
Tipos de recursos	Máquinas virtuales de Azure (VM) Máquinas virtuales habilitadas con Azure Arc Conjuntos de escalado de máquinas virtuales
Tipos de datos	Registro de Windows Servicios de Windows Demonios de Linux Archivos Software

> **Nota:**
>
> Seguimiento de cambios e inventario actualmente no admiten la configuración para recopilar datos solo de servicios específicos (como servicios seleccionados de Windows o daemons de Linux). El servicio recopila datos de todos los servicios y no puede personalizar este comportamiento. Además, las transformaciones DCR no se admiten para DCR de Seguimiento de cambios.

Limits

En la siguiente tabla se muestran los límites de elementos sometidos a seguimiento por máquina para Change Tracking e Inventario.

Expandir tabla

Resource	Limit	Notas
Archivo	500	
Tamaño de archivo	5 MB	
Registro	250	
Software de Windows	250	No se incluye el costo de las actualizaciones de software.
Paquetes Linux	1250	
Servicios de Windows	250	
Demonios de Linux	250	

Sistemas operativos compatibles

El seguimiento de cambios e inventario es compatible con todos los sistemas operativos que cumplen con los requisitos de AMA. Para obtener una lista de las versiones del sistema operativo Windows y Linux compatibles actualmente con AMA, consulte Sistemas operativos compatibles.

Para comprender los requisitos de cliente para la seguridad de la capa de transporte (TLS), consulte TLS para Azure Automation.

Compatibilidad con la recursión

Seguimiento de cambios e inventario es compatible con la recursión, que puede usar para especificar comodines para simplificar el seguimiento entre directorios. La recursión también proporciona variables de entorno que puede utilizar para realizar un seguimiento de los archivos en entornos con nombres de unidad varios o dinámicos. En la lista siguiente se incluye información común que debe saber al configurar la recursividad:

- Utiliza caracteres comodín para rastrear varios archivos.
- Use caracteres comodín solo en el último segmento de una ruta de acceso de archivo, por ejemplo, `c:\folder\\file*` o `/etc/*.conf`.
- Tenga en cuenta que si una variable de entorno tiene una ruta de acceso no válida, la validación se realiza correctamente, pero la ruta de acceso produce un error durante la ejecución.
- Evite nombres de ruta generales al establecer la ruta. Este tipo de configuración puede hacer que se recorran demasiadas carpetas.

Recopilación de datos de Change Tracking e Inventory

En la tabla siguiente se muestra la frecuencia de recopilación de datos para los tipos de cambios que admite Change Tracking e Inventario. Los registros de inventario (ConfigurationData) se rellenan durante los ciclos de inventario programados. Cuando se detecta un cambio, solo se registra en la tabla ConfigurationChange. ConfigurationData no se actualiza entre exámenes de inventario programados.

Expandir tabla

Tipo de modificación	Frecuencia
Registro de Windows	50 minutos
Archivo de Windows	De 30 a 40 minutos
Archivo de Linux	15 minutos
Servicios de Windows	De 10 minutos a 30 minutos Valor predeterminado: 30 minutos
Software de Windows	30 minutos
Software Linux	5 minutos
Demonios de Linux	5 minutos

Nota:

La capacidad de personalizar la frecuencia de recopilación de datos es limitada. Actualmente, esta opción solo está disponible para archivos de Windows y servicios de Windows. Debe cumplir los intervalos especificados en la tabla anterior.

En la siguiente tabla se muestran los límites de elementos sometidos a seguimiento por máquina para Change Tracking e Inventario.

Expandir tabla

Resource	Limit
Archivo	500
Registro	250
Software de Windows (sin incluir revisiones)	250
Paquetes Linux	1250
Servicios de Windows	250
Demonios de Linux	500

Datos de servicios de Windows

Prerrequisitos

Para habilitar el seguimiento de los datos de servicios de Windows, debe actualizar la extensión Change Tracking y usar la extensión 2.11.0.0 o posterior.

Para máquinas virtuales de Windows Azure

PowerShell

```
- az vm extension set --publisher Microsoft.Azure.ChangeTrackingAndInventory --version
2.11.0 --ids
/subscriptions/<subscriptionids>/resourceGroups/<resourcegroupname>/providers/Microsof
t.Compute/virtualMachines/<vmname> --name ChangeTracking-Windows --enable-auto-upgrade
true
```

Para máquinas virtuales linux de Azure

PowerShell

```
– az vm extension set --publisher Microsoft.Azure.ChangeTrackingAndInventory --version
2.11.0 --ids
/subscriptions/<subscriptionids>/resourceGroups/<resourcegroupname>/providers/Microsof
t.Compute/virtualMachines/<vmname> --name ChangeTracking-Linux --enable-auto-upgrade
true
```

Para máquinas virtuales Windows habilitadas para Azure Arc

PowerShell

```
– az connectedmachine extension create --name ChangeTracking-Windows --publisher
Microsoft.Azure.ChangeTrackingAndInventory --type ChangeTracking-Windows --machine-
name <arc-server-name> --resource-group <resource-group-name> --location <arc-server-
location> --enable-auto-upgrade true
```

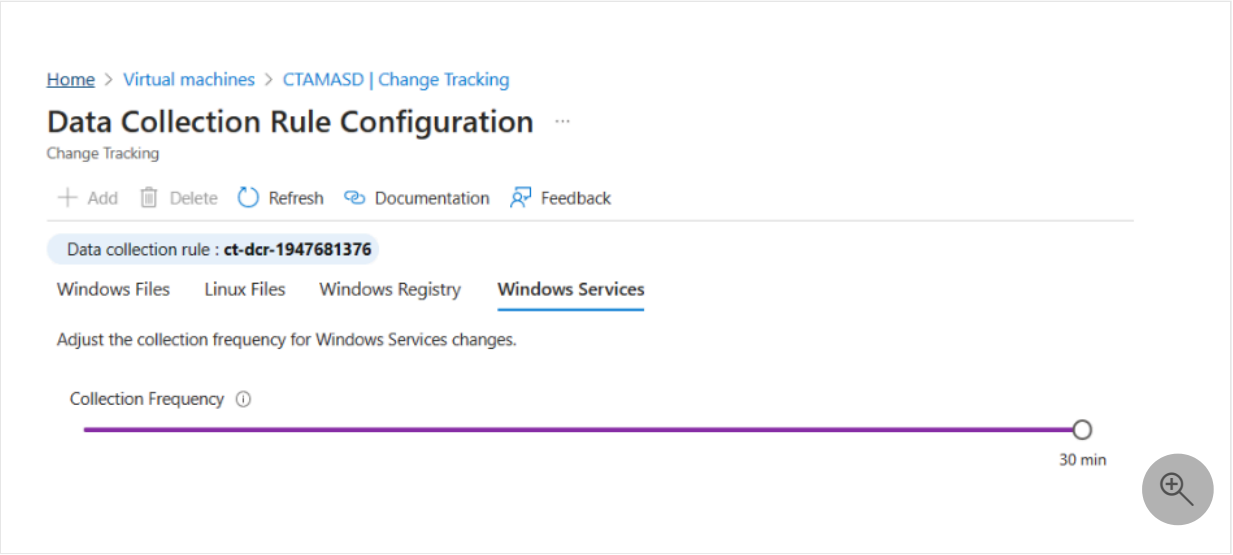
Para máquinas virtuales Linux habilitadas para Azure Arc

PowerShell

```
- az connectedmachine extension create --name ChangeTracking-Linux --publisher
Microsoft.Azure.ChangeTrackingAndInventory --type ChangeTracking-Linux --machine-name
<arc-server-name> --resource-group <resource-group-name> --location <arc-server-
location> --enable-auto-upgrade true
```

Configuración de la frecuencia

La frecuencia de recopilación predeterminado para los servicios de Windows es de 30 minutos. Para configurar la frecuencia, en **Editar** opciones, use un control deslizante en la pestaña **Servicios de Windows** .



Limitaciones actuales

Seguimiento de Cambios e Inventario usando el AMA no admite las siguientes capacidades:

- Recursión para el seguimiento del registro de Windows.
- Cualquier cosa que no sea `HKEY_LOCAL_MACHINE`. Esta limitación se produce cada vez que agregue la clave del Registro manualmente.
- Sistemas de archivos de red.
- Diferentes métodos de instalación.
- Columna **Tamaño máximo de archivo** y valores de la implementación actual.
- Recopilación de actualizaciones rápidas (hotfixes) en máquinas con Windows Server 2016 Core RS3.
- Estándares de endurecimiento para cualquier sistema o distribución Linux.
- Inventario de aplicaciones de microsoft store para cualquier sistema operativo Windows o distribuciones.

Seguimiento de Cambios y Inventario utilizando el AMA tiene las siguientes limitaciones:

- Si cambia el archivo de seguimiento, se limita a un tamaño de archivo de 5 MB o menos.
- Si el tamaño del archivo aparece >1,25 MB, entonces `FileContentChecksum` es incorrecto debido a las restricciones de memoria en el cálculo de la suma de comprobación.
- Si intenta recopilar más de 2500 archivos en un ciclo de recopilación de 30 minutos, el rendimiento de Change Tracking e Inventario podría verse degradado.
- Cuando el tráfico de red es elevado, los registros de cambios pueden tardar hasta seis horas en aparecer.
- Si modifica una configuración mientras una máquina o un servidor están apagados, podrían publicarse los cambios que pertenecían a la configuración anterior.
- Aunque no se haya producido ningún cambio, los daemons de Linux podrían mostrar un estado alterado. Este problema se debe a cómo escriben los datos de `SvcRunLevels` en la tabla ConfigurationChange de Azure Monitor.

Compatibilidad con alertas sobre el estado de configuración

Una funcionalidad clave de Change Tracking e Inventario son las alertas sobre los cambios en el estado de configuración de su entorno híbrido. Hay muchas acciones útiles que se pueden desencadenar en respuesta a las alertas. Algunos ejemplos incluyen acciones en Azure Functions, runbooks de Azure Automation y webhooks. Las alertas sobre los cambios en el archivo `c:\windows\system32\drivers\etc\hosts` de una máquina son una buena aplicación de alertas para el seguimiento de cambios y datos de inventario. Hay muchos más escenarios para las alertas, incluidos los escenarios de consulta definidos en la tabla siguiente.

Expandir tabla

Query	Descriptio
`ConfigurationChange  | where ConfigChangeType == "Files" and FileSystemPath contains " c:\\windows\\system32\\drivers\\"`	Útil para e seguimien cambios e archivos críticos del sistema.
`ConfigurationChange  | where FieldsChanged contains "FileContentChecksum" and FileSystemPath == "c:\\windows\\system32\\drivers\\etc\\hosts"`	Útil para e seguimien las modificaci de los arch de configurac de claves.
`ConfigurationChange  | where ConfigChangeType == "WindowsServices" and SvcName contains "w3svc" and SvcState == "Stopped"`	Útil para e seguimien cambios e servicios críticos del sistema.
`ConfigurationChange  | where ConfigChangeType == "Daemons" and SvcName contains "ssh" and SvcState!= "Running"`	Útil para e seguimien cambios e servicios críticos del sistema.
`ConfigurationChange  | where ConfigChangeType == "Software" and ChangeCategory == "Added"`	Útil para entornos c necesitan configurac de softwar bloqueada

Query	Descriptio
`ConfigurationData  | where SoftwareName contains "Monitoring Agent" and CurrentVersion!= "8.0.11081.0"`	Resulta úti para ver qu máquinas tienen instaladas versiones d software obsoletas o compatible Esta consu notifica el último esta de configurac notificado, no notifica cambios.
`ConfigurationChange  | where RegistryKey == @"HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\QualityCompat"`	Útil para e seguimient los cambio claves de antivirus fundament
`ConfigurationChange  | where RegistryKey contains @"HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Services\\SharedAccess\\Parameters\\FirewallPolicy"`	Útil para e seguimient los cambio la configurac del firewal

Regiones y asignaciones admitidas para Seguimiento de cambios e inventario con el Agente de Azure Monitor

En la tabla siguiente se enumeran las regiones y asignaciones admitidas.

Expandir tabla

Geografía	Región del área de trabajo del agente de Azure Monitor
Asia Pacífico	Este de Asia Sudeste asiático
Australia	Este de Australia Sudeste de Australia
Brasil	Sur de Brasil

Geografía	Región del área de trabajo del agente de Azure Monitor
Canadá	Canada Central
Europa	Norte de Europa Oeste de Europa
Francia	Centro de Francia
Alemania	Centro-oeste de Alemania
India	Centro de la India
Italia	Norte de Italia
Japón	Japón Oriental
Corea	Centro de Corea del Sur
Noruega	Norway East
España	Centro de España
Suecia	Centro de Suecia
Suiza	Norte de Suiza
Emiratos Árabes Unidos	Norte de Emiratos Árabes Unidos
Reino Unido	UK South
Gobierno de EE. UU. [1]	Gobierno de EE.UU. Virginia Gobierno de EE.UU. Arizona
EE. UU.	Este de EE. UU. Este de EE. UU. 2 Oeste de EE. UU. Oeste de EE. UU. 2 Centro Norte de EE. UU. Centro de EE. UU. Centro Sur de EE. UU. Centro Oeste de EE. UU.

[1] Actualmente, la incorporación solo se admite a través de Azure Portal.

Contenido relacionado

- Para habilitar Change Tracking e Inventario desde Azure Portal, consulte Inicio rápido: Habilitación de Azure Change Tracking e Inventario.

Last updated on 26/05/2026

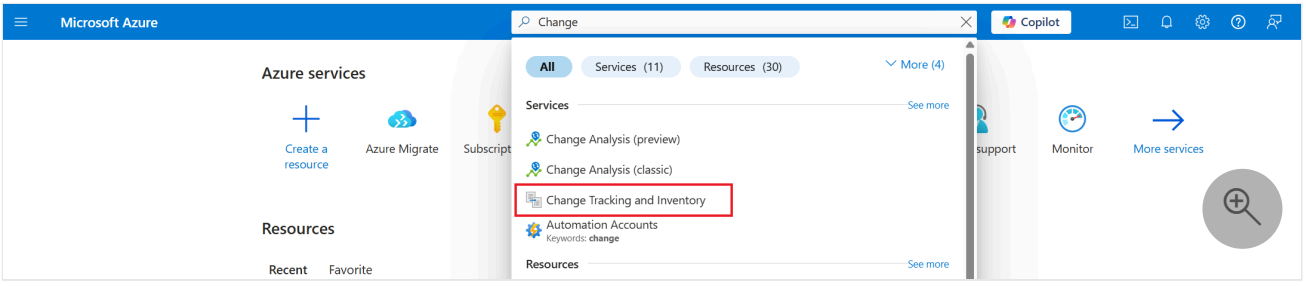
Habilitación de Seguimiento de cambios e inventario a gran escala mediante Azure Portal

Se aplica a: ✔ máquinas virtuales Windows ✔ máquinas virtuales Linux ✔ Registro de Windows ✔ Archivos de Windows ✔ Archivos de Linux ✔ Software de Windows ✔ máquinas habilitadas para Arc

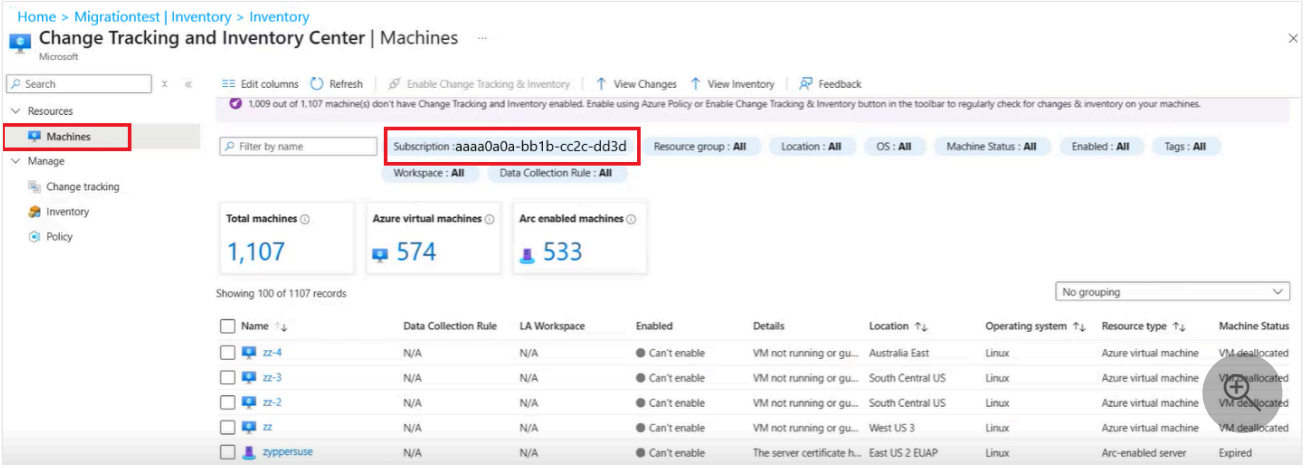
En este artículo se proporcionan procedimientos que muestran cómo habilitar Azure Change Tracking e Inventario a escala mediante el panel **Máquinas** del portal de Azure.

Implementación a gran escala

1. En Azure Portal, busque **Seguimiento de cambios e inventario**.



2. En la pestaña **Recursos** , seleccione **Máquinas**. En función de la suscripción seleccionada, las máquinas habilitadas para Change Tracking e Inventario aparecen en la columna **Habilitado** .

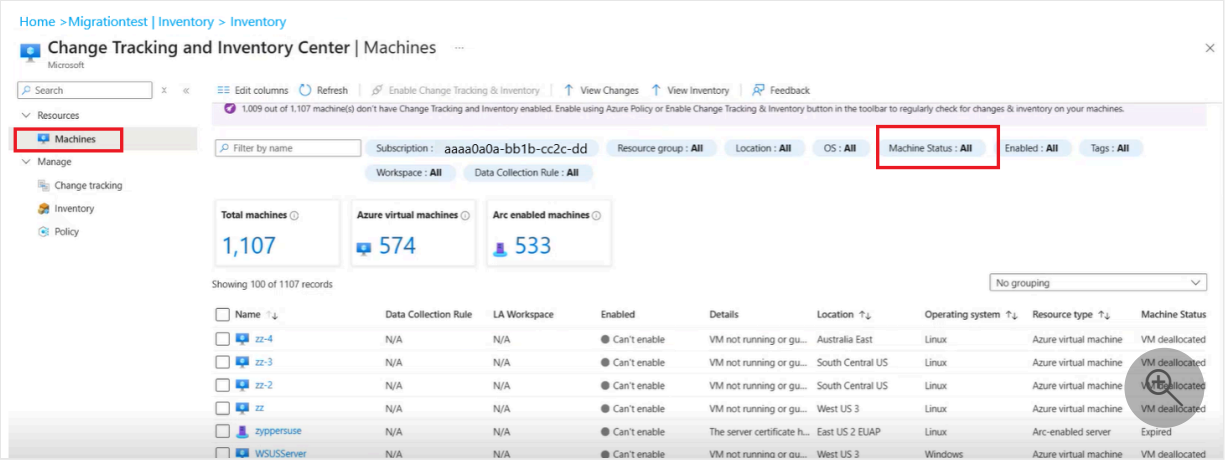


En la parte superior del panel, también verá un banner que muestra el número total de máquinas de la suscripción seleccionada que están habilitadas para Change Tracking y para

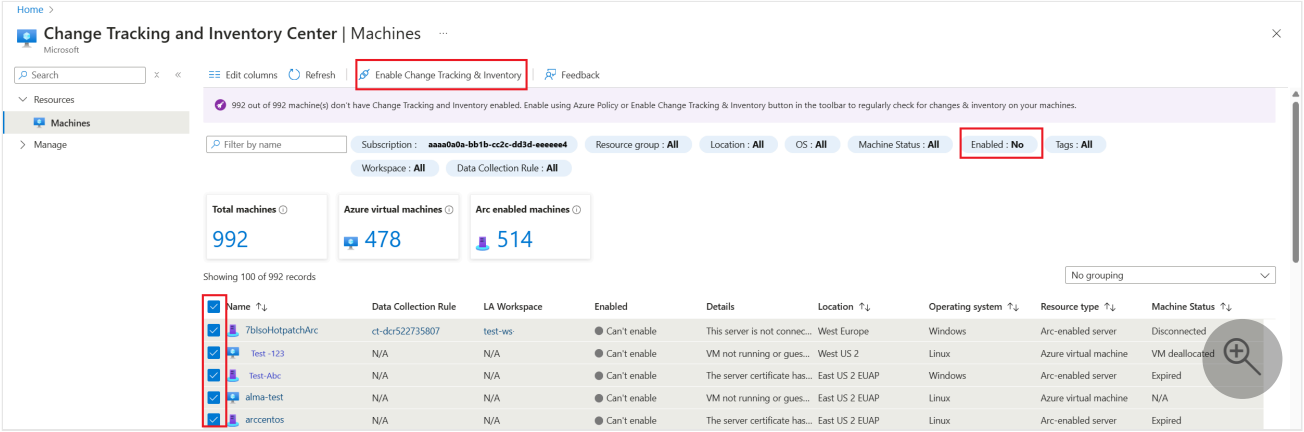
el Inventario.

3. En los filtros, seleccione **Habilitado** para ver las opciones:

   - Seleccione **Sí** para ver las máquinas que están habilitadas con Seguimiento de Cambios e Inventario.
   - Seleccione **No** para ver las máquinas que no están habilitadas con Change Tracking e Inventario.
   - Elija **Seleccionar todo** para ver todas las máquinas de la suscripción seleccionada con o sin Change Tracking e Inventario habilitado.



4. Para habilitar Change Tracking e Inventario a escala, siga estos pasos:
   a. En la columna **Filtros habilitados** , seleccione **No**.
   b. En la columna Machine Status filters (**Filtros de estado** de la máquina), seleccione **VM running (Máquina virtual en ejecución)** y **Connected (Conectado).**
   c. En la columna **Nombre** , seleccione todas las máquinas para ver cuáles están listas para habilitarse.
   d. Seleccione todas las máquinas para las que quiere habilitar Change Tracking e Inventario y, a continuación, seleccione **Habilitar Change Tracking e Inventario**.



Contenido relacionado

- Aprender a habilitar el seguimiento de cambios e inventario a escala mediante una directiva de Azure.

Last updated on 01/05/2026

Habilitar Azure Change Tracking & Inventory a escala para máquinas virtuales de Azure mediante Azure Policy.

Se aplica a: ✔ Máquinas virtuales de Windows ✔ Máquinas virtuales de Linux ✔ Registro de Windows ✔ Archivos de Windows ✔ Archivos de Linux ✔ Software de Windows

En este artículo se detallan los procedimientos para habilitar Seguimiento de cambios e inventario de Azure a gran escala mediante Azure Policy.

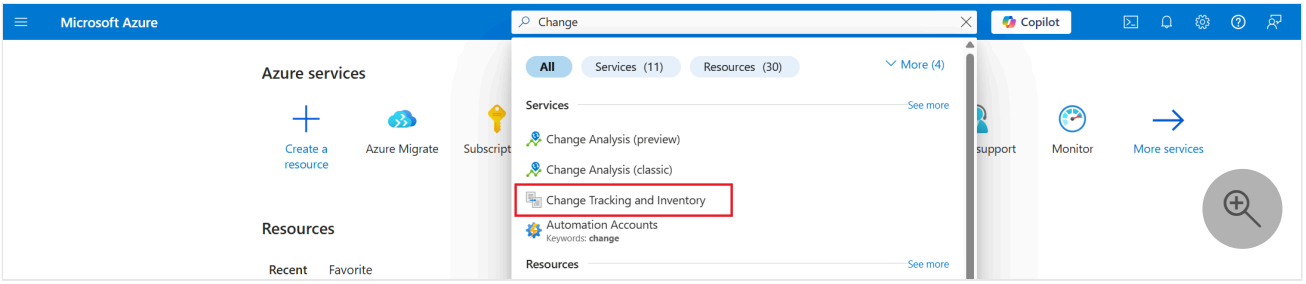
Prerrequisito

Antes de habilitar Seguimiento de cambios e inventario, asegúrese de crear una regla de recopilación de datos (DCR) o use una existente.

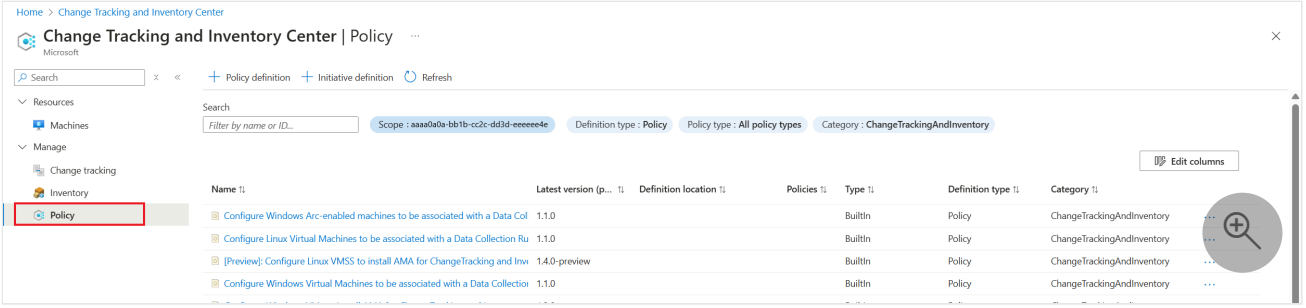
Habilitar el Seguimiento de cambios e inventario de Azure a escala

Al utilizar la directiva Implementar si no existe (DINE), puede habilitar el seguimiento de cambios con el agente de Azure Monitor a gran escala y de la manera más eficiente.

1. Inicie sesión en Azure Portal y seleccione **Seguimiento de cambios e inventario**.



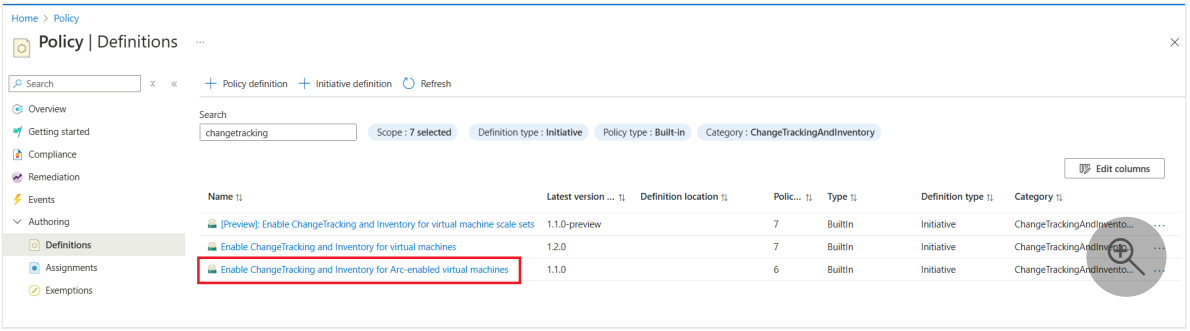
2. En el **Centro de seguimiento de cambios e inventario | Panel Máquinas** , en **Administrar**, seleccione **Directiva**.



3. En el **Centro de seguimiento de cambios e inventario | Panel directiva** , en el filtro **Tipo de definición**, seleccione **Iniciativa**. En el filtro **Categoría**, seleccione **Change Tracking y Inventario** para ver tres políticas.

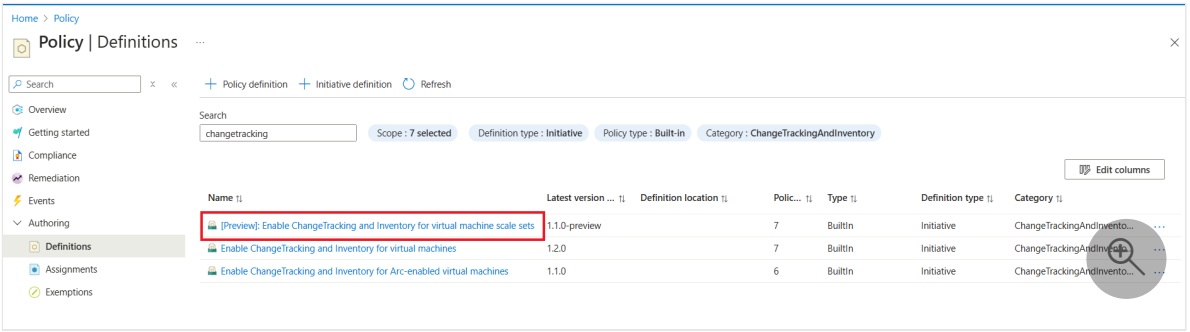
   - Máquinas virtuales habilitadas para Azure Arc

     - Seleccione **Habilitar Seguimiento de cambios e inventario para máquinas virtuales habilitadas para Arc**.



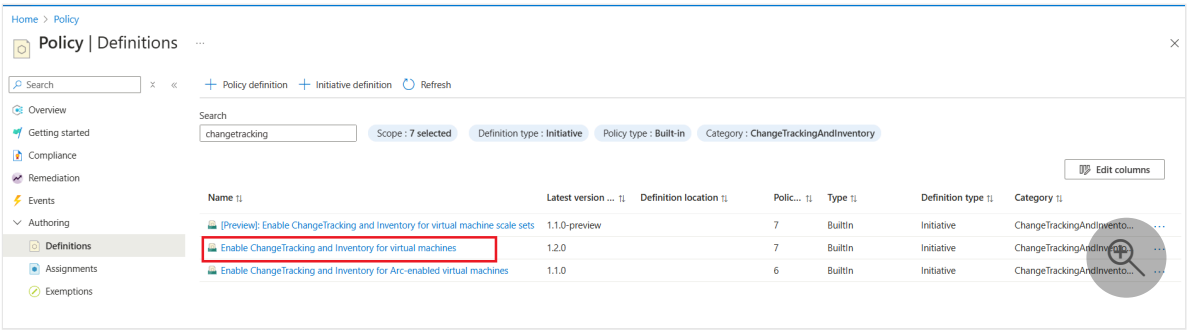
   - Conjuntos de escalado de máquinas virtuales de Azure

     - Seleccione **Habilitar Seguimiento de cambios e inventario para los conjuntos de escalado de máquinas virtuales**.



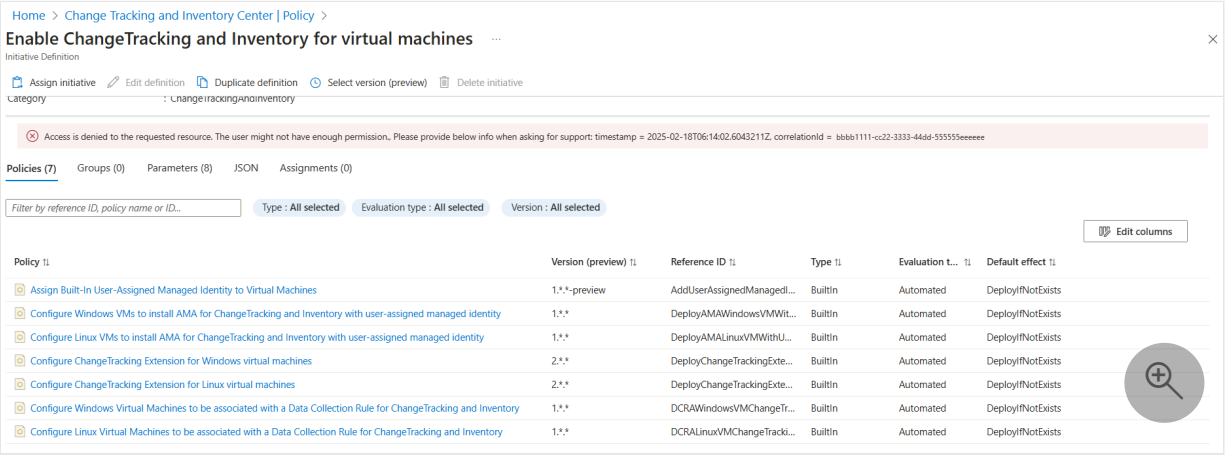
   - Máquinas virtuales

     - Seleccione **Habilitar Seguimiento de Cambios e Inventario para máquinas virtuales**.



4. Seleccione **Habilitar ChangeTracking e Inventario para máquinas virtuales** para habilitar Change Tracking en máquinas virtuales de Azure. Este paso incluye tres directivas. Cada directiva viene determinada por el tipo de sistema operativo de la máquina seleccionada:

   - **Asignación de una identidad administrada integrada asignada por el usuario a máquinas virtuales**
   - **Configuración de la extensión ChangeTracking para máquinas virtuales Windows**
   - **Configuración de la extensión ChangeTracking para máquinas virtuales Linux**



5. Seleccione **Asignar iniciativa** para asignar la directiva a un grupo de recursos. Una directiva de ejemplo es **Asignar identidad administrada asignada por el usuario incorporada a máquinas virtuales**.

> ⓘ **Nota:**
>
> El grupo de recursos contiene máquinas virtuales. Al asignar la directiva, se habilita el seguimiento de cambios a gran escala en un grupo de recursos. Las máquinas virtuales que se incorporan al mismo grupo de recursos automáticamente tienen habilitado Change Tracking.

6. En el panel **Habilitar Seguimiento de Cambios e Inventario para máquinas virtuales**, escriba las siguientes opciones:
   a. En la pestaña **Aspectos básicos** , puede definir el ámbito. Seleccione los puntos suspensivos para configurar un ámbito.
   b. En el panel **Ámbito**, escriba **Suscripción** y **Grupo de Recursos**.
   c. En la pestaña **Parámetros** , seleccione la opción **en Bring Your Own User-Assigned Managed Identity**.
   d. Escriba el valor del **Identificador de recurso de la regla de recopilación de datos**. Obtenga más información sobre cómo obtener el identificador de recurso de la regla de recopilación de datos después de crear la regla de recopilación de datos.
   e. Selecciona **Revisar + crear**.

Contenido relacionado

- Obtenga más información sobre cómo habilitar a escala el Seguimiento de Cambios y el Inventario mediante el portal de Azure.

Last updated on 12/01/2026

Inicio rápido: Habilita el seguimiento de cambios e inventario de Azure mediante el portal de Azure

Se aplica a: ✔ Máquinas virtuales Windows ✔ Máquinas Virtuales Linux ✔ Registro de Windows ✔ Archivos de Linux ✔ Software de Windows ✔ Cambios de contenido del archivo

En este artículo se describe cómo habilitar Seguimiento de cambios e inventario de Azure para una y varias máquinas virtuales de Azure desde Azure Portal.

Prerrequisitos

- Una suscripción de Azure. Si aún no tiene una, puede activar las ventajas del suscriptor de MSDN o registrarse para obtener una cuenta gratuita.
- Una máquina virtual configurada en la región especificada.

Habilitar el seguimiento de cambios e inventario para única y múltiples máquinas virtuales de Azure y máquinas virtuales habilitadas para Azure Arc desde el portal de Azure

En las secciones siguientes se muestra cómo habilitar Seguimiento de cambios e inventario en una o varias máquinas virtuales de Azure y en máquinas virtuales habilitadas para Azure Arc desde Azure Portal.

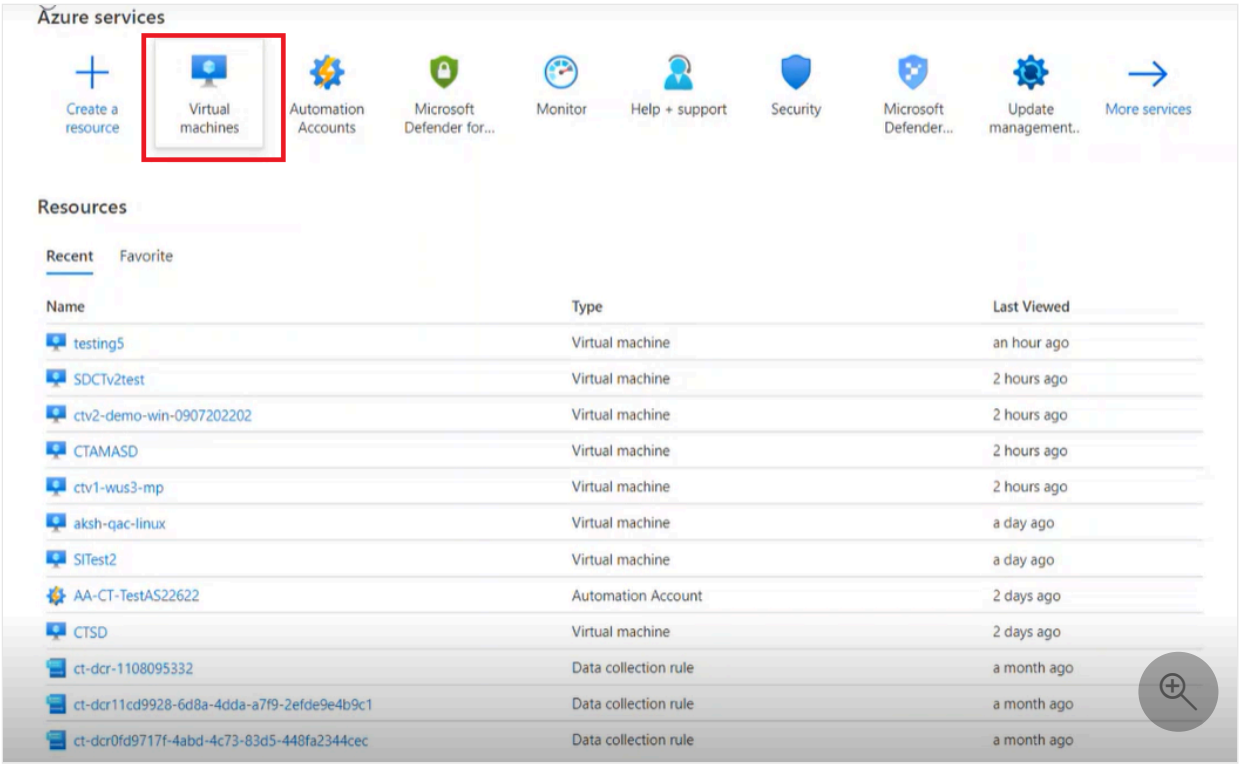
Una sola máquina virtual de Azure y máquina virtual habilitada para Azure Arc

En esta sección se proporcionan procedimientos detallados sobre cómo habilitar Change Tracking en una sola máquina virtual Azure y una sola máquina virtual habilitada para Azure Arc.

Habilitar el seguimiento de cambios e inventario para una única máquina virtual de Azure mediante el portal

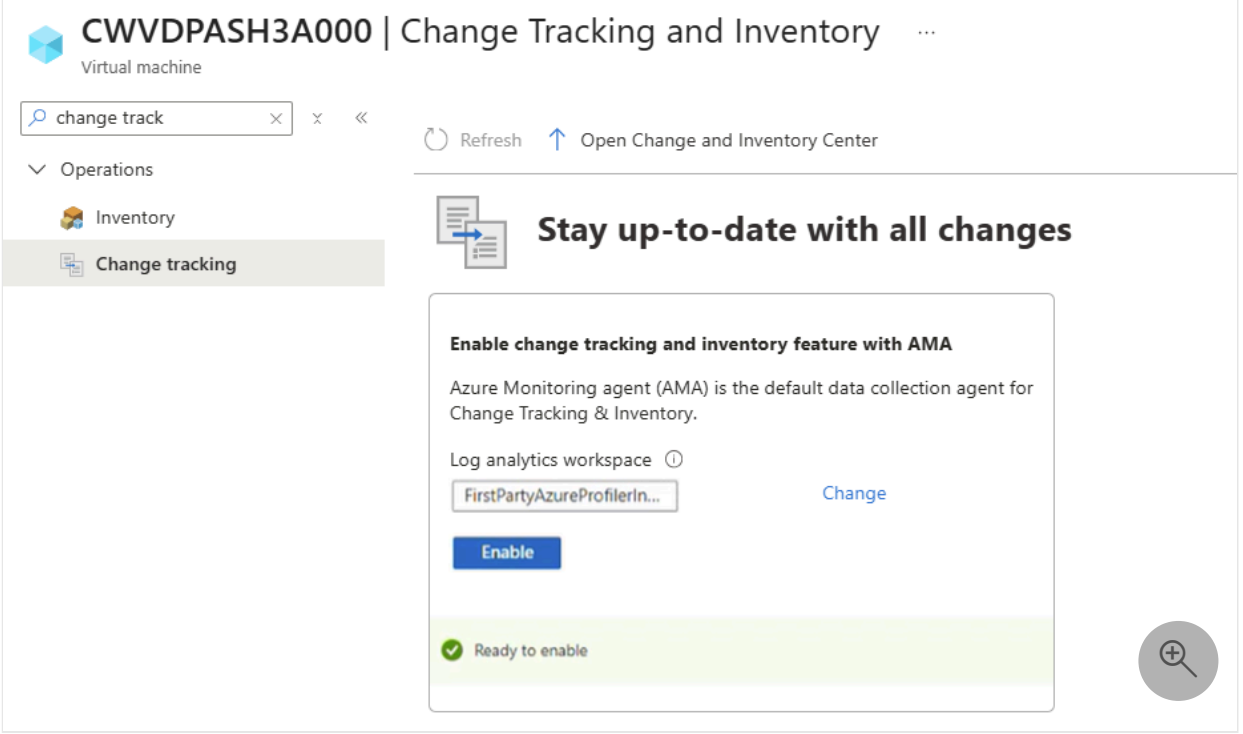
Para habilitar Change Tracking e Inventario para una sola máquina virtual de Azure mediante el portal, siga estos pasos:

1. Inicie sesión en Azure Portal y vaya a **Máquinas virtuales**.



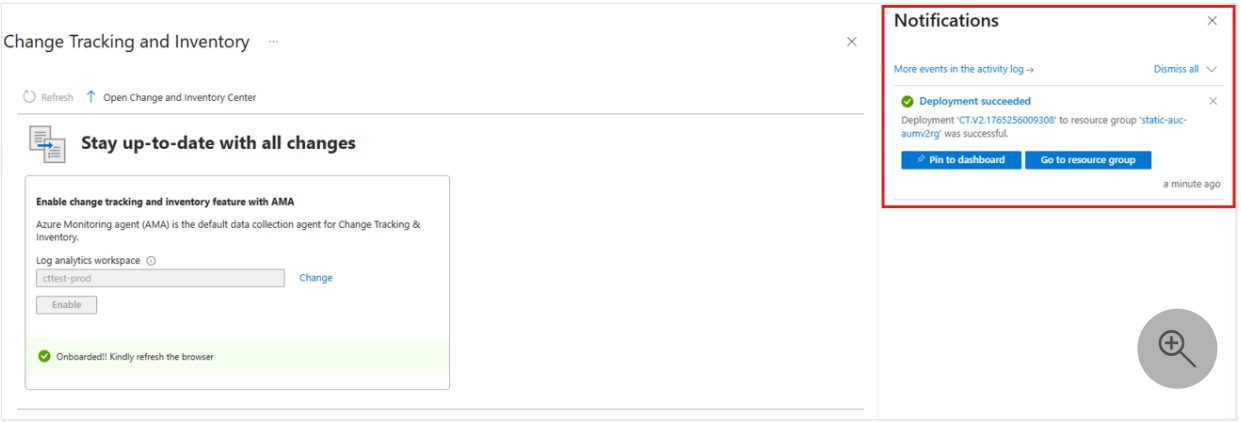
Seleccione la máquina virtual para la que desea habilitar Change Tracking.

2. En la barra de búsqueda, escriba **Change tracking**. Seleccione **Seguimiento de cambios** para ver el **panel de Seguimiento de Cambios e Inventario**.



3. En el panel **Mantener up-to-date con todos los cambios** , seleccione la **opción Habilitar seguimiento de cambios e inventario con AMA** y seleccione **Habilitar**.

La implementación de Seguimiento de Cambios e Inventario comienza con una notificación en la esquina superior derecha del panel.

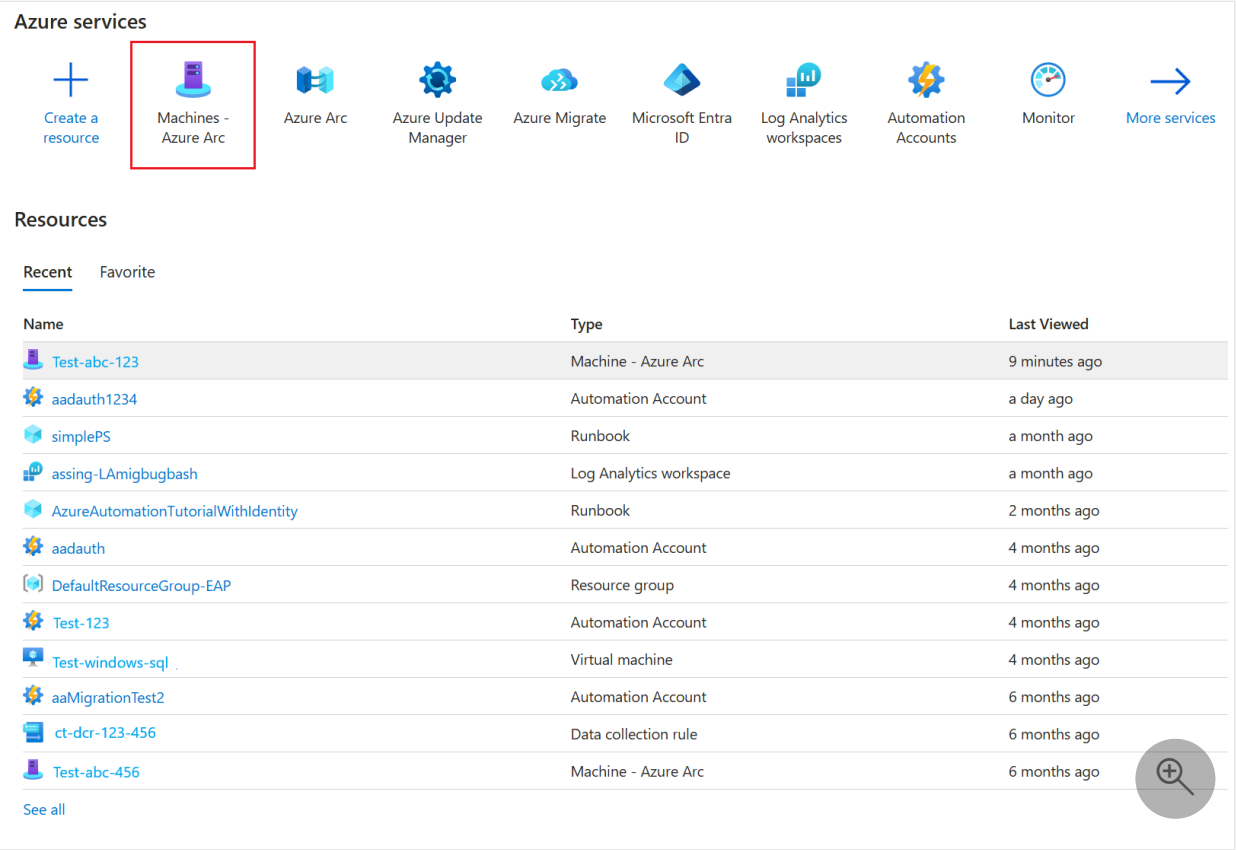


> **Nota:**
>
> Normalmente, las máquinas virtuales tardan hasta dos o tres minutos en incorporarse correctamente y habilitarlas. Después de habilitar una máquina virtual para Change Tracking, puede realizar cambios en los archivos, registros o software de la máquina virtual específica.

Habilitar el seguimiento de cambios e inventario para una VM única habilitada con Azure Arc mediante el portal.

Para habilitar Change Tracking e Inventario para una sola máquina virtual habilitada para Azure Arc mediante el portal, siga estos pasos:

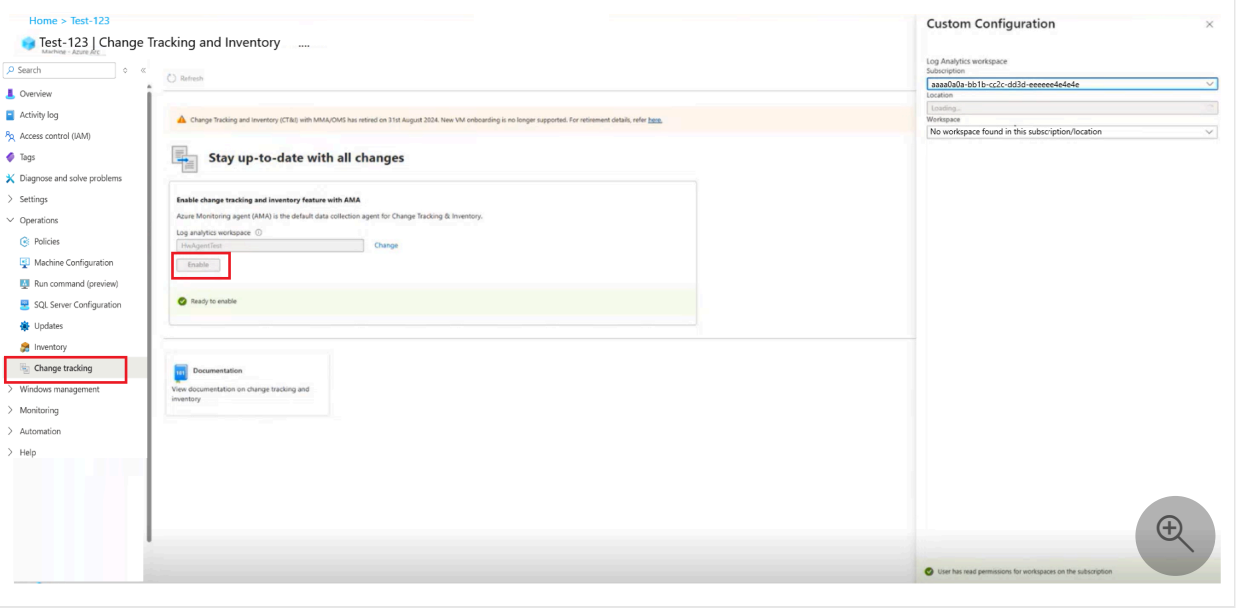
1. Inicie sesión en Azure Portal. Busque **Machines-Azure Arc**.



2. Seleccione la máquina de Azure Arc para la que desea habilitar Change Tracking.

3. En **Operaciones**, seleccione **Seguimiento de cambios** para ver el panel de **Seguimiento de cambios e Inventario**.

4. En el panel **Mantener up-to-date con todos los cambios** , seleccione la **opción Habilitar seguimiento de cambios e inventario con AMA** y seleccione **Habilitar**.



La implementación de Seguimiento de Cambios e Inventario comienza con una notificación en la esquina superior derecha del panel.

Contenido relacionado

- Para crear un DCR, consulte Creación de reglas de recopilación de datos.
- Para realizar un seguimiento de los cambios en Windows y Linux, consulte Matriz de compatibilidad y regiones.

Last updated on 05/06/2026

Tutorial: Cambio de un área de trabajo y configuración de reglas de recopilación de datos

Se aplica a: ✔ Máquinas virtuales de Windows ✔ Máquinas virtuales de Linux ✔ Registro de Windows ✔ Archivos de Windows ✔ Archivos de Linux ✔ Software de Windows

En este tutorial se describe cómo cambiar un área de trabajo y configurar reglas de recopilación de datos.

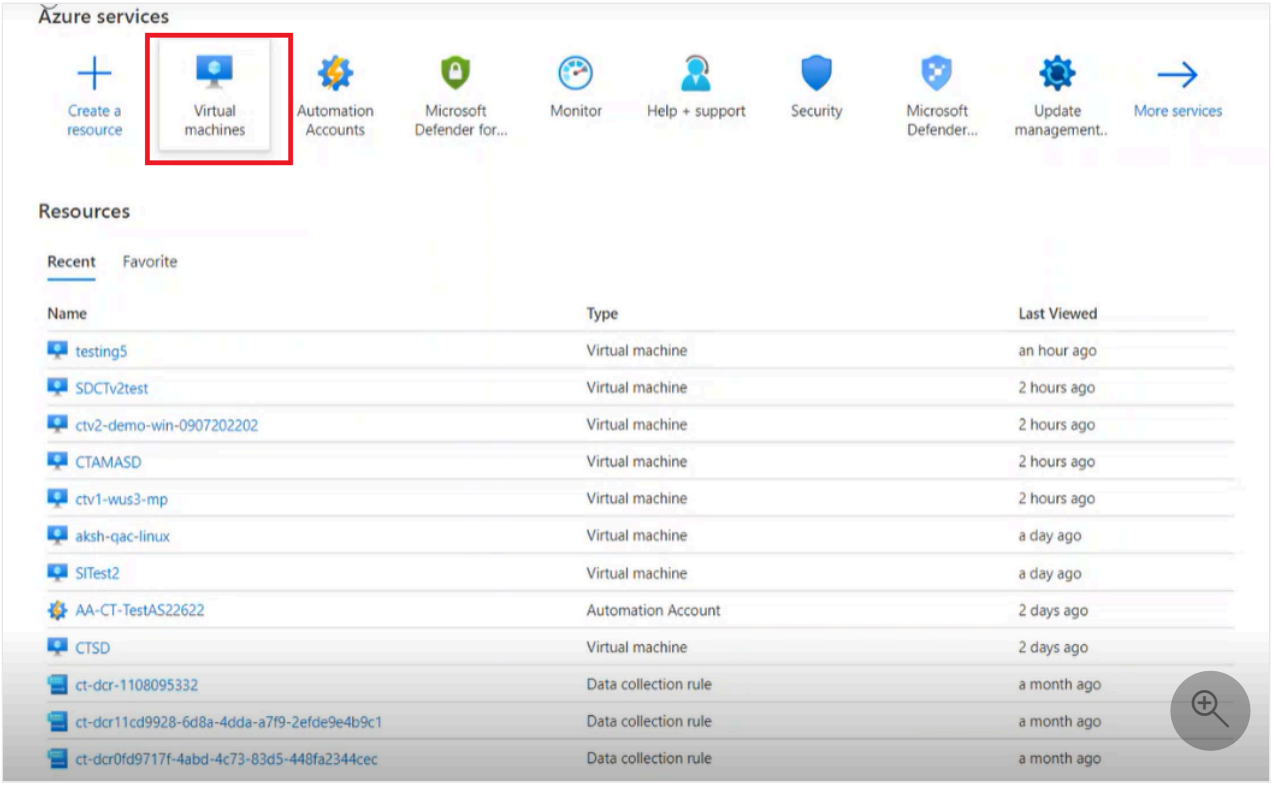
Prerrequisitos

Antes de cambiar un área de trabajo para la máquina virtual (VM) y configurar reglas de recopilación de datos (DCR), asegúrese de haber habilitado Azure Change Tracking e Inventory en la máquina virtual con un DCR. Para obtener información detallada sobre cómo crear un DCR, consulte Creación de un DCR.

Configuración de archivos de Windows, Linux y el registro de Windows mediante DCR

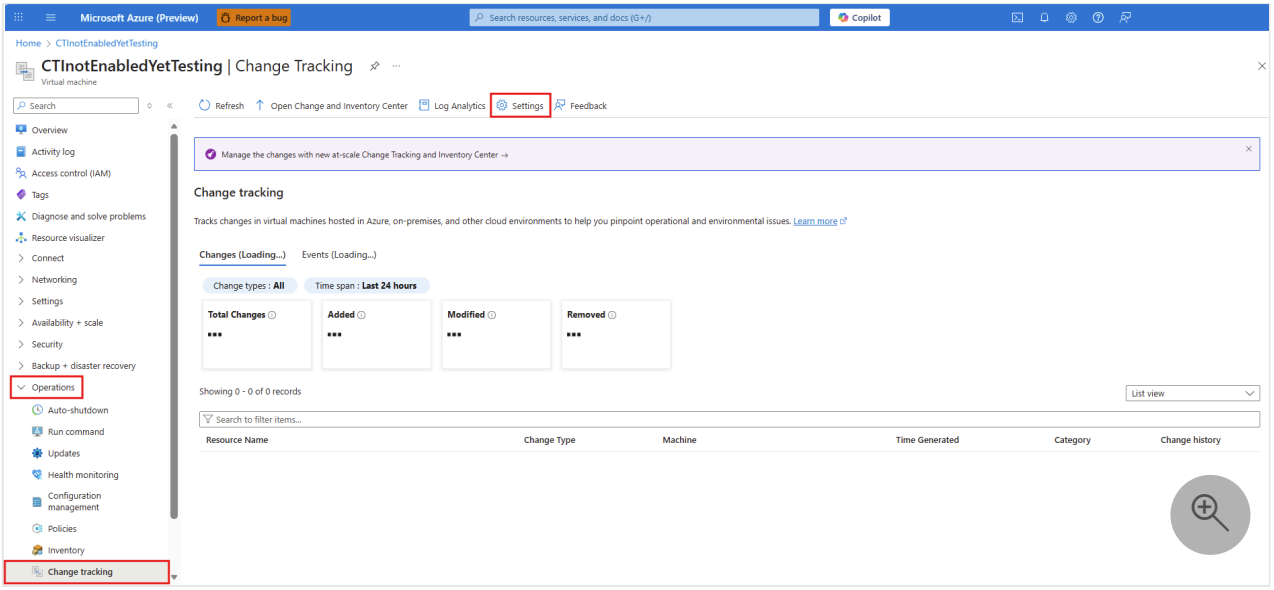
Para configurar archivos de Windows, Linux y el Registro de Windows mediante DCR, siga estos pasos:

1. Inicie sesión en Azure Portal y seleccione la máquina virtual.



2. Seleccione una máquina virtual específica para la que desea configurar las opciones **de Seguimiento de cambios** .

3. En **Operaciones**, seleccione **Seguimiento de** cambios para ver todos los cambios que tuvieron lugar en la máquina virtual.

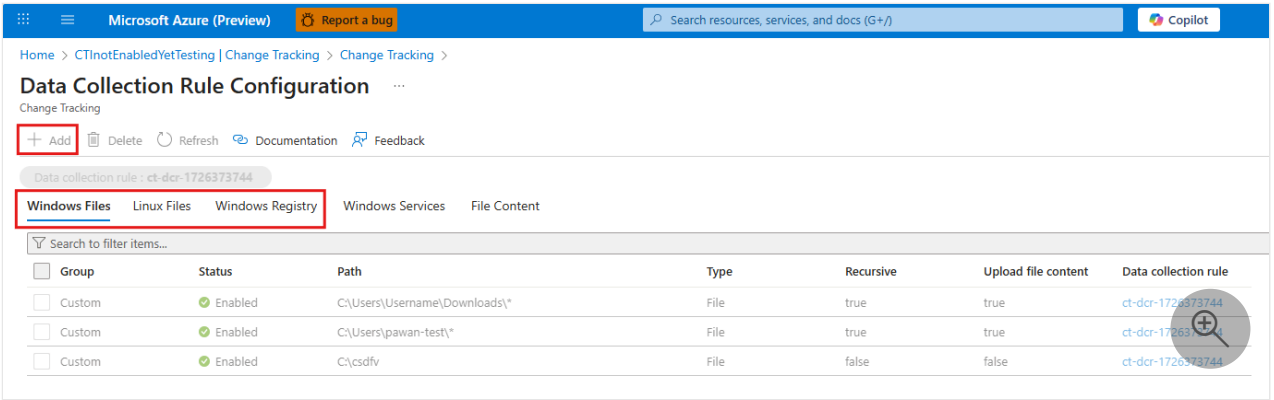


4. Seleccione **Configuración** para ver el panel **Configuración de reglas de recopilación de datos** . Aquí puede realizar las siguientes acciones:

   - Configurar cambios en una máquina virtual a nivel granular.
   - Seleccione el filtro para configurar el área de trabajo.
   - Usa el filtro para ver todas las reglas de recopilación de datos configuradas en el nivel específico del área de trabajo de Registros de Azure Monitor.

> **Nota:**
>
> La configuración que configure se aplica a todas las máquinas virtuales asociadas al DCR especificado. Para más información sobre DCR, consulte Reglas de recopilación de datos en Azure Monitor.

5. Seleccione + **Agregar** para configurar nuevos valores de archivo. Use el procedimiento tal como se especifica para los archivos Windows, Linux y el Registro de Windows.



Archivos de Windows

En la pestaña **Archivos de Windows**, seleccione + **Agregar**>**Agregar configuración de archivo de Windows**. Escriba la información del archivo o carpeta para realizar el seguimiento y seleccione **Agregar**. En la tabla siguiente se describen las propiedades que puede usar para escribir la información.

Expandir tabla

Propiedad	Description
Enabled	Verdadero si se aplica la configuración y falso en caso contrario.
Nombre del artículo	Nombre amigable del archivo al que se va a realizar el seguimiento.
Grupo	Un nombre de grupo para agrupar archivos lógicamente.
Ruta	La ruta para comprobar el archivo, por ejemplo, `c:\temp\*.txt`. También puede usar variables de entorno, como `%winDir%\System32\*.*`.
Tipo de ruta	El tipo de ruta. Los valores posibles son Archivo y Carpeta.
Recursividad	True si se usa la recursividad cuando se busca el elemento al que se va a realizar el seguimiento y false en caso contrario.

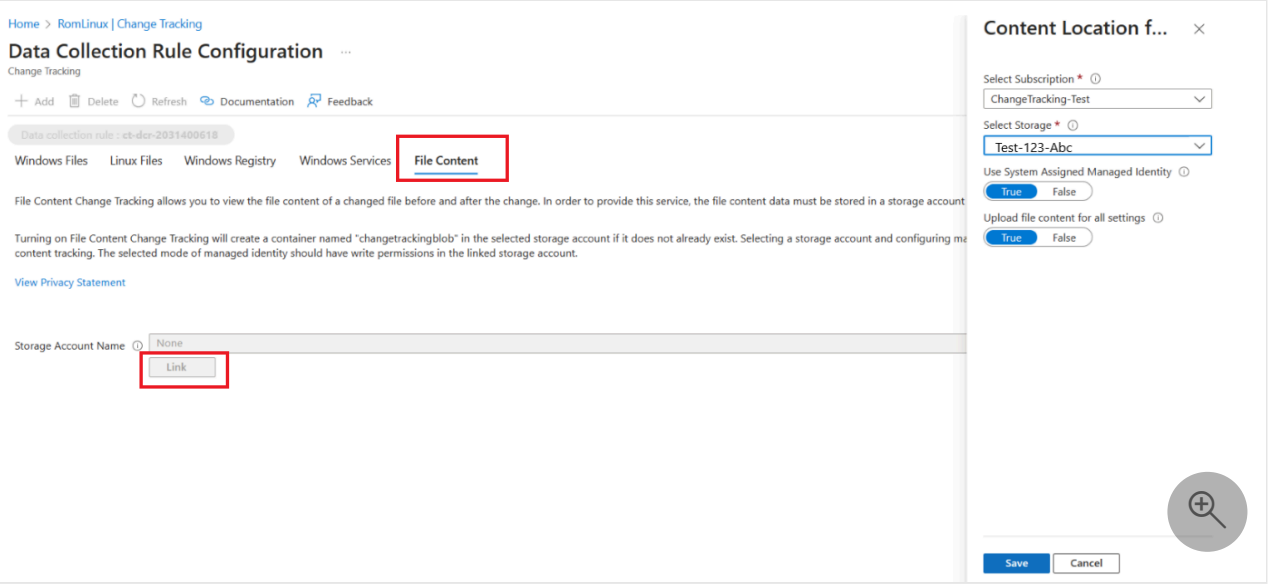
Ahora puede ver las máquinas virtuales configuradas en la DCR desde el panel **Reglas de recopilación de datos** en el portal de Azure.

Configuración de los cambios de contenido del archivo

Para configurar los cambios de contenido del archivo, siga estos pasos:

1. En la máquina virtual, en **Operaciones**, seleccione **Configuración de seguimiento de cambios**>.

2. En el panel **Configuración de regla de recopilación de datos**, seleccione **Vínculo de contenido**> del archivo para vincular la cuenta de almacenamiento.



3. En el panel **Ubicación de contenido de Change Tracking** , seleccione la suscripción y la cuenta de almacenamiento que desea usar para el seguimiento de cambios del contenido del archivo.

4. Confirme que usa la **identidad administrada asignada por el sistema**.

5. Seleccione **Cargar contenido de archivo para toda la configuración**. Seleccione **Guardar** para asegurarse de que se realiza un seguimiento de los cambios de contenido del archivo para todos los archivos que residen en este DCR.

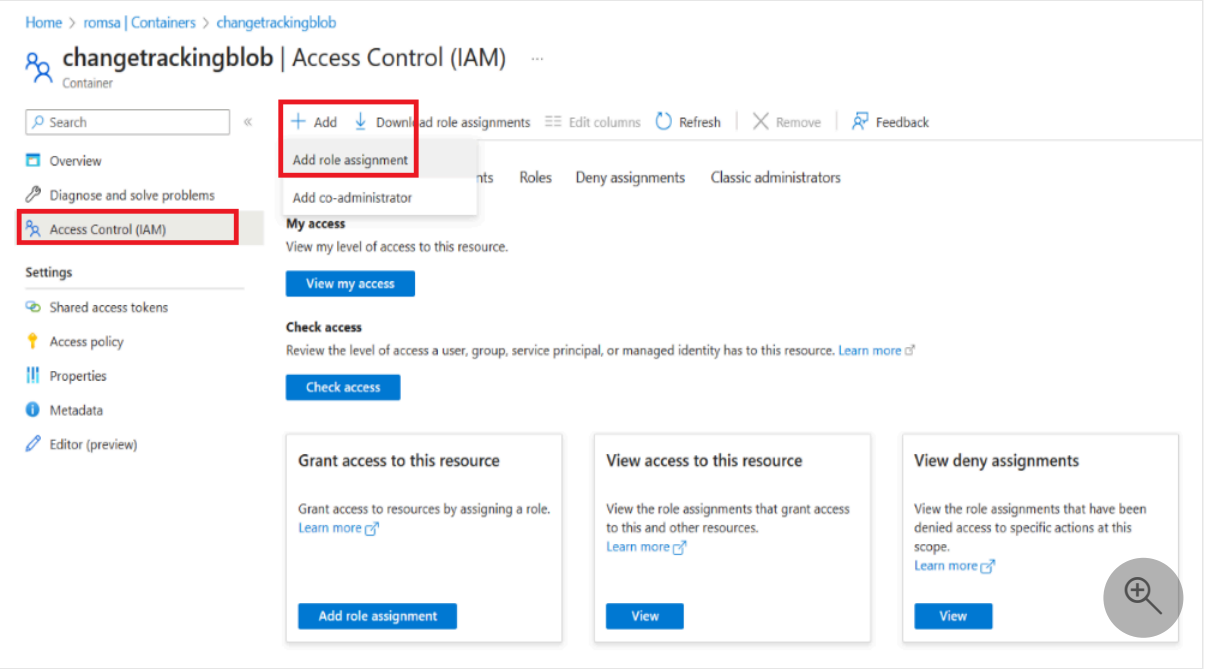
Identidad administrada asignada por el sistema

Cuando la cuenta de almacenamiento está vinculada mediante la identidad administrada asignada por el sistema, se crea un blob. Para la identidad administrada asignada por el sistema, siga estos pasos:

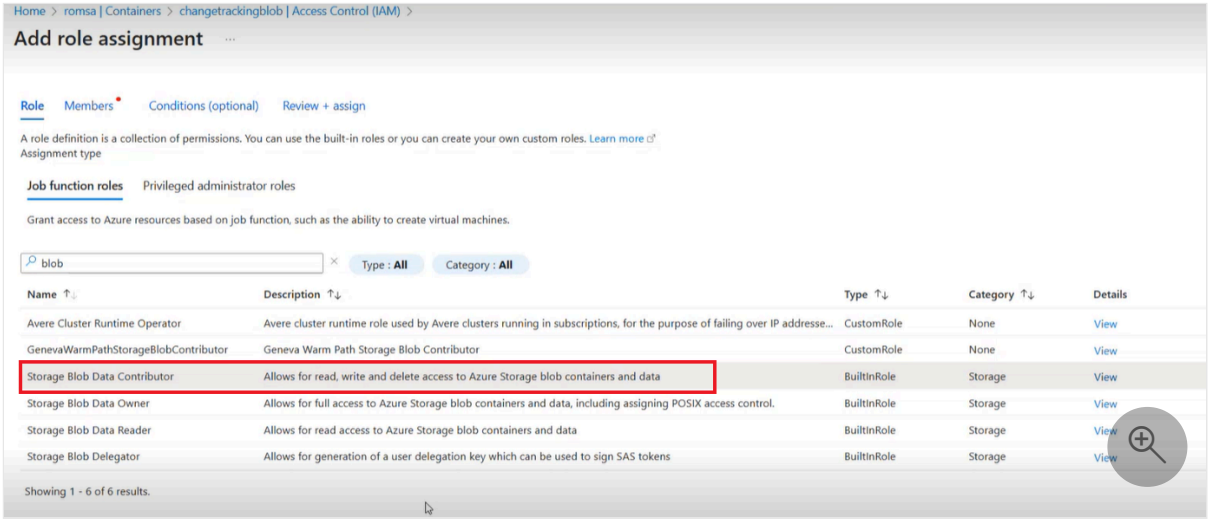
1. Inicie sesión en Azure Portal, vaya a **Cuentas de almacenamiento** y seleccione la cuenta de almacenamiento.

2. En el panel **Cuentas de almacenamiento** , en **Almacenamiento de datos**, seleccione **Contenedores**>**changetracking blob**>**Access Control (IAM)**.

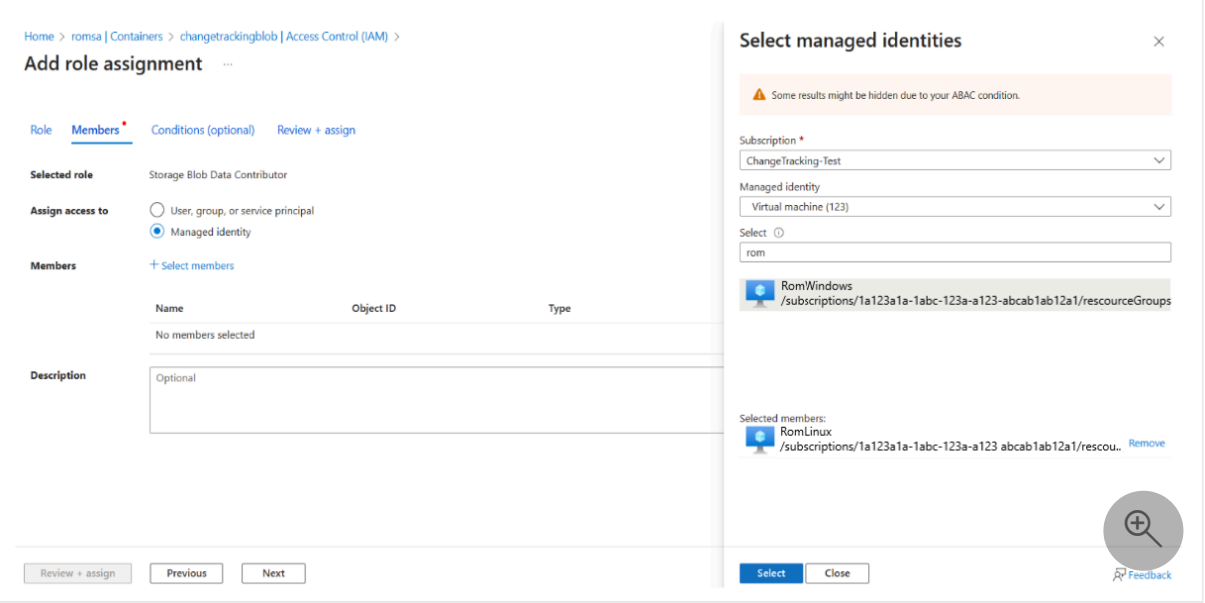
3. En **changetrackingblob | Panel Control de acceso (IAM),** seleccione **Agregar** y, después, **Agregar asignación de roles**.



4. En el panel **Agregar asignación de roles**, escriba **Colaborador de datos de blobs** en la búsqueda para asignar un rol de colaborador de Azure Blob Storage para la máquina virtual determinada. Este permiso proporciona acceso para leer, escribir y eliminar el acceso a los contenedores y datos de Blob Storage.



5. Seleccione el rol y asígnelo a la máquina virtual.



Actualización de la versión de extensión para Windows y Linux

> ⓘ **Nota:**
>
> Asegúrese de que la versión de extensión ChangeTracking-Linux/ChangeTracking-Windows se actualice a la versión actual de disponibilidad general (GA): versión de la extensión ga.

Use el siguiente comando para actualizar la versión de la extensión:

Azure CLI

```
az vm extension set -n {ExtensionName} --publisher
Microsoft.Azure.ChangeTrackingAndInventory --ids {VirtualMachineResourceId}
```

La extensión para Windows es `Vms - ChangeTracking-Windows`. La extensión para Linux es `Vms - ChangeTracking-Linux`.

Configuración mediante caracteres comodín

Para configurar la supervisión de archivos y carpetas mediante caracteres comodín, tenga en cuenta los siguientes puntos:

- Los caracteres comodín son necesarios para realizar el seguimiento de varios archivos.
- Los caracteres comodín solo se pueden usar en el último segmento de una ruta, como `C:\folder\file` o `/etc/.conf*`.

- Si una variable de entorno incluye una ruta de acceso que no es válida, la validación se realiza correctamente, pero se produce un error en la ruta de acceso cuando se ejecuta el inventario.
- Al establecer la ruta de acceso, evite rutas de acceso generales como `*C:\*`, lo que da lugar a que se atraviesan demasiadas carpetas.

Contenido relacionado

- Para habilitar Azure Change Tracking e Inventario desde Azure Portal, consulte Inicio rápido: Habilitación de Azure Change Tracking e Inventario.

Last updated on 10/01/2026

Guía de migración de Change Tracking e inventario mediante Log Analytics para Change Tracking e inventario mediante la versión del Agente de supervisión de Azure

Se aplica a: ✔ VM de Windows ✔ VM de Linux ✔ Servidores habilitados para Azure Arc.

En este artículo, se proporcionan instrucciones para pasar de Seguimiento de cambios e inventario mediante la versión de Log Analytics (LA) a la versión del agente de supervisión de Azure (AMA).

Con Azure Portal, puede migrar de Change Tracking e Inventory con el agente de LA a Change Tracking e Inventory con AMA y hay dos maneras de realizar esta migración:

- Migre máquinas únicas o varias desde la página Azure Virtual Machines o la página Machines-Azure Arc.
- Migre varias máquinas virtuales en la solución de versión de LA dentro de una cuenta de Automation determinada.

Además, puede usar un script para migrar todas las máquinas virtuales y las máquinas que no son de Azure habilitadas para Arc en un nivel de área de trabajo de Log Analytics desde la solución de versión de LA a Seguimiento de cambios e inventario e Inventario con AMA. Esto no es posible mediante la experiencia de Azure Portal mencionada anteriormente.

El script también permite migrar a la misma área de trabajo. Si va a migrar al mismo área de trabajo, el script quitará el agente LA(MMA/OMS) de las máquinas. Esto puede hacer que las otras soluciones dejen de funcionar. Por lo tanto, se recomienda planear la migración en consecuencia. Por ejemplo, migre primero otra solución y, a continuación, continúe con la migración de Change Tracking.

> ⓘ **Nota:**
>
> La eliminación no funciona en agentes de MMA que se instalaron mediante el instalador MSI. Sólo funciona en extensiones VM/Arc VM.

> ⓘ **Nota:**
>
> La supervisión de la integridad de los archivos (FIM) con Microsoft Defender para punto de conexión (MDE) ya está disponible actualmente. Siga las instrucciones para migrar desde:

- FIM con Seguimiento de cambios e inventario mediante AMA.
- FIM con Seguimiento de cambios e inventario mediante MMA.

Incorporación a Seguimiento de cambios e inventario mediante el agente de supervisión de Azure

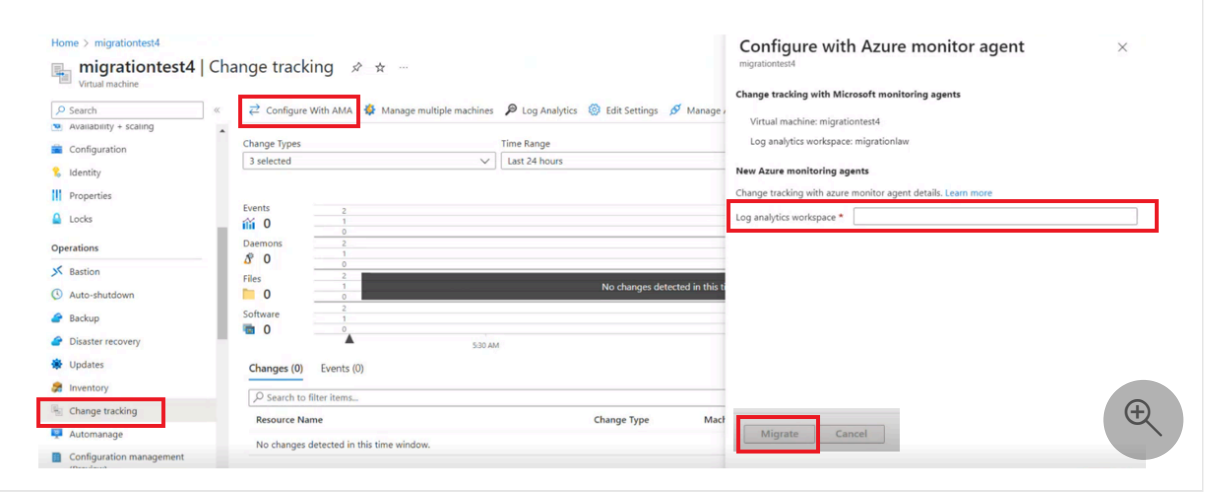
Máquina virtual de Azure única: Azure Portal

Para incorporar a través de Azure Portal, siga estos pasos:

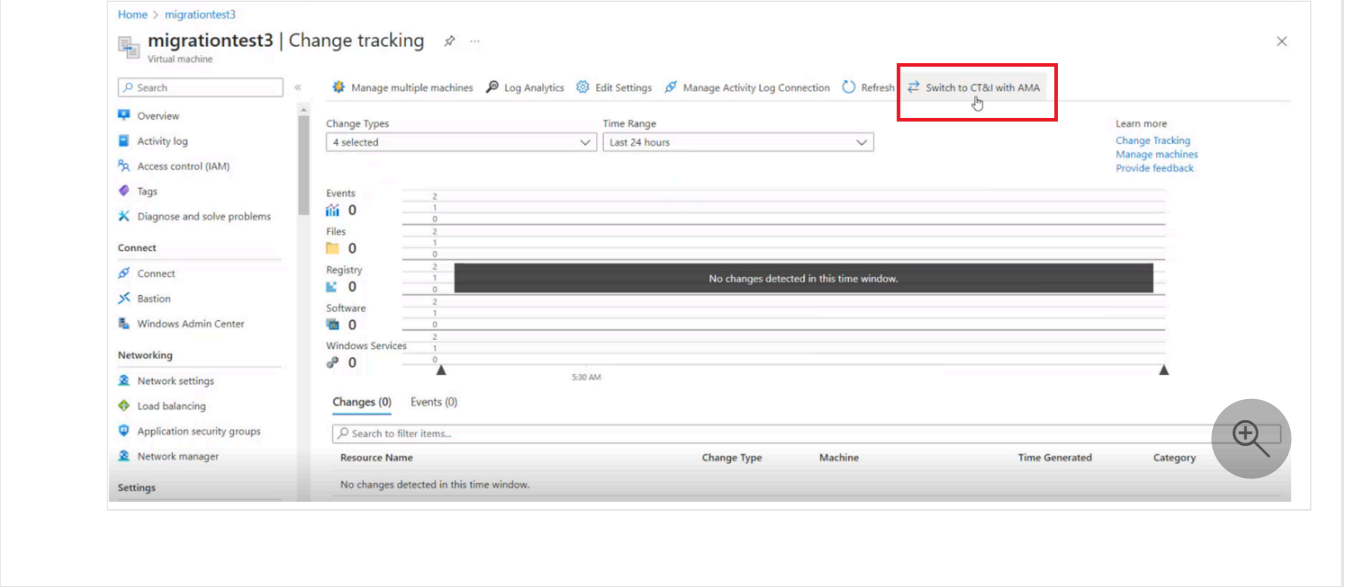
1. Inicie sesión en Azure Portal y seleccione la máquina virtual

2. En **Operaciones**, seleccione **Seguimiento de cambios**.

3. Seleccione **Configurar con AMA** y, en **Configurar con el agente de Azure Monitor**, proporcione el **área de trabajo de Log Analytics** y seleccione **Migrar** para iniciar la implementación.

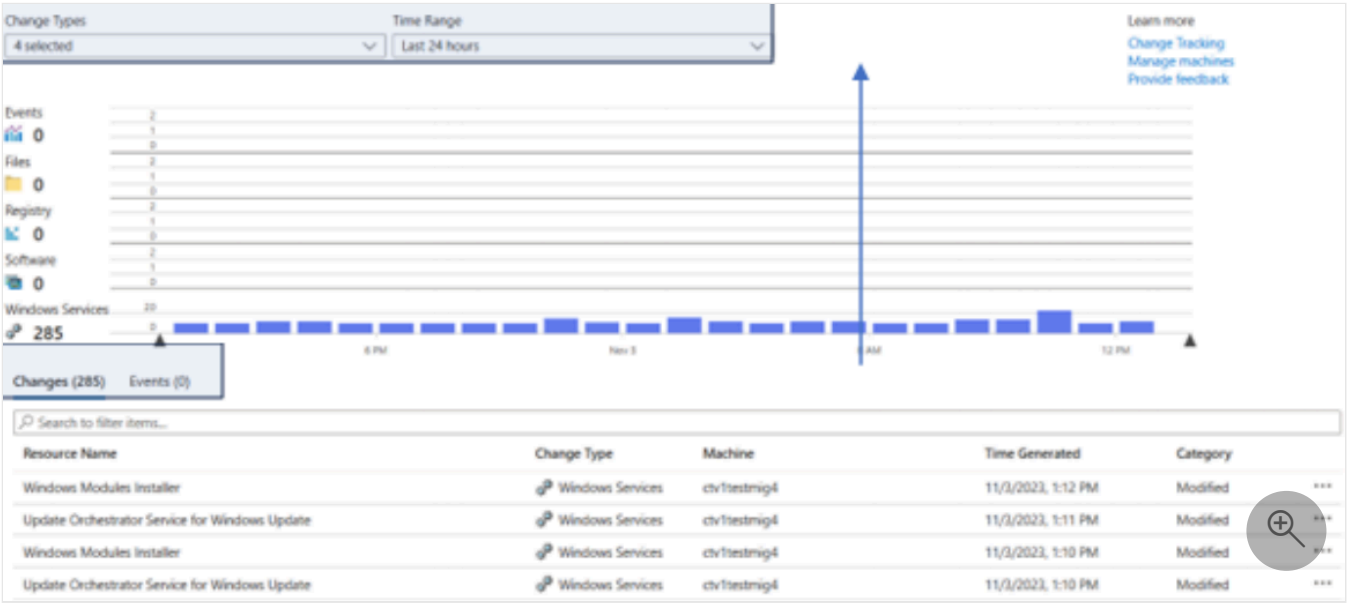


4. Seleccione **Cambiar a CT&I con AMA** para evaluar los eventos y registros entrantes en el agente de LA y la versión AMA.



Comparación de datos entre el agente de Log Analytics y la versión del agente de supervisión de Azure

Después de completar la incorporación a Change Tracking con la versión de AMA, seleccione **Cambiar a CT con AMA** en la página de aterrizaje para cambiar entre las dos versiones y comparar los siguientes eventos.



Por ejemplo, si la incorporación a la versión de AMA del servicio tuviera lugar después del 3 de noviembre a las 6:00 a.m. Es posible comparar los datos manteniendo filtros coherentes entre parámetros como **Tipos de cambio** e **Intervalo de tiempo**. Es posible comparar los registros entrantes en la sección **Cambios** y en la sección gráfica para garantizar la coherencia de los datos.

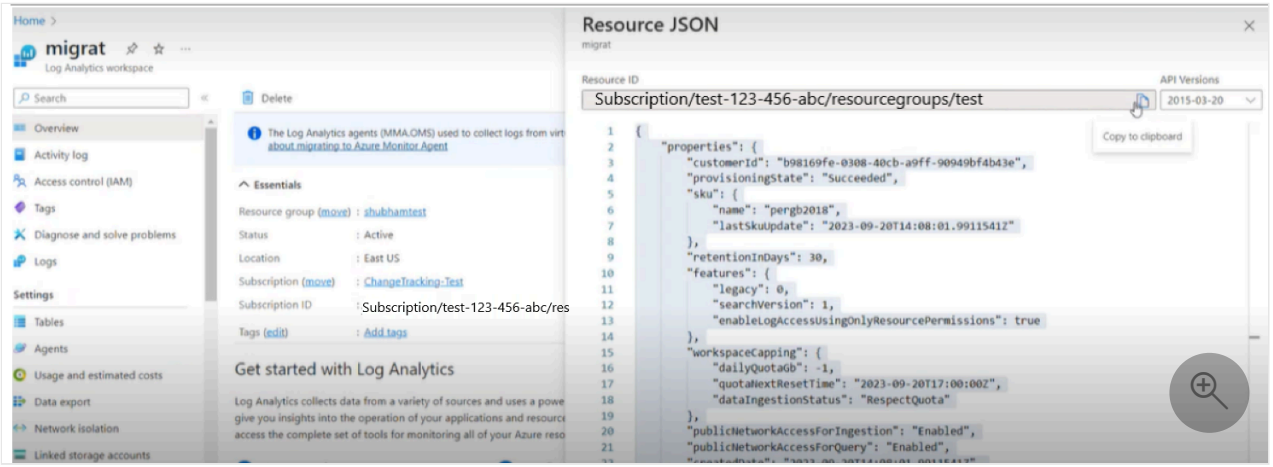
> ⓘ **Nota:**

Es necesario comparar los datos y registros entrantes una vez finalizada la incorporación a la versión de AMA.

Obtención del id. de recurso del área de trabajo de Log Analytics

Para obtener el id. de recurso del área de trabajo de Log Analytics, siga estos pasos:

1. Inicie sesión en Azure Portal
2. En **Área de trabajo de Log Analytics**, seleccione el área de trabajo específica y seleccione **Vista Json**.
3. Copie el **id. de recurso**.



Limitaciones

Mediante Azure Portal

Para una sola máquina virtual y la cuenta de Automation

1. Para la configuración basada en cambios de contenido de archivos, es necesario migrar manualmente de la versión de LA a la versión de AMA de Seguimiento de cambios e inventario. Siga las instrucciones que aparecen en Seguimiento del contenido del archivo.
2. Si se realiza la migración a un área de trabajo diferente, las alertas configuradas mediante el área de trabajo de Log Analytics deben configurar manualmente.

Deshabilitación de Change Tracking mediante el agente de Log Analytics

Después de habilitar la administración de máquinas virtuales mediante Seguimiento de cambios e inventario a través del agente de supervisión de Azure, podría decidir dejar de usar Seguimiento de cambios e inventario con la versión del agente de LA y quitar la configuración de la cuenta.

El método de deshabilitación incorpora lo siguiente:

- Quita el seguimiento de cambios con el agente de LA para algunas máquinas virtuales seleccionadas en el área de trabajo de Log Analytics.
- Quita el seguimiento de cambios con el agente de LA de toda el área de trabajo de Log Analytics.

Pasos siguientes

- Para habilitar la característica desde Azure Portal, consulte Habilitación de Seguimiento de cambios e inventario desde Azure Portal.

Last updated on 17/11/2025

Envío de un correo electrónico desde un runbook de Azure Automation

Puede enviar un correo electrónico desde un runbook con SendGrid mediante PowerShell.

Si no tiene una suscripción a Azure, cree una cuenta gratuita antes de empezar.

Requisitos previos

- Una cuenta de SendGrid.
- Comprobación del remitente de SendGrid. Dominio o remitente único
- La clave de API de SendGrid.
- Una cuenta de Azure Automation con al menos una identidad administrada asignada por el usuario. Para más información, consulte Habilitación de identidades administradas.
- Módulos de az: `Az.Accounts` y `Az.KeyVault` se han importado en la cuenta de Automation. Para más información, consulte Importación de módulos Az.
- El módulo Azure Az de PowerShell instalado en la máquina. Para instalar o actualizar, vea Instalación del módulo Azure Az de PowerShell.

Crear una instancia de Azure Key Vault

Cree una instancia de Azure Key Vault y una directiva de acceso de Key Vault que permita a la credencial obtener y establecer los secretos del almacén de claves especificado.

1. Inicie sesión en Azure de forma interactiva con el cmdlet Connect-AzAccount y siga las instrucciones.

PowerShell

```
# Sign in to your Azure subscription
$sub = Get-AzSubscription -ErrorAction SilentlyContinue
if(-not($sub))
{
    Connect-AzAccount
}

# If you have multiple subscriptions, set the one to use
# Select-AzSubscription -SubscriptionId <SUBSCRIPTIONID>
```

2. Proporcione un valor adecuado para las variables siguientes y, a continuación, ejecute el script.

PowerShell

```
$resourceGroup = "<Resource group>"
$automationAccount = "<Automation account>"
$region = "<Region>"
$SendGridAPIKey = "<SendGrid API key>"
$VaultName = "<A universally unique vault name>"

$userAssignedManagedIdentity = "<User-assigned managed identity>"
```

3. Creación del almacén de claves y asignación de permisos

PowerShell

```
# Create the new key vault
$newKeyVault = New-AzKeyVault `
    -VaultName $VaultName `
    -ResourceGroupName $resourceGroup `
    -Location $region

$resourceId = $newKeyVault.ResourceId

# Convert the SendGrid API key into a SecureString
$Secret = ConvertTo-SecureString -String $SendGridAPIKey `
    -AsPlainText -Force

Set-AzKeyVaultSecret -VaultName $VaultName `
    -Name 'SendGridAPIKey' `
    -SecretValue $Secret

# Grant Key Vault access to the Automation account's system-assigned managed
identity.
$SA_PrincipalId = (Get-AzAutomationAccount `
    -ResourceGroupName $resourceGroup `
    -Name $automationAccount).Identity.PrincipalId

Set-AzKeyVaultAccessPolicy `
    -VaultName $vaultName `
    -ObjectId $SA_PrincipalId `
    -PermissionsToSecrets Set, Get

# Grant Key Vault access to the user-assigned managed identity.
$UAMI = Get-AzUserAssignedIdentity `
    -ResourceGroupName $resourceGroup `
    -Name $userAssignedManagedIdentity

Set-AzKeyVaultAccessPolicy `
    -VaultName $vaultName `
```

```
    -ObjectId $UAMI.PrincipalId `
    -PermissionsToSecrets Set, Get
```

Para ver otras formas de crear un almacén de Azure Key Vault y almacenar un secreto, consulte los inicios rápidos de Key Vault.

Asignación de permisos a las identidades administradas

Asigne permisos a la identidad administrada adecuada. El runbook puede usar la identidad administrada asignada por el sistema de la cuenta de Automation o una identidad administrada asignada por el usuario. Se proporcionan los pasos necesarios para asignar permisos a cada identidad. En los pasos siguientes se usa PowerShell. Si prefiere utilizar Azure Portal, consulte Asignación de roles de Azure mediante Azure Portal.

1. Use el cmdlet de PowerShell New-AzRoleAssignment para asignar un rol a la identidad administrada asignada por el sistema.

   PowerShell
   ```
   New-AzRoleAssignment `
       -ObjectId $SA_PrincipalId `
       -ResourceGroupName $resourceGroup `
       -RoleDefinitionName "Reader"
   ```

2. Asigne un rol a una identidad administrada asignada por el usuario.

   PowerShell
   ```
   New-AzRoleAssignment `
       -ObjectId $UAMI.PrincipalId`
       -ResourceGroupName $resourceGroup `
       -RoleDefinitionName "Reader"
   ```

3. Para la identidad administrada asignada por el sistema, muestre `ClientId` y registre el valor para su uso posterior.

   PowerShell
   ```
   $UAMI.ClientId
   ```

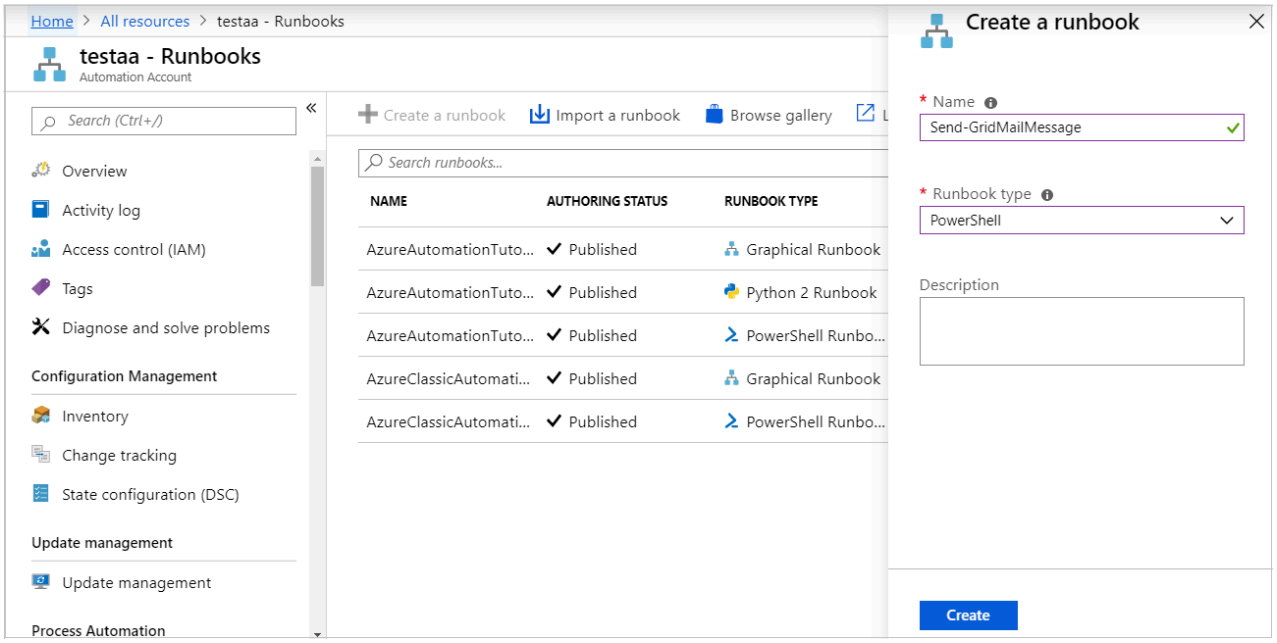
Creación del runbook para enviar un correo electrónico

Después de crear un almacén de claves y de almacenar la clave de API de `SendGrid`, es el momento de crear el runbook que recupere la clave de API y envíe un correo electrónico. Vamos a usar un runbook que utiliza la identidad administrada asignada por el sistema para autenticarse en Azure con el fin de recuperar el secreto de Azure Key Vault. Llamaremos al runbook **Send-GridMailMessage**. Puede modificar el script de PowerShell utilizado para distintos escenarios.

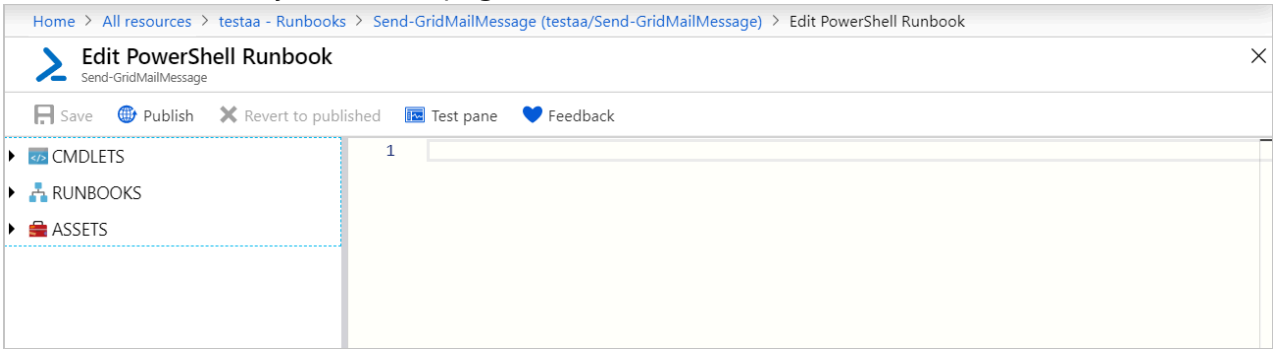
1. Inicie sesión en Azure Portal y vaya a su cuenta de Automation.

2. En la página de la cuenta de Automation abierta, en **Automatización de procesos**, seleccione **Runbooks**.

3. Seleccione **+ Crear un runbook**.
   a. Asigne al runbook el nombre `Send-GridMailMessage`.
   b. En la lista desplegable **Tipo de Runbook**, seleccione **PowerShell**.
   c. Seleccione **Crear**.



4. Se crea el runbook y se abre la página Editar Runbook de PowerShell.



5. Copie el siguiente ejemplo de PowerShell en la página Editar. Asegúrese de que `VaultName` especifique el nombre que haya elegido para la instancia de Key Vault.

PowerShell

```
Param(
  [Parameter(Mandatory=$True)]
  [String] $destEmailAddress,
  [Parameter(Mandatory=$True)]
  [String] $fromEmailAddress,
  [Parameter(Mandatory=$True)]
  [String] $subject,
  [Parameter(Mandatory=$True)]
  [String] $content,
  [Parameter(Mandatory=$True)]
  [String] $ResourceGroupName
)

# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process

# Connect to Azure with system-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription -
DefaultProfile $AzureContext

$VaultName = "<Enter your vault name>"

$SENDGRID_API_KEY = Get-AzKeyVaultSecret `
    -VaultName $VaultName `
    -Name "SendGridAPIKey" `
    -AsPlainText -DefaultProfile $AzureContext

$headers = New-Object "System.Collections.Generic.Dictionary[[String],
[String]]"
$headers.Add("Authorization", "Bearer " + $SENDGRID_API_KEY)
$headers.Add("Content-Type", "application/json")

$body = @{
personalizations = @(
    @{
        to = @(
                @{
                    email = $destEmailAddress
                }
        )
    }
)
from = @{
    email = $fromEmailAddress
}
subject = $subject
```

```
content = @(
    @{
        type = "text/plain"
        value = $content
    }
)
}

$bodyJson = $body | ConvertTo-Json -Depth 4

$response = Invoke-RestMethod -Uri https://api.sendgrid.com/v3/mail/send -
Method Post -Headers $headers -Body $bodyJson
```

6. Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:
   a. En la línea 18, elimine `$AzureContext = (Connect-AzAccount -Identity).context`.
   b. Reemplace el valor por `$AzureContext = (Connect-AzAccount -Identity -AccountId <ClientId>).context`.
   c. Escriba el identificador de cliente que obtuvo anteriormente.

7. Seleccione **Guardar**, luego **Publicar** y, finalmente, **Sí** cuando se le pida confirmación.

Para comprobar que el runbook se ejecuta correctamente, puede seguir los pasos descritos en Prueba de un runbook o de Inicio de un runbook.

Si no ve inicialmente el correo electrónico de prueba, compruebe las carpetas de **correo no deseado** y de **spam**.

Limpieza de recursos

1. Cuando ya no sea necesario el runbook, selecciónelo en la lista de runbooks y seleccione **Eliminar**.

2. Elimine el almacén de claves mediante el cmdlet Remove-AzKeyVault.

   PowerShell

```
$VaultName = "<your KeyVault name>"
$resourceGroup = "<your ResourceGroup name>"
Remove-AzKeyVault -VaultName $VaultName -ResourceGroupName $resourceGroup
```

Pasos siguientes

- Para enviar datos de trabajo de runbook al área de trabajo de Log Analytics, consulte Reenvío de datos de un trabajo de Azure Automation a registros de Azure Monitor.
- Para supervisar los registros y las métricas de nivel básico, consulte Uso de una alerta para desencadenar un runbook de Azure Automation.

Last updated on 17/11/2025

Supervisión de runbooks con alertas de métricas

Artículo • 01/06/2023

En este artículo, aprenderá a crear una alerta de métrica basada en el estado de finalización del runbook.

Inicio de sesión en Azure

Inicie sesión en el Portal de Azure

Crear una alerta

Las alertas le permiten definir una condición para supervisar y una acción para realizar cuando se cumple esa condición.

1. Inicie el servicio Azure Automation en Azure Portal. Para ello, haga clic en **Todos los servicios** y, luego, busque y seleccione **Cuentas de Automation**.

2. En la lista de cuentas de Automation, seleccione la cuenta para la que quiere crear una alerta.

3. En **Monitoring** (Supervisión), seleccione **Alerts** (Alertas) y, luego, elija **+ New Alert Rule** (+ Nueva regla de alertas). El ámbito del destino ya está definido y asociado con la cuenta de Automation.

Configuración de los criterios de alerta

1. Haga clic en **Select Condition** (Seleccionar condición). Seleccione **Metrics** (Métricas) en **Signal type** (Tipo de señal) y elija **Total Jobs** (Total de trabajos) en la lista.

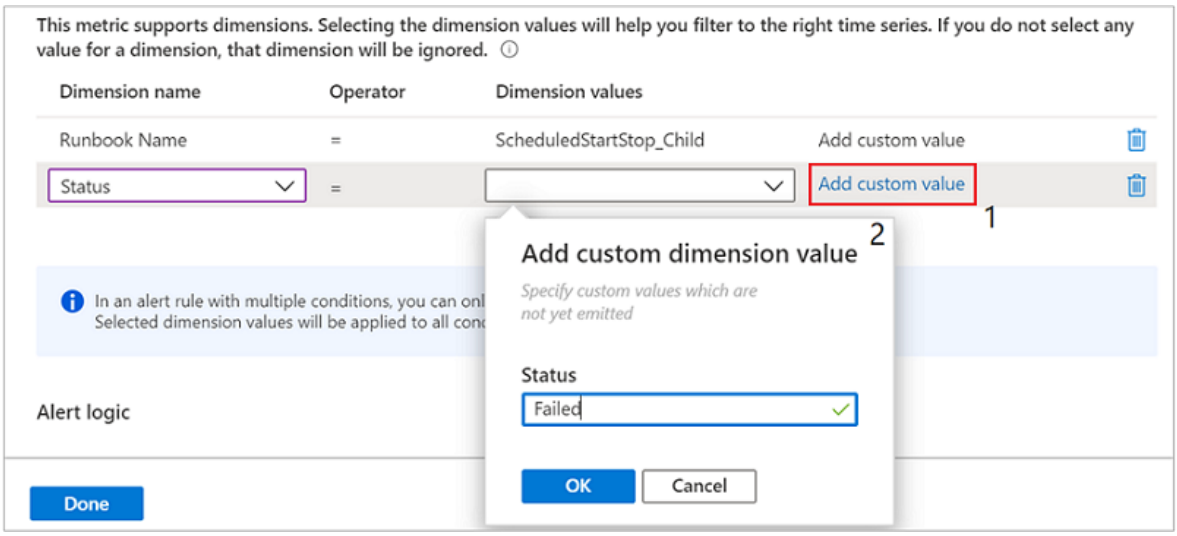
2. La página **Configurar lógica de señal** es donde se define la lógica que desencadena la alerta. En el gráfico del historial se presentan dos dimensiones, **Nombre de Runbook** y **Estado**. Las dimensiones son propiedades diferentes de una métrica que puede usarse para filtrar los resultados. En **Nombre de Runbook**, seleccione el runbook sobre el que quiera enviar la alerta o déjelo en blanco enviar alertas de todos los runbooks. En **Estado**, seleccione un estado de la lista desplegable que quiera supervisar. Los valores de nombre y estado del runbook

que aparecen en la lista desplegable son solo para los trabajos que se ejecutaron en la última semana.

Si quiere enviar una alerta sobre un estado o un runbook que no se muestran en la lista desplegable, haga clic en **Add custom value** (Agregar valor personalizado) junto a la dimensión. Esta acción abre un cuadro de diálogo que le permite especificar un valor personalizado que no se haya emitido recientemente para esa dimensión. Si escribe un valor que no existe para una propiedad, no se activará la alerta. Para obtener más información, consulte Estado del trabajo.

> **Nota**
>
> Si no especifica un nombre para la dimensión **Runbook Name** (Nombre del runbook) y existen runbooks que cumplen los criterios de estado, incluidos los runbooks ocultos del sistema, recibirá una alerta.

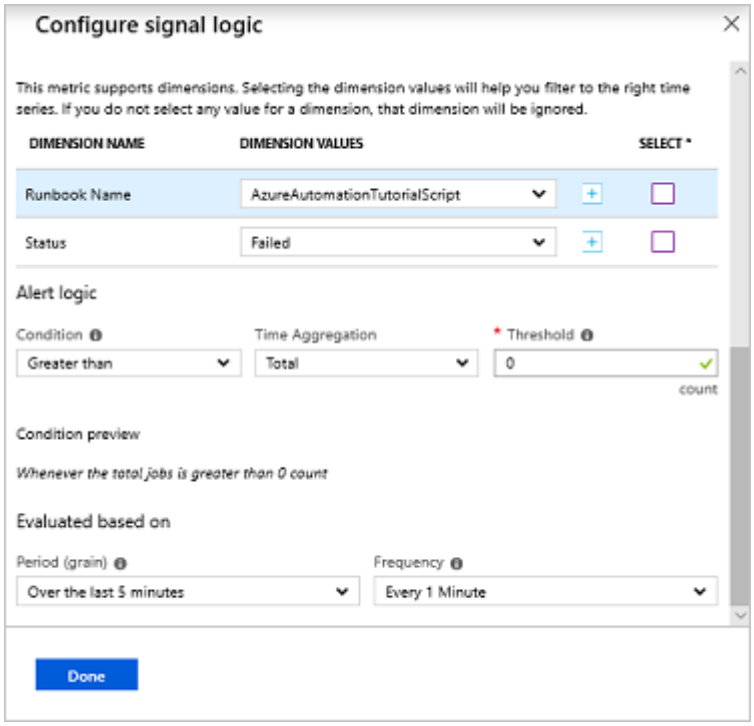
Por ejemplo, para generar una alerta cuando un runbook devuelva un estado *Failed* (Error), además de especificar el nombre del runbook, en la dimensión **Status** (Estado), agregue el valor de dimensión personalizado **Failed** (Error).



3. En **Lógica de alerta**, defina la condición y el umbral de la alerta. Debajo se muestra una vista previa de la condición definida.

4. En **Evaluated based on** (Se evalúa según), seleccione el intervalo de tiempo de la consulta y la frecuencia con la que quiere que se ejecute esa consulta. Por ejemplo, si elige **Over the last 5 minutes** (En los últimos 5 minutos) en **Period** (Período) y **Every 1 Minute** (Cada minuto) en **Frequency** (Frecuencia), la alerta busca el número de runbooks que cumplen los criterios en los últimos 5 minutos. Esta consulta se ejecuta cada minuto y, una vez que los criterios de alerta que ha

definido ya no se encuentran en un periodo de 5 minutos, la alerta se resuelve. Cuando termine, haga clic en **Listo**.



Definición de la acción que se realizará

1. En **Action group** (Grupo de acciones), seleccione **Specify action group** (Especificar grupo de acciones). Un grupo de acciones es un conjunto de acciones que puede usar en varias alertas. Por ejemplo, notificaciones por correo electrónico, runbooks, webhooks y muchas más. Para más información sobre los grupos de acciones y los pasos para crear uno que envíe una notificación por correo electrónico, consulte Creación y administración de grupos de acciones.

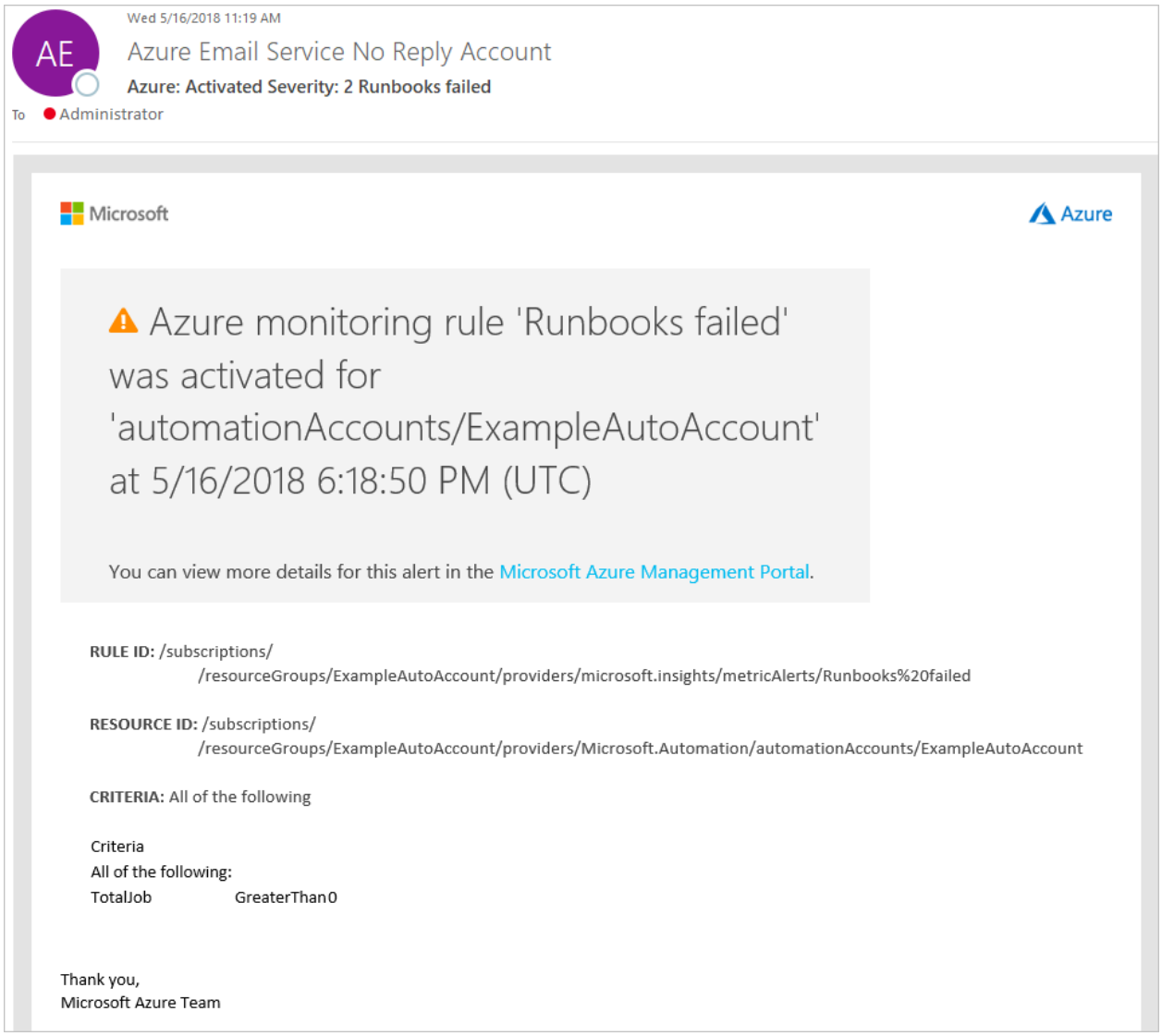
Definición de detalles de la alerta

1. En **Alert rule details** (Detalles de la regla de alertas), asigne a la alerta un nombre descriptivo y proporcione una descripción. Establezca la **Gravedad** para que coincida con la condición de la alerta. Hay cinco niveles de gravedad que van de 0 a 5. Las alertas se tratan de igual manera, independientemente de la gravedad; puede hacer coincidir la gravedad con la lógica de negocios.

2. De forma predeterminada, las reglas se habilitan durante la creación, a menos que seleccione **No** para la opción **Enable alert rule upon creation** (Habilitar regla de alertas tras la creación). En el caso de las alertas creadas en un estado deshabilitado, puede habilitarlas en el futuro cuando esté listo. Seleccione **Create alert rule** (Crear regla de alertas) para guardar los cambios.

Recepción de una notificación

Cuando se cumplen los criterios de la alerta, el grupo de acciones ejecuta la acción definida. En el ejemplo de este artículo, se envía un correo electrónico. La imagen siguiente es un ejemplo de un correo electrónico que recibirá después de que se desencadene la alerta:



Una vez que la métrica ya no esté fuera del umbral definido, se desactivará la alerta y el grupo de acciones ejecutará la acción definida. Si se selecciona un tipo de acción de correo electrónico, se envía un correo electrónico de resolución que indica que esa acción se ha resuelto.

Pasos siguientes

- Para más información, consulte Uso de una alerta para desencadenar un runbook de Azure Automation.

Usar una alerta para desencadenar un runbook de Azure Automation

Puede usar Azure Monitor para supervisar métricas de nivel básico y registros para la mayoría de los servicios de Azure. Puede llamar a runbooks de Azure Automation mediante grupos de acciones para automatizar tareas basadas en alertas. En este artículo se muestra cómo configurar y ejecutar un runbook mediante alertas.

Requisitos previos

- Una cuenta de Azure Automation con al menos una identidad administrada asignada por el usuario. Para más información, consulte Uso de una identidad administrada asignada por el usuario para una cuenta de Azure Automation.
- Módulos de az: `Az.Accounts` y `Az.Compute` se han importado en la cuenta de Automation. Para más información, consulte Importación de módulos Az.
- Una máquina virtual de Azure.
- El módulo Azure Az de PowerShell instalado en la máquina. Para instalar o actualizar, vea Instalación del módulo Azure Az de PowerShell.
- Familiaridad general con los runbooks de Automation.

Tipos de alerta

Puede usar runbooks de Automation con tres tipos de alerta:

- Alertas comunes
- Alertas de registros de actividad
- Alertas de métricas casi en tiempo real

> **Nota:**
>
> El esquema de alerta común normaliza la experiencia de consumo de notificaciones de alerta en Azure. Históricamente, los tres tipos de alerta de Azure (métrica, registro y registro de actividad) han tenido sus propias plantillas de correo electrónico, esquemas de webhook, etc. Para más información, consulte Esquema de alertas comunes.

Cuando una alerta llama a un runbook, la llamada real es una solicitud HTTP POST al webhook. El cuerpo de la solicitud POST contiene un objeto con formato JSON que tiene propiedades útiles relacionadas con la alerta. En la tabla siguiente se enumeran vínculos al esquema de carga de cada tipo de alerta:

Expandir tabla

Alerta	Descripción	Esquema de carga
Alerta común	Esquema de alerta común que normaliza la experiencia de consumo de notificaciones de alerta en Azure en la actualidad.	Esquema de carga de alertas comunes.
Alerta de registro de actividad	Envía una notificación cuando cualquier evento nuevo del registro de actividad de Azure coincide con las condiciones específicas. Por ejemplo, cuando una operación `Delete VM` se produce en **myProductionResourceGroup** o cuando aparece un nuevo evento de Azure Service Health con un estado Activo.	Esquema de carga de alertas de registros de actividad
Alertas de métricas casi en tiempo real	Envía una notificación con más rapidez que las alertas de métrica cuando una o varias métricas de nivel de plataforma cumplen las condiciones especificadas. Por ejemplo, cuando el valor de **% de CPU** en una VM es mayor que 90 y el valor de **Entrada de red** es mayor que 500 MB durante los últimos cinco minutos.	Esquema de carga de alertas de métricas casi en tiempo real

Dado que los datos que cada tipo de alerta proporciona son diferentes, cada tipo de alerta se administra de manera diferente. En la siguiente sección, aprenderá a crear un runbook para administrar distintos tipos de alertas.

Asignación de permisos a las identidades administradas

Asigne permisos a la identidad administrada apropiada para permitirle detener una máquina virtual. El runbook puede usar la identidad administrada asignada por el sistema de la cuenta de Automation o una identidad administrada asignada por el usuario. Se proporcionan los pasos necesarios para asignar permisos a cada identidad. En los pasos siguientes se usa PowerShell. Si prefiere utilizar Azure Portal, consulte Asignación de roles de Azure mediante Azure Portal.

1. Inicie sesión en Azure de forma interactiva con el cmdlet Connect-AzAccount y siga las instrucciones.

PowerShell

```
# Sign in to your Azure subscription
$sub = Get-AzSubscription -ErrorAction SilentlyContinue
if(-not($sub))
{
    Connect-AzAccount
}
```

```
# If you have multiple subscriptions, set the one to use
# Select-AzSubscription -SubscriptionId <SUBSCRIPTIONID>
```

2. Proporcione un valor adecuado para las variables siguientes y, a continuación, ejecute el script.

PowerShell

```
$resourceGroup = "resourceGroup"
$automationAccount = "AutomationAccount"
$userAssignedManagedIdentity = "userAssignedManagedIdentity"
```

3. Use el cmdlet de PowerShell New-AzRoleAssignment para asignar un rol a la identidad administrada asignada por el sistema.

PowerShell

```
$SAMI = (Get-AzAutomationAccount -ResourceGroupName $resourceGroup -Name
$automationAccount).Identity.PrincipalId
New-AzRoleAssignment `
    -ObjectId $SAMI `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName "DevTest Labs User"
```

4. Asigne un rol a una identidad administrada asignada por el usuario.

PowerShell

```
$UAMI = (Get-AzUserAssignedIdentity -ResourceGroupName $resourceGroup -Name
$userAssignedManagedIdentity)
New-AzRoleAssignment `
    -ObjectId $UAMI.PrincipalId `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName "DevTest Labs User"
```

5. Para la identidad administrada asignada por el sistema, muestre `ClientId` y registre el valor para su uso posterior.

PowerShell

```
$UAMI.ClientId
```

Creación de un runbook para administrar las alertas

Para usar Automation con alertas, necesita un runbook que administre la carga de JSON de alerta que se pasa al runbook. El siguiente runbook de ejemplo se debe llamar desde una alerta de Azure.

Como se ha descrito en la sección anterior, cada tipo de alerta tiene un esquema diferente. El script toma los datos del webhook de una alerta del parámetro de entrada del runbook `WebhookData`. Después, el script evalúa la carga de JSON para determinar qué tipo de alerta se usa.

En este ejemplo se utiliza una alerta de una máquina virtual (VM) de Azure. Extrae los datos de la máquina virtual de la carga, específicamente del recurso de destino de la alerta desencadenada y, a continuación, utiliza esa información para detener la máquina virtual. La conexión debe estar configurada en la cuenta de Automation donde se ejecuta el runbook. Al usar alertas para desencadenar runbooks, es importante comprobar el estado de la alerta en el runbook que se desencadena. El runbook se desencadena cada vez que la alerta cambia el estado. Las alertas tienen varios estados, y los dos más comunes son Activado y Resuelto. Compruebe el estado de la lógica del runbook para asegurarse de que no se ejecuta más de una vez. En el ejemplo de este artículo se muestra cómo buscar únicamente alertas con el estado Activado.

El runbook usa la identidad administrada asignada por el sistema de la cuenta de Automation para autenticarse en Azure, con el fin de realizar la acción de administración en la máquina virtual. El runbook se puede modificar fácilmente para que use una identidad administrada asignada por el usuario.

> **Nota:**
>
> Se recomienda usar el acceso a la red pública, ya que no es posible usar una alerta de Azure (métrica, registro y registro de actividad) para desencadenar un webhook de Automation cuando la cuenta de Automation usa vínculos privados y está configurado con **Acceso público** establecido en **Deshabilitar**.

Use este ejemplo para crear un runbook llamado **AzureVmInResponsetoVMAlert Stop**. Puede modificar el script de PowerShell y usarlo con muchos recursos diferentes.

1. Inicie sesión en Azure Portal y vaya a la cuenta de Automation.
2. En **Automatización de procesos**, seleccione **Runbooks**.
3. Seleccione **+ Crear un runbook**.
   a. Asigne al runbook el nombre `Stop-AzureVmInResponsetoVMAlert`.
   b. En la lista desplegable **Tipo de Runbook**, seleccione **PowerShell**.

c. Seleccione **Crear**.

4. En el editor de runbooks, pegue el código siguiente:

PowerShell

```
[OutputType("PSAzureOperationResponse")]
param
(
    [Parameter (Mandatory=$false)]
    [object] $WebhookData
)
$ErrorActionPreference = "stop"

if ($WebhookData)
{
    # Get the data object from WebhookData
    $WebhookBody = (ConvertFrom-Json -InputObject $WebhookData.RequestBody)

    # Get the info needed to identify the VM (depends on the payload schema)
    $schemaId = $WebhookBody.schemaId
    Write-Verbose "schemaId: $schemaId" -Verbose
    if ($schemaId -eq "azureMonitorCommonAlertSchema") {
        # This is the common Metric Alert schema (released March 2019)
        $Essentials = [object] ($WebhookBody.data).essentials
        # Get the first target only as this script doesn't handle multiple
        $alertTargetIdArray = (($Essentials.alertTargetIds)[0]).Split("/")
        $SubId = ($alertTargetIdArray)[2]
        $ResourceGroupName = ($alertTargetIdArray)[4]
        $ResourceType = ($alertTargetIdArray)[6] + "/" + ($alertTargetIdArray)
[7]
        $ResourceName = ($alertTargetIdArray)[-1]
        $status = $Essentials.monitorCondition
    }
    elseif ($schemaId -eq "AzureMonitorMetricAlert") {
        # This is the near-real-time Metric Alert schema
        $AlertContext = [object] ($WebhookBody.data).context
        $SubId = $AlertContext.subscriptionId
        $ResourceGroupName = $AlertContext.resourceGroupName
        $ResourceType = $AlertContext.resourceType
        $ResourceName = $AlertContext.resourceName
        $status = ($WebhookBody.data).status
    }
    elseif ($schemaId -eq "Microsoft.Insights/activityLogs") {
        # This is the Activity Log Alert schema
        $AlertContext = [object] (($WebhookBody.data).context).activityLog
        $SubId = $AlertContext.subscriptionId
        $ResourceGroupName = $AlertContext.resourceGroupName
        $ResourceType = $AlertContext.resourceType
        $ResourceName = (($AlertContext.resourceId).Split("/"))[-1]
        $status = ($WebhookBody.data).status
    }
    elseif ($schemaId -eq $null) {
        # This is the original Metric Alert schema
```

```
        $AlertContext = [object] $WebhookBody.context
        $SubId = $AlertContext.subscriptionId
        $ResourceGroupName = $AlertContext.resourceGroupName
        $ResourceType = $AlertContext.resourceType
        $ResourceName = $AlertContext.resourceName
        $status = $WebhookBody.status
    }
    else {
        # Schema not supported
        Write-Error "The alert data schema - $schemaId - is not supported."
    }

    Write-Verbose "status: $status" -Verbose
    if (($status -eq "Activated") -or ($status -eq "Fired"))
    {
        Write-Verbose "resourceType: $ResourceType" -Verbose
        Write-Verbose "resourceName: $ResourceName" -Verbose
        Write-Verbose "resourceGroupName: $ResourceGroupName" -Verbose
        Write-Verbose "subscriptionId: $SubId" -Verbose

        # Determine code path depending on the resourceType
        if ($ResourceType -eq "Microsoft.Compute/virtualMachines")
        {
            # This is an Resource Manager VM
            Write-Verbose "This is an Resource Manager VM." -Verbose

            # Ensures you do not inherit an AzContext in your runbook
            Disable-AzContextAutosave -Scope Process

            # Connect to Azure with system-assigned managed identity
            $AzureContext = (Connect-AzAccount -Identity).context

            # set and store context
            $AzureContext = Set-AzContext -SubscriptionName
$AzureContext.Subscription -DefaultProfile $AzureContext

            # Stop the Resource Manager VM
            Write-Verbose "Stopping the VM - $ResourceName - in resource group
- $ResourceGroupName -" -Verbose
            Stop-AzVM -Name $ResourceName -ResourceGroupName $ResourceGroupName
-DefaultProfile $AzureContext -Force
            # [OutputType(PSAzureOperationResponse")]
        }
        else {
            # ResourceType not supported
            Write-Error "$ResourceType is not a supported resource type for
this runbook."
        }
    }
    else {
        # The alert status was not 'Activated' or 'Fired' so no action taken
        Write-Verbose ("No action taken. Alert status: " + $status) -Verbose
    }
}
else {
```

```
    # Error
    Write-Error "This runbook is meant to be started from an Azure alert
webhook only."
}
```

5. Si desea que el runbook se ejecute con la identidad administrada asignada por el sistema, deje el código tal y como está. Si prefiere usar una identidad administrada asignada por el usuario, haga lo siguiente:
   a. En la línea 78, quite `$AzureContext = (Connect-AzAccount -Identity).context`.
   b. Reemplace el valor por `$AzureContext = (Connect-AzAccount -Identity -AccountId <ClientId>).context`.
   c. Escriba el identificador de cliente que obtuvo anteriormente.

6. Seleccione **Guardar**, luego **Publicar** y, finalmente, **Sí** cuando se le pida confirmación.

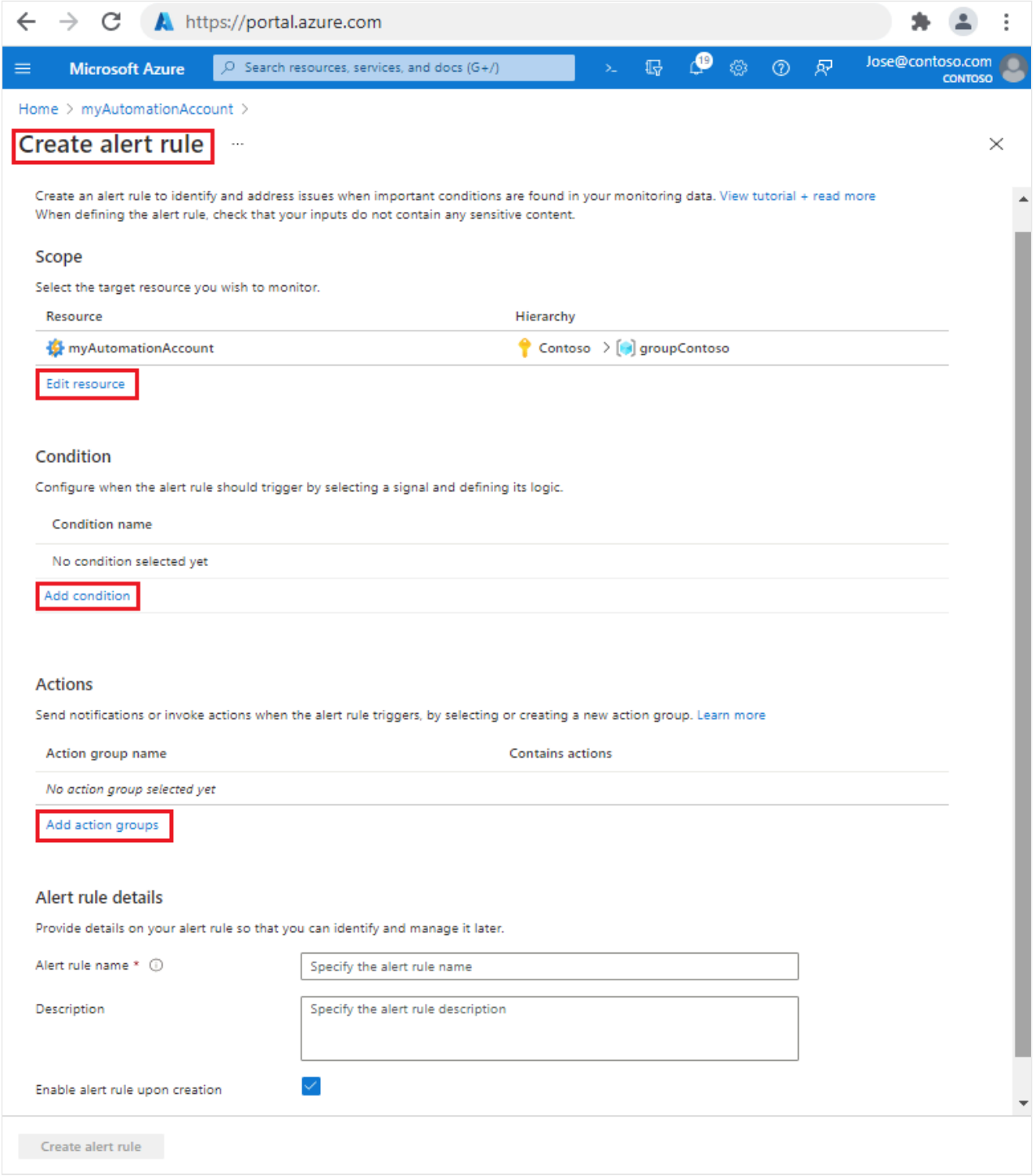
7. Cierre la página **Runbook** para volver a la página **Cuenta de Automation**.

Crear la alerta

Las alertas usan grupos de acciones, que son colecciones de acciones que la alerta desencadena. Los runbooks son simplemente una de las muchas acciones que puede usar con los grupos de acciones.

1. En la cuenta de Automation, en **Supervisión**, seleccione **Alertas**.

2. Seleccione **+ Nueva regla de alertas** para abrir la página **Crear regla de alertas**.



3. En **Ámbito**, seleccione **Editar recurso**.

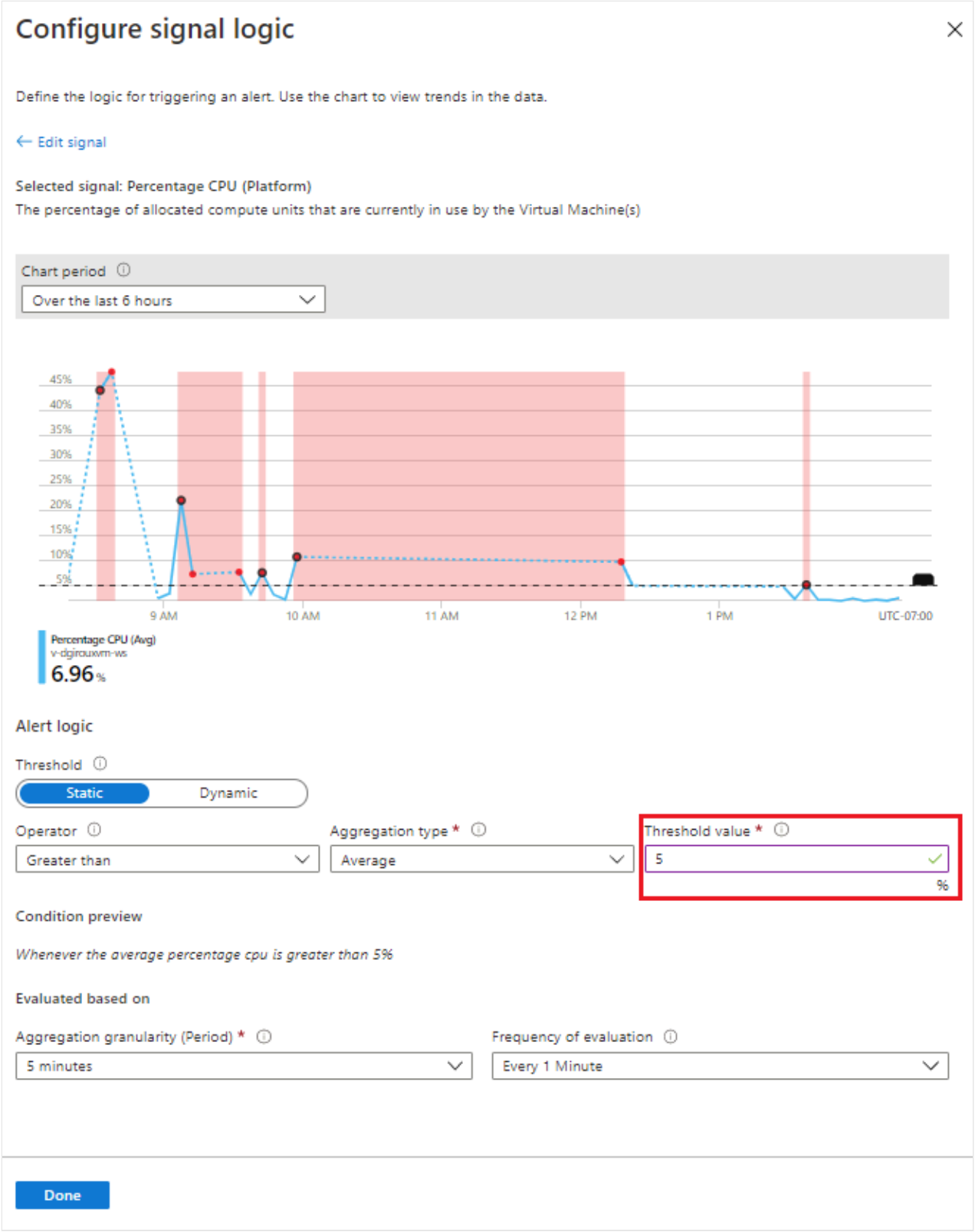
4. En la página **Seleccionar un recurso**, en la lista desplegable **Filtrar por tipo de recurso**, seleccione **Máquinas virtuales**.

5. Active la casilla situada junto a las máquinas virtuales que desea que se supervisen. Luego, seleccione **Listo**, para volver a la página **Crear regla de alertas**.

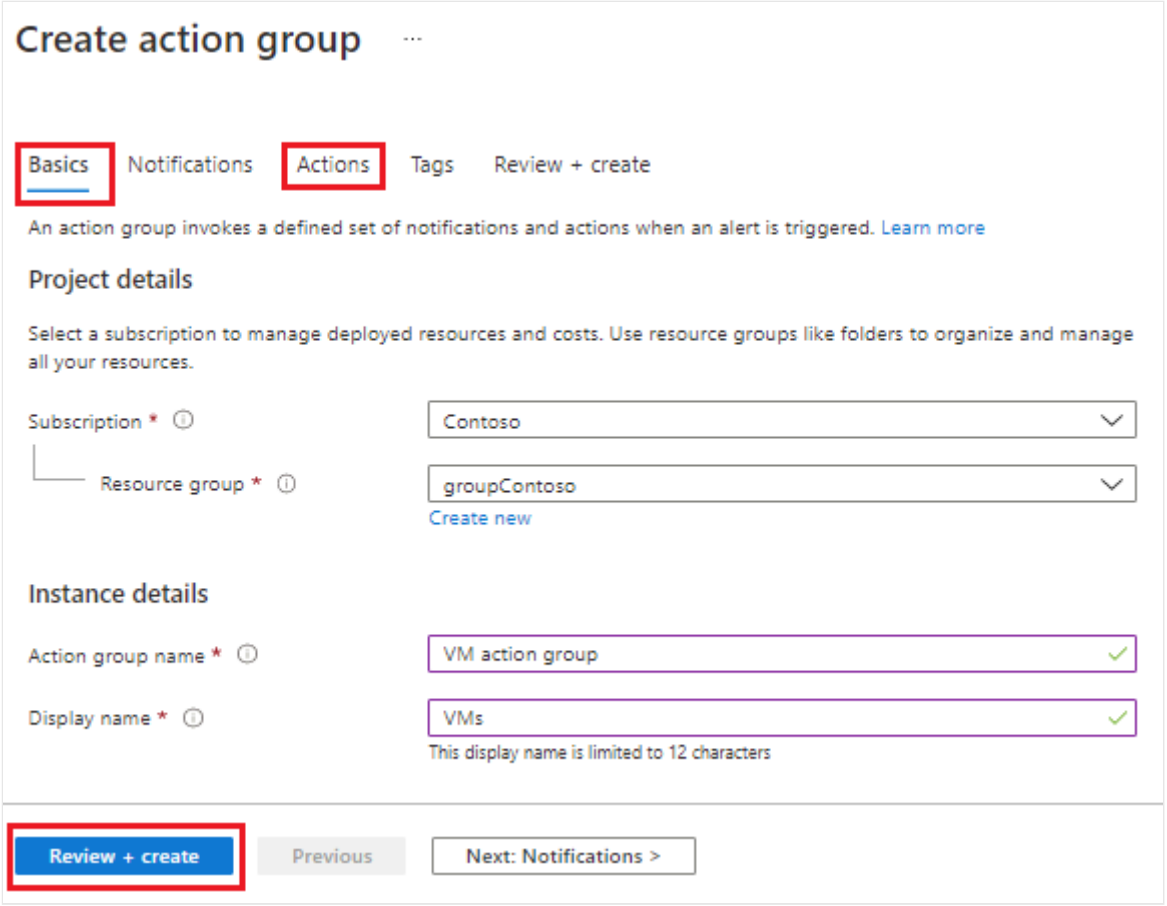
6. En **Condición**, seleccione **Agregar condición**.

7. En la página **Seleccionar una señal**, escriba `Percentage CPU` en el cuadro de texto de búsqueda y seleccione **Porcentaje de CPU** en los resultados.

8. En la página **Configurar lógica de señal**, en **Valor del umbral**, escriba un valor bajo inicial con fines de prueba, como `5`. Puede volver y actualizar este valor una vez que haya confirmado que la alerta funciona según lo esperado. Luego, seleccione **Listo**, para volver a la página **Crear regla de alertas**.



9. En **Acciones**, seleccione **Agregar grupos de acciones** y después **+ Crear grupo de acciones**.



10. En la página **Crear grupo de acciones**:

    a. En la pestaña **Aspectos básicos**, escriba los valores correspondientes en **Nombre del grupo de acciones** y **Nombre para mostrar**.

    b. En la pestaña **Acciones**, en el cuadro de texto **Nombre**, escriba un nombre. A continuación, en la lista desplegable **Tipo de acción**, seleccione **Runbook de Automation** para abrir la página **Configurar runbook**.

        i. En el elemento **Origen de Runbook**, seleccione **Usuario**.

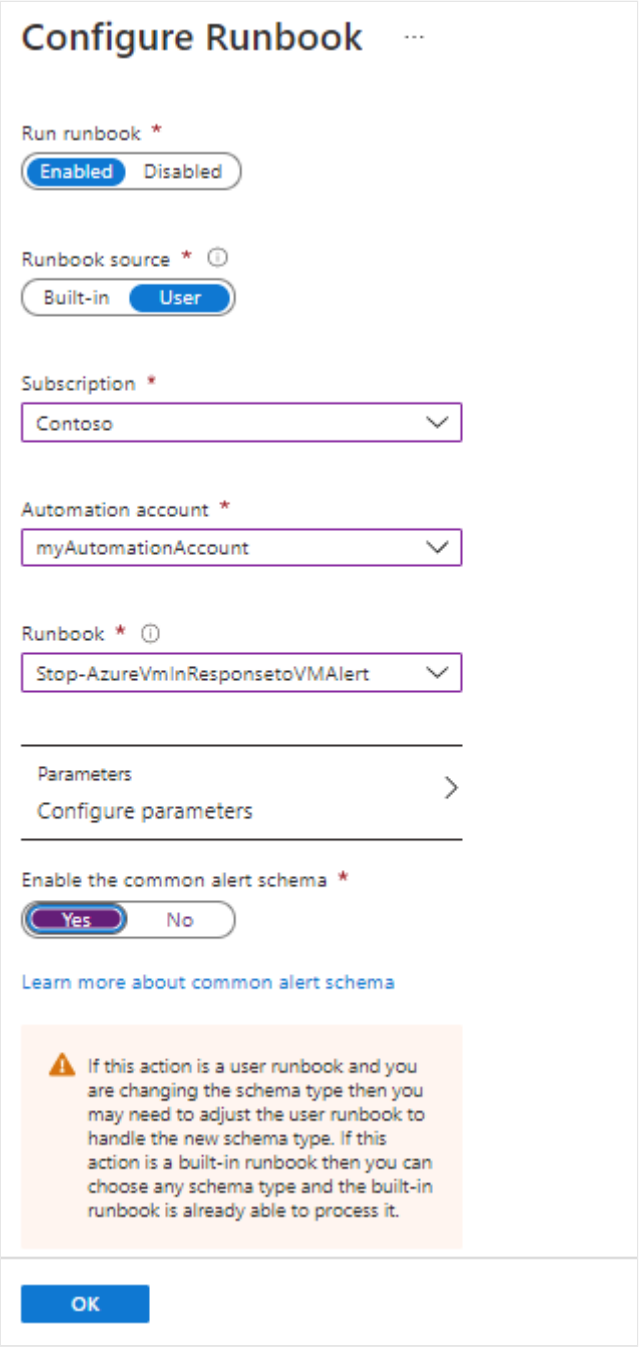
        ii. En la lista desplegable **Suscripción**, seleccione una suscripción.

        iii. En la lista desplegable **Cuenta de Automation**, seleccione su cuenta de Automation.

        iv. En la lista desplegable **Runbook**, seleccione **Stop-AzureVmInResponsetoVMAlert**.

        v. En el elemento **Habilitar el esquema de alerta común**, seleccione **Sí**.

        vi. Seleccione **Aceptar** para volver a la página **Crear grupo de acciones**.



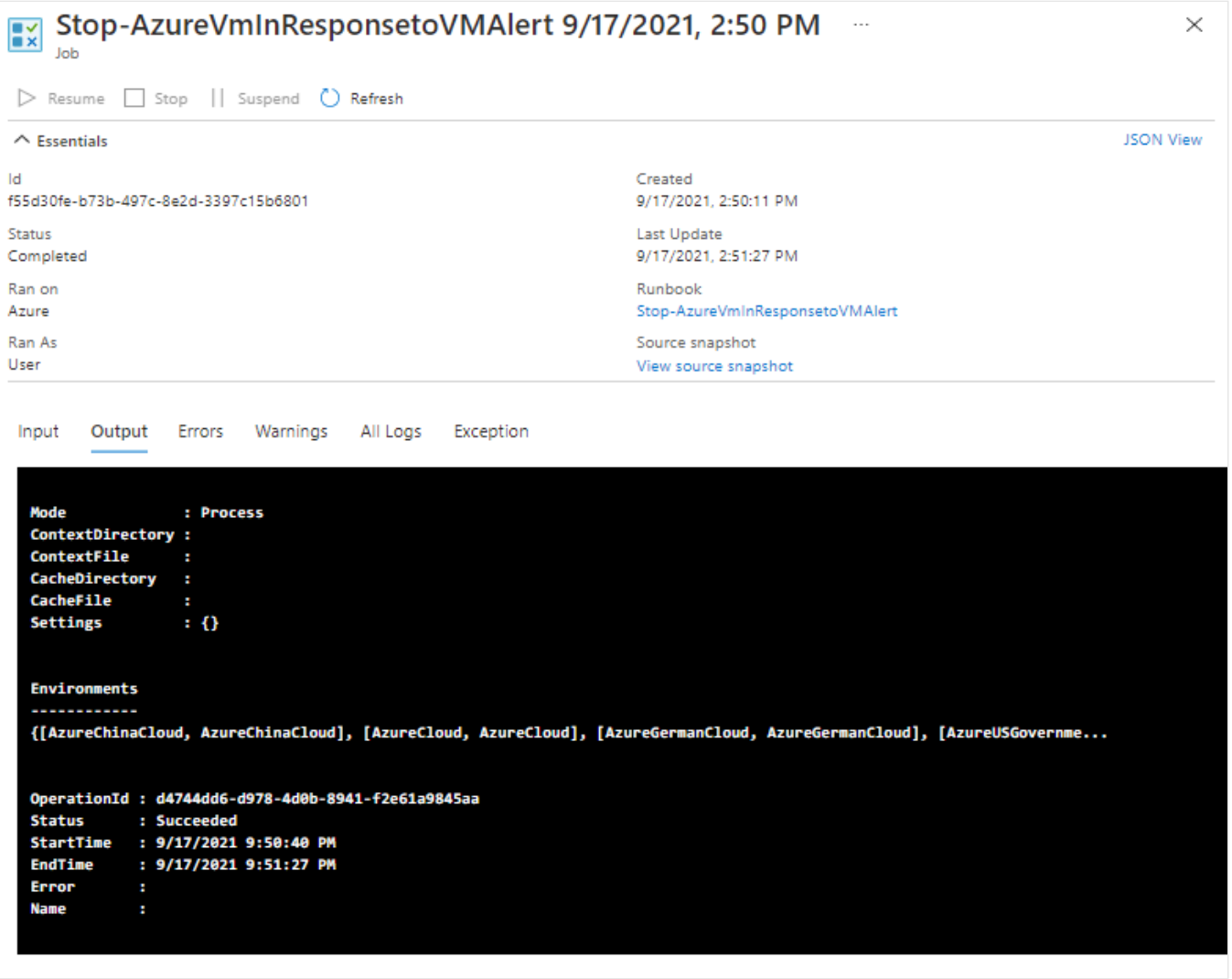
c. Seleccione **Revisar y crear** y, después, **Crear** para volver a la página **Crear regla de alertas**.

11. En **Detalles de la regla de alertas**, para el cuadro de texto **Nombre de la regla de alertas**.

12. Seleccione **Crear regla de alertas**. Puede usar el grupo de acciones en las alertas de registro de actividad y las alertas casi en tiempo real que cree.

Comprobación

Asegúrese de que la máquina virtual se está ejecutando. Vaya al runbook **Stop-AzureVmInResponsetoVMAlert** y observe la lista **Trabajos recientes** para rellenarla. Una vez que aparezca un trabajo completado, selecciónelo y revise la salida. Compruebe también si la máquina virtual se ha detenido.



Operaciones comunes de administración de máquinas virtuales de Azure

Azure Automation proporciona scripts para operaciones comunes de administración de máquinas virtuales de Azure, como escenarios de reinicio, detención, eliminación y escalado y reducción vertical de máquinas virtuales en la galería de runbooks. Los scripts también se pueden encontrar en el repositorio de GitHub de Azure Automation. Además, puede usar estos scripts como se mencionó en los pasos anteriores.

Expandir tabla

Operaciones de administración de máquinas virtuales de Azure	Detalles
Stop-Azure-VM-On-Alert	Este runbook detendrá una máquina virtual de Azure Resource Manager en respuesta a un desencadenador de alertas de Azure. El recurso de destino de la alerta desencadenada debe ser la máquina virtual para detenerse. Esto se pasa en un parámetro de entrada de la

Operaciones de administración de máquinas virtuales de Azure	Detalles
	carga de alerta desencadenada. Se debe llamar al runbook desde una alerta de Azure a través de un webhook. La versión más reciente del módulo Az debe agregarse a la cuenta de Automation. La identidad administrada debe estar habilitada y se debe dar acceso de colaborador a la cuenta de Automation.
Restart-Azure-VM-On-Alert	Este runbook detendrá una máquina virtual de Azure Resource Manager en respuesta a un desencadenador de alertas de Azure. El recurso de destino de la alerta desencadenada debe ser la máquina virtual para reiniciarse. Esto se pasa en un parámetro de entrada de la carga de alerta desencadenada. Se debe llamar al runbook desde una alerta de Azure a través de un webhook. La versión más reciente del módulo Az debe agregarse a la cuenta de Automation. La identidad administrada debe estar habilitada y se debe dar acceso de colaborador a la cuenta de Automation.
Delete-Azure-VM-On-Alert	Este runbook detendrá una máquina virtual de Azure Resource Manager en respuesta a un desencadenador de alertas de Azure. El recurso de destino de la alerta desencadenada debe ser la máquina virtual para eliminarse. Esto se pasa en un parámetro de entrada de la carga de alerta desencadenada. Se debe llamar al runbook desde una alerta de Azure a través de un webhook. La versión más reciente del módulo Az debe agregarse a la cuenta de Automation. La identidad administrada debe estar habilitada y se debe dar acceso de colaborador a la cuenta de Automation.
ScaleDown-Azure-VM-On-Alert	Este runbook detendrá una máquina virtual de Azure Resource Manager en respuesta a un desencadenador de alertas de Azure.

Operaciones de administración de máquinas virtuales de Azure	Detalles
	El recurso de destino de la alerta desencadenada debe ser la máquina virtual para reducirse verticalmente. Esto se pasa en un parámetro de entrada de la carga de alerta desencadenada. Se debe llamar al runbook desde una alerta de Azure a través de un webhook. La versión más reciente del módulo Az debe agregarse a la cuenta de Automation. La identidad administrada debe estar habilitada y se debe dar acceso de colaborador a la cuenta de Automation.
ScaleUp-Azure-VM-On-Alert	Este runbook detendrá una máquina virtual de Azure Resource Manager en respuesta a un desencadenador de alertas de Azure. El recurso de destino de la alerta desencadenada debe ser la máquina virtual para escalarse verticalmente. Esto se pasa en un parámetro de entrada de la carga de alerta desencadenada. Se debe llamar al runbook desde una alerta de Azure a través de un webhook. La versión más reciente del módulo Az debe agregarse a la cuenta de Automation. La identidad administrada debe estar habilitada y se debe dar acceso de colaborador a la cuenta de Automation.

Pasos siguientes

- Obtenga más información acerca de cómo crear presupuesto y asociarlo con runbook de Automation.
- Para descubrir diferentes maneras de iniciar un runbook, consulte Inicio de un runbook.
- Para crear una alerta de registro de actividad, consulte Creación de alertas del registro de actividad.
- Para aprender a crear una alerta casi en tiempo real, consulte Creación de una regla de alertas en Azure Portal.

Last updated on 17/11/2025

Seguimiento de archivos actualizados con una tarea de monitor

Azure Automation usa una tarea de observador para buscar eventos y desencadenar acciones con runbooks de PowerShell. La tarea de observador contiene dos partes, el observador y la acción. Un runbook de monitor se ejecuta en un intervalo definido en la tarea de monitor y envía los datos a un runbook de acción.

> **Nota:**
>
> Las tareas de observador no se admiten en Microsoft Azure operado por 21Vianet.

> **Importante**
>
> A partir de mayo de 2020, el uso de Azure Logic Apps es la manera recomendada y compatible de supervisar eventos, programar tareas periódicas y desencadenar acciones. Consulte Programación y ejecución de tareas, procesos y flujos de trabajo automatizados periódicos con Azure Logic Apps.

Este artículo le guía a través de la creación de una tarea de monitor para supervisar cuándo se agrega un nuevo archivo a un directorio. Aprenderá a:

- Importar un runbook de monitor
- Crear una variable de automatización
- Crear un runbook de acción
- Crear una tarea de observador
- Desencadenar un monitor
- Inspección de la salida

Prerrequisitos

Para completar este artículo, se requieren lo siguiente:

- Suscripción de Azure. Si aún no tiene una, puede activar las ventajas del suscriptor de MSDN o registrarse para obtener una cuenta gratuita.
- Cuenta de Automation para contener los runbooks de monitor y de acción y la tarea de monitor.
- Hybrid Runbook Worker donde se ejecuta la tarea de monitor.

- Runbooks de PowerShell. Los runbooks de flujo de trabajo de PowerShell y los runbooks gráficos no son compatibles con las tareas de monitor.

Importar un runbook de monitor

En este artículo se utiliza un runbook de tipo watcher llamado **Watcher runbook que busca nuevos archivos en un directorio** para buscar nuevos archivos en un directorio. El runbook de monitor recupera la última hora de escritura conocida en los archivos de una carpeta y examina los archivos más recientes que esa marca de agua.

Puede importar este runbook a la cuenta de Automation desde el portal mediante los pasos siguientes.

1. Inicie sesión en Azure Portal.
2. Busque y seleccione **Cuentas de Automatización**.
3. En la página **Automation Accounts**, seleccione el nombre de su Cuenta de Automation en la lista.
4. En el panel izquierdo, seleccione **Galería de Runbooks** en **Automatización de Procesos**.
5. Asegúrese de que **GitHub** está seleccionado en la lista desplegable **Origen** .
6. Busque el **runbook de monitor**.
7. Seleccione **Watcher Runbook que busca nuevos archivos en un directorio** y seleccione **Importar** en la página de detalles.
8. Asigne un nombre al runbook y, opcionalmente, una descripción y haga clic en **Aceptar** para importar el runbook en su cuenta de Automation. Debería ver un mensaje **Importación exitosa** en la esquina superior derecha de su ventana.
9. El runbook importado aparece en la lista bajo el nombre que le diste cuando seleccionas Runbooks en el panel izquierdo.
10. Haga clic en el runbook y, en la página de detalles del runbook, seleccione **Editar** y, a continuación, haga clic en **Publicar**. Cuando se le solicite, haga clic en **Sí** para publicar el runbook.

También puede descargar el runbook desde la organización GitHub de Azure Automation.

1. Vaya a la página de la organización GitHub de Azure Automation para Watch-NewFile.ps1.
2. Para descargar el runbook desde GitHub, seleccione **Código** en el lado derecho de la página y, a continuación, seleccione **Descargar archivo ZIP** para descargar todo el código en un archivo zip.
3. Extraiga el contenido e importe el runbook.

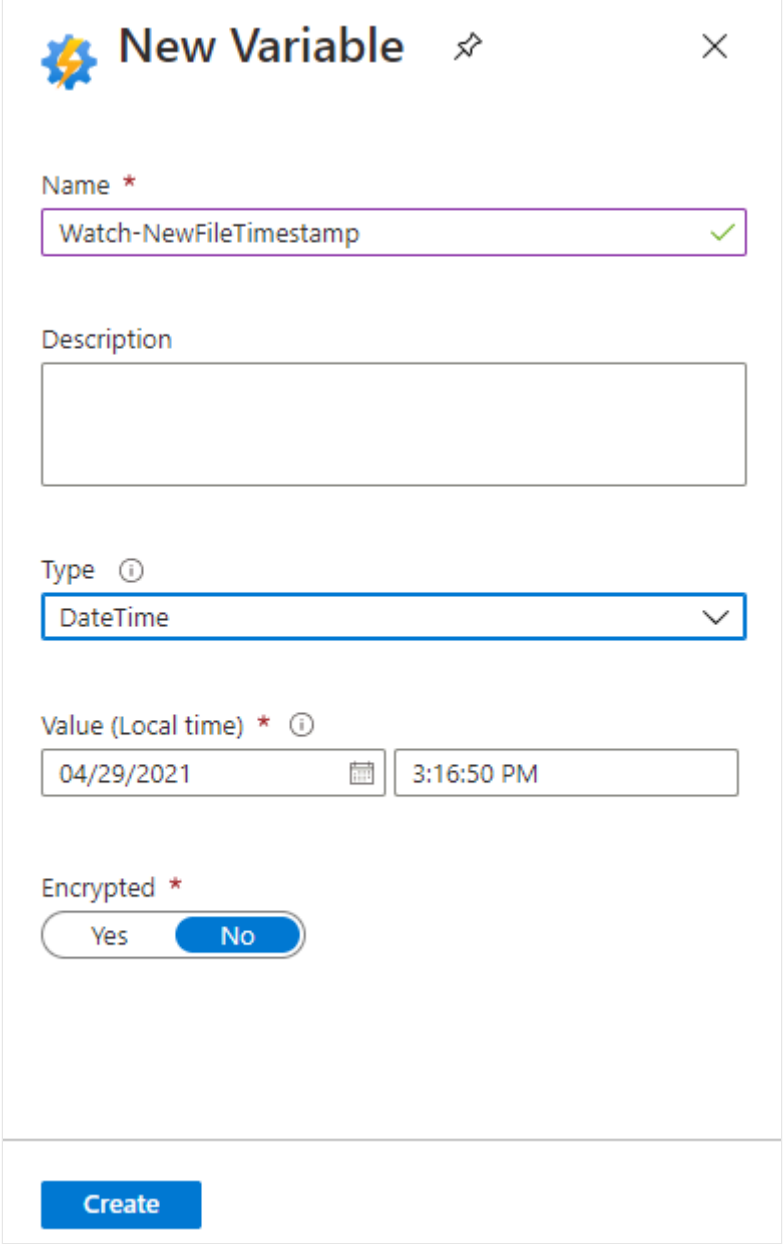
Crear una variable de automatización

Se usa una variable de Automation para almacenar las marcas de tiempo que el runbook anterior lee y almacena de cada archivo.

1. Seleccione **Variables** en **Recursos compartidos** y haga clic en **+ Agregar una variable**.

2. Escriba **Watch-NewFileTimestamp** como nombre.

3. Seleccione **DateTime** para el tipo. Se configurará automáticamente con la fecha y hora actuales por defecto.



4. Haga clic en **Crear** para crear la variable automation.

Crear un runbook de acción

Un runbook de acción se usa en una tarea de monitor para actuar sobre los datos transferidos a él desde un runbook de monitor. Debe importar un runbook de acción predefinido desde

Azure Portal o desde la organización GitHub de Azure Automation.

Puede importar este runbook a la cuenta de Automation desde Azure Portal:

1. Inicie sesión en Azure Portal.
2. Busque y seleccione **Cuentas de Automatización**.
3. En la página **Automation Accounts**, seleccione el nombre de su Cuenta de Automation en la lista.
4. En el panel izquierdo, seleccione **Galería de Runbooks** en **Automatización de Procesos**.
5. Asegúrese de que **GitHub** está seleccionado en la lista desplegable **Origen** .
6. Busque **Watcher action**, seleccione **Watcher action that processes events triggered by a watcher runbook** (Acción de monitor que procesa eventos desencadenados por un runbook de monitor) y haga clic en **Importar**.
7. Opcionalmente, cambie el nombre del runbook en la página de importación y, a continuación, haga clic en **Aceptar** para importar el runbook. Debería ver un mensaje **Importar correctamente** en el panel de notificaciones en la parte superior derecha del explorador.
8. Vaya a la página Cuenta de Automation y haga clic en **Runbooks** a la izquierda. El nuevo runbook debe aparecer debajo del nombre que le asignó en el paso anterior. Haga clic en el runbook y, en la página de detalles del runbook, seleccione **Editar** y, a continuación, haga clic en **Publicar**. Cuando se le solicite, haga clic en **Sí** para publicar el runbook.

Para crear un runbook de acción descargándolo de la organización GitHub de Azure Automation:

1. Vaya a la página de la organización de GitHub de Azure Automation para Process-NewFile.ps1.
2. Para descargar el runbook desde GitHub, seleccione **Código** en el lado derecho de la página y, a continuación, seleccione **Descargar archivo ZIP** para descargar todo el código en un archivo zip.
3. Extraiga el contenido e importe el runbook.

Crear una tarea de observador

En este paso, va a configurar la tarea de monitor que hace referencia a los runbooks de monitor y de acción definidos en las secciones anteriores.

1. Vaya a la cuenta de Automation y seleccione **Tareas de monitor** en **Automatización de procesos**.

2. Seleccione la página Tareas de vigilancia y haga clic en **+ Agregar una tarea de vigilante**.

3. Escriba **WatchMyFolder** como nombre.

4. Seleccione **Configurar monitor** y elija el runbook **Watch-NewFile**.

5. Escriba los valores siguientes para los parámetros:

    - **FOLDERPATH** : una carpeta de Hybrid Runbook Worker donde se crean nuevos archivos, por ejemplo, **d:\examplefiles**.
    - **EXTENSIÓN: extensión** para la configuración. Deje en blanco para procesar todas las extensiones de archivo.
    - **RECURSE** : operación recursiva. Deje este valor como valor predeterminado.
    - **CONFIGURACIÓN DE EJECUCIÓN** : configuración para ejecutar el runbook. Elija trabajador híbrido.

6. Haga clic en **Aceptar** y, a continuación, en **Seleccionar** para volver a la página Watcher.

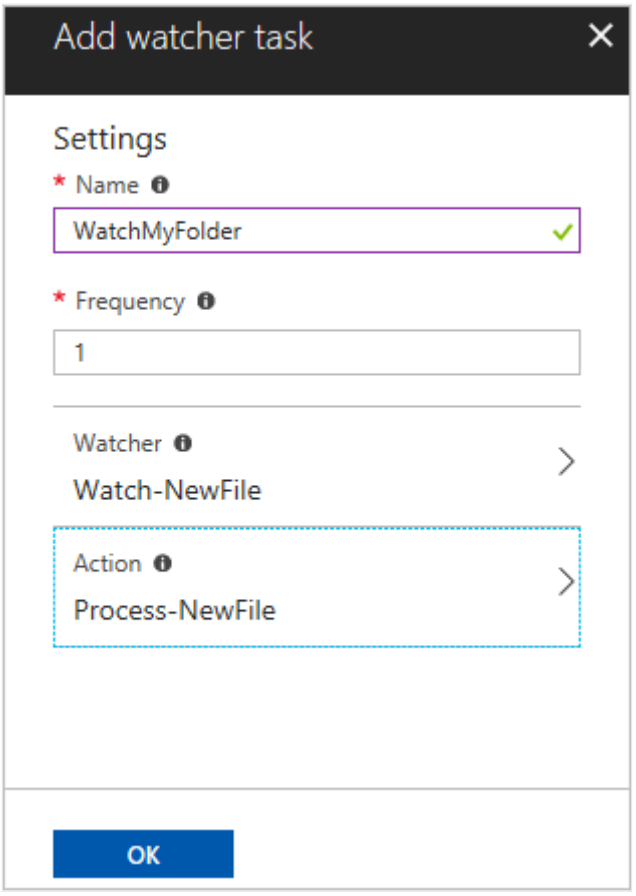
7. Seleccione **Configurar acción** y elija el directorio de operaciones **Process-NewFile**.

8. Escriba los valores siguientes para los parámetros:

    - **EVENTDATA** : datos de eventos. Dejar en blanco. Los datos se pasan desde el runbook de monitor.
    - **Configuración de ejecución** - Configuración para la ejecución del runbook. Déjelo como Azure porque este runbook se ejecuta en Azure Automation.

9. Haga clic en **Aceptar** y, a continuación, en **Seleccionar** para volver a la página Watcher.

10. Haga clic en **Aceptar** para crear la tarea de supervisión.



Desencadenar un monitor

Debe ejecutar una prueba como se describe a continuación para asegurarse de que la tarea de monitoreo funciona como se espera.

1. Inicie una sesión remota en Hybrid Runbook Worker.
2. Abra **PowerShell** y cree un archivo de prueba en la carpeta .

azurepowerShell

```
New-Item -Name ExampleFile1.txt
```

En el ejemplo siguiente se muestra la salida esperada.

Resultados

```
    Directory: D:\examplefiles


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----       12/11/2017   9:05 PM              0 ExampleFile1.txt
```

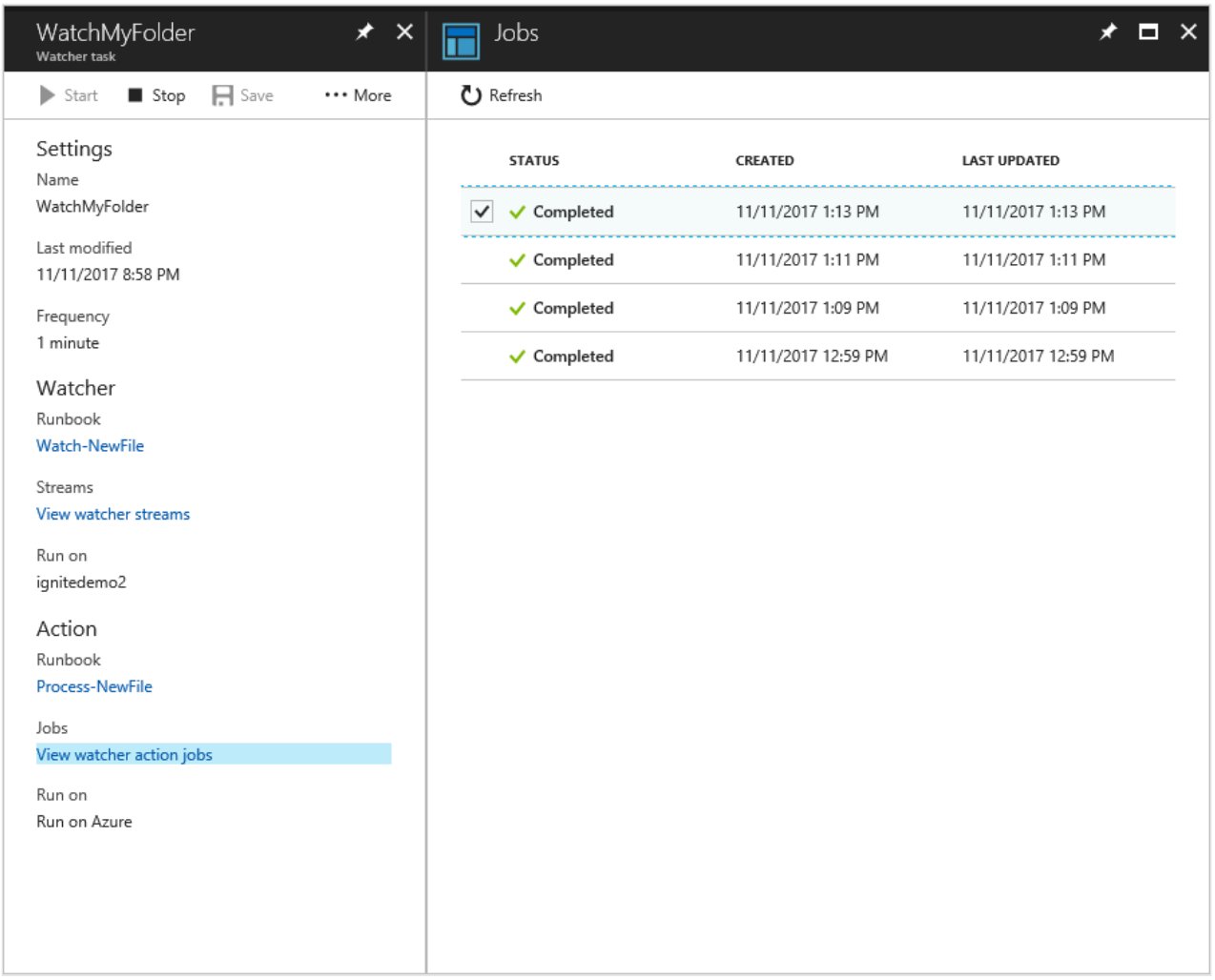
Inspección de la salida

1. Vaya a la cuenta de Automation y seleccione **Tareas de monitor** en **Automatización de procesos**.

2. Seleccione la tarea del observador **WatchMyFolder**.

3. Haga clic en **Ver secuencias de monitor** en **Secuencias** para ver si el monitor ha encontrado el archivo nuevo y ha iniciado el runbook de acción.

4. Haga clic en **Ver trabajos de acción del monitor** para ver los trabajos del runbook de acción. Cada trabajo se puede seleccionar para ver los detalles del trabajo.



La salida esperada cuando se encuentra el nuevo archivo se puede ver en el ejemplo siguiente:

Resultados

```
Message is Process new file...

Passed in data is @{FileName=D:\examplefiles\ExampleFile1.txt; Length=0}
```

Pasos siguientes

Para más información sobre cómo crear su propio runbook, consulte Creación de un runbook de PowerShell.

Last updated on 17/11/2025

Administración de servicios de Office 365

Artículo • 15/09/2024

Puede usar Azure Automation para la administración de los servicios de suscripción de Office 365, en productos como Microsoft Word y Microsoft Outlook. Las interacciones con Office 365 están habilitadas por iMicrosoft Entra ID. Consulte Uso de Microsoft Entra ID en Azure Automation para autenticarse en Azure.

Requisitos previos

Necesita los siguientes elementos para administrar los servicios de suscripción de Office 365 en Azure Automation.

- Suscripción a Azure. Consulte Guía de decisiones de suscripción.
- Un objeto de Automation en Azure que contenga las credenciales y los runbooks de la cuenta de usuario. Consulte Introducción a Azure Automation.
- Microsoft Entra ID. Consulte Uso de Microsoft Entra ID en Azure Automation para autenticarse en Azure.
- Un inquilino de Office 365, con una cuenta. Consulte Configurar el espacio empresarial de Office 365.

Instalar Microsoft Graph PowerShell

El uso de Office 365 dentro de Azure Automation requiere el módulo Microsoft Graph PowerShell.

PowerShell

```
Install-Module Microsoft.Graph -Scope CurrentUser
```

> **Nota**
>
> Para usar MSOnline PowerShell, debe ser miembro de Microsoft Entra ID. Los usuarios invitados no pueden usar el módulo.

Creación de una cuenta de Azure Automation

Para completar los pasos de este artículo, necesitará una cuenta en Azure Automation. Consulte Creación de una cuenta de Azure Automation.

Creación de un recurso de credencial (opcional)

La creación de un recurso de credencial para el usuario administrativo de Office 365 que tiene permisos para ejecutar el script es opcional; sin embargo, puede ser útil para evitar la exposición de nombres de usuario y contraseñas dentro de scripts de PowerShell. Para obtener instrucciones, vea Creación de un recurso de credencial.

Creación de una cuenta de servicio de Office 365

Para ejecutar los servicios de suscripción de Office 365, necesita una cuenta de servicio de Office 365 con permisos para hacer lo que desee. Puede usar una cuenta de administrador de administración de permisos, una cuenta por servicio o tener una función o script para ejecutar. En cualquier caso, la cuenta de servicio requiere una contraseña compleja y segura. Consulte Configurar Office 365 para empresas.

Conexión al servicio en línea de Microsoft Entra

> **Nota**
>
> Para usar los cmdlets de módulo MSOnline, debe ejecutarlos desde Windows PowerShell. PowerShell Core no es compatible con estos cmdlets.

Puede conectarse a Microsoft Entra ID desde la suscripción a Office 365. La conexión usa un nombre de usuario y una contraseña de Office 365 o la autenticación multifactor (MFA). Puede conectarse mediante Azure Portal o un símbolo del sistema de Windows PowerShell (no tiene que ser elevado).

A continuación se muestra el ejemplo de PowerShell. Para más información, consulte Connect-MgGraph.

PowerShell

```
Connect-MgGraph -Scopes "Directory.Read.All"
```

Si no recibe ningún error, se ha conectado correctamente. Una prueba rápida es ejecutar un cmdlet de Office 365, por ejemplo, Get-MgUser, y ver los resultados.

Creación de un runbook de PowerShell a partir de un script existente

Puede acceder a la funcionalidad de Office 365 desde un script de PowerShell.

PowerShell

```
$emailFromAddress = "admin@TenantOne.com"
$emailToAddress = "servicedesk@TenantOne.com"
$emailSMTPServer = "outlook.office365.com"
$emailSubject = "Office 365 License Report"
$credObject = Get-AutomationPSCredential -Name "Office-Credentials"

Connect-MgGraph -Scopes "Directory.Read.All"

$O365Licenses = Get-MgSubscribedSku | Out-String
Send-MailMessage -Credential $credObject -From $emailFromAddress -To
$emailToAddress -Subject $emailSubject -Body $O365Licenses -SmtpServer
$emailSMTPServer -UseSSL
```

Ejecución del script en un runbook

Puede usar el script en un runbook de Azure Automation. A modo de ejemplo, usaremos el tipo de runbook de PowerShell.

1. Cree un runbook de PowerShell. Consulte Creación de un runbook de Azure Automation.
2. En la cuenta de Automation, seleccione **Runbooks** en **Automatización de procesos**.
3. Seleccione el nuevo runbook y haga clic en **Editar**.
4. Copie el script y péguelo en el editor de texto para el runbook.
5. Seleccione **ASSETS** (Recursos), expanda **Credentials** (Credenciales) y compruebe que la credencial de Office 365 está incluida.
6. Haga clic en **Save**(Guardar).
7. Seleccione **Panel de prueba** y, a continuación, haga clic en **Iniciar** para empezar a probar el runbook. Consulte Administración de runbooks en Azure Automation.
8. Una vez finalizada la prueba, salga del panel prueba.

Publicación y programación del runbook

Para publicar y programar el runbook, consulte Administración de runbooks en Azure Automation.

Pasos siguientes

- Para obtener más información sobre el uso de credenciales, vea Administración de credenciales en Azure Automation.
- Para obtener más información sobre los módulos, vea Administración de módulos en Azure Automation.
- Si necesita iniciar un runbook, vea Inicio de un runbook en Azure Automation.
- Para obtener información sobre PowerShell, vea Documentación de PowerShell.

Comentarios

¿Le ha resultado útil esta página? Sí No

Proporcionar comentarios sobre el producto | Obtener ayuda en Microsoft Q&A

Implementa una VM de Amazon Web Services mediante un runbook

En este artículo aprenderá cómo se puede aprovechar Azure Automation para aprovisionar una máquina virtual en su suscripción de Amazon Web Services (AWS) y asignar a dicha máquina virtual un nombre específico, lo que AWS denomina "etiquetar" la máquina virtual.

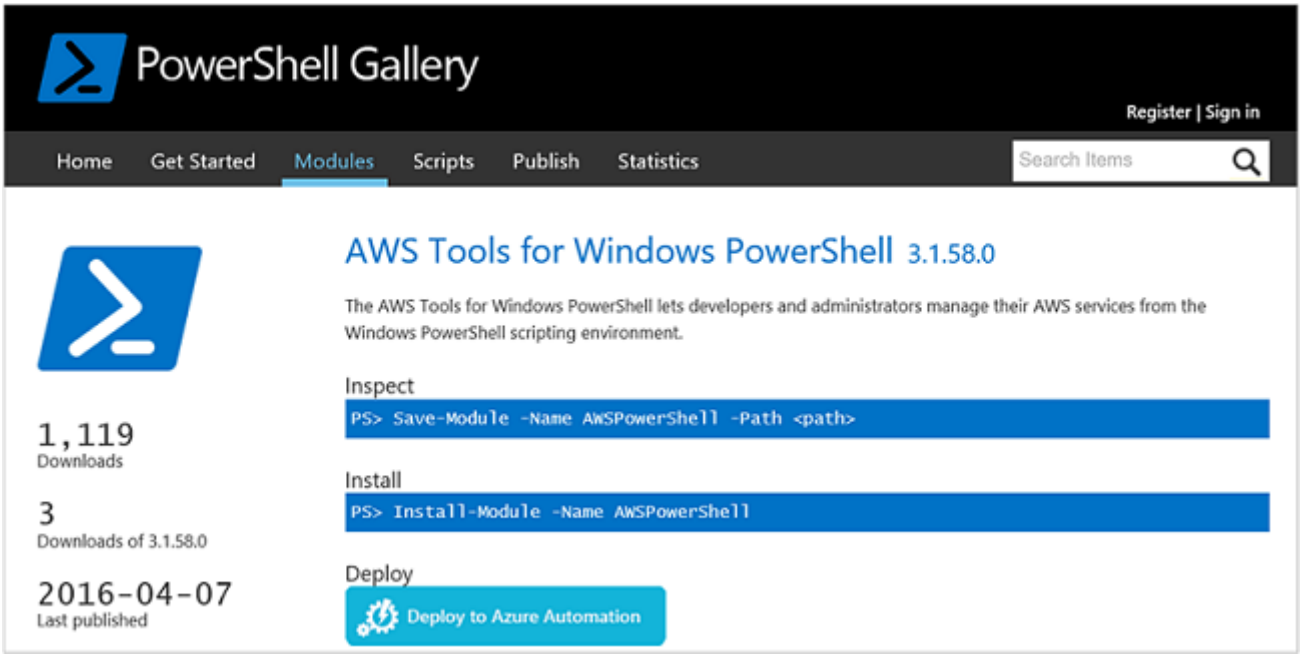
Requisitos previos

Debe tener una cuenta de Azure Automation y una suscripción de Amazon Web Services (AWS). Para más información sobre cómo configurar una cuenta de Azure Automation con sus credenciales de la suscripción de AWS, consulte Configuración de la autenticación con Amazon Web Services. Antes de continuar, debe crear o actualizar esta cuenta con las credenciales de suscripción de AWS, ya que en las secciones siguientes haremos referencia a esta cuenta.

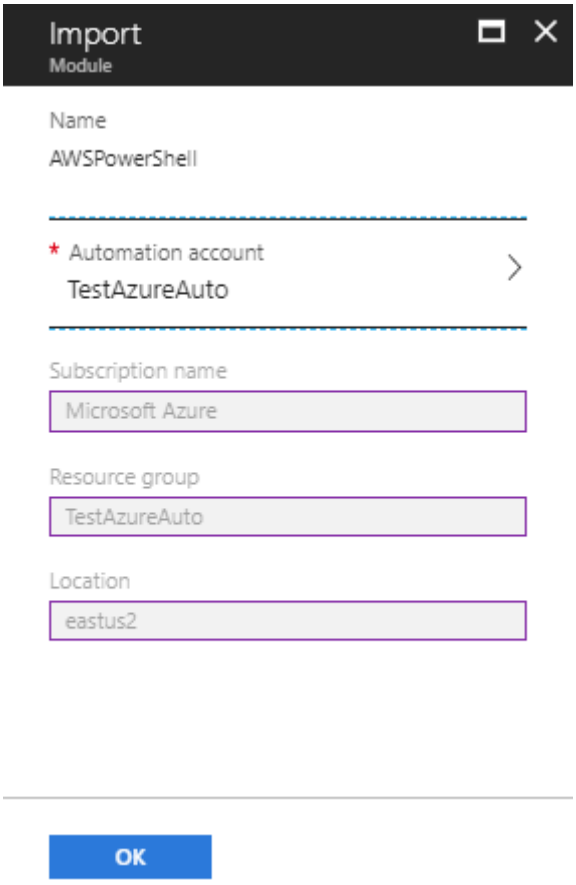
Implementar un módulo Amazon Web Services de PowerShell

El manual de procedimientos de aprovisionamiento de máquinas virtuales usa el módulo de AWS para PowerShell para llevar a cabo su tarea. Utilice los pasos siguientes para agregar el módulo a la cuenta de Automation que esté configurada con sus credenciales de suscripción de AWS.

1. Abra el explorador web, navegue a la Galería de PowerShell y haga clic en el **botón Deploy to Azure Automation** (Implementar en Azure Automation).



2. Aparecerá la página de inicio de sesión de Azure y, una vez que se autentique, se le dirigirá a Azure Portal y se mostrará la página siguiente:



3. Seleccione la cuenta de Automation que usar y haga clic en **Aceptar** para iniciar la implementación.

> ⓘ **Nota:**

Cuando Azure Automation importa un módulo de PowerShell, extrae los cmdlets. Las actividades no aparecen hasta que Automation haya terminado de importar el módulo y extraer los cmdlets. El proceso puede tardar unos minutos.

4. En Azure Portal, abra la cuenta de Automation.
5. Haga clic en el mosaico **Recursos** y, en el panel Recursos, seleccione el icono **Módulos**.
6. En la página Módulos, verá el módulo **AWSPowerShell** en la lista.

Implementar AWS y crear runbook de VM

Una vez que haya implementado el módulo AWS de PowerShell, puede crear un runbook para automatizar el aprovisionamiento de una máquina virtual en AWS mediante un script de PowerShell. Los pasos siguientes muestran cómo usar un script nativo de PowerShell en Azure Automation.

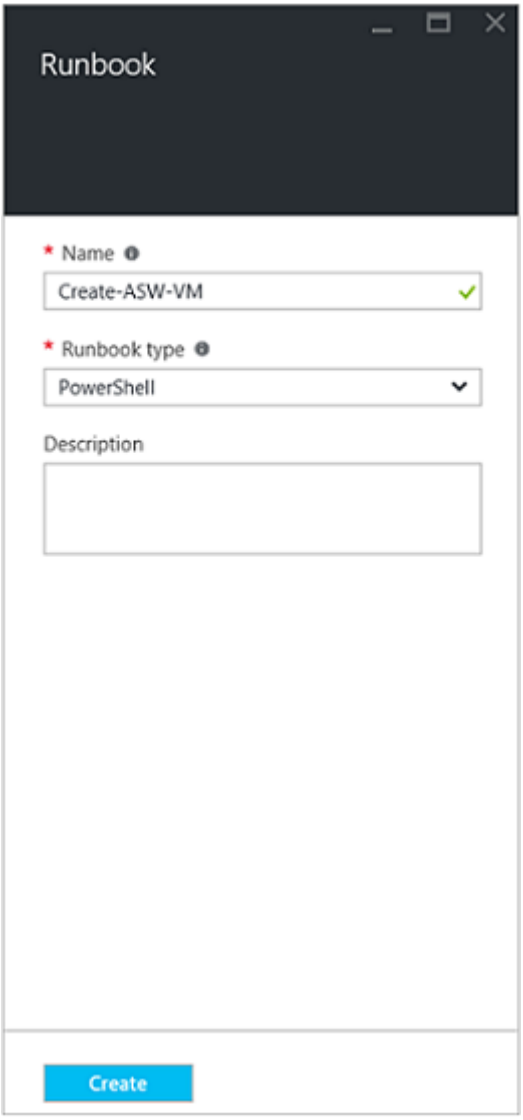
> **Nota:**
>
> Para obtener más información y conocer las opciones de este script, visite la Galería de PowerShell.

1. Descargue el script de New-AwsVM de PowerShell de la Galería de PowerShell; para ello, abra una sesión de PowerShell y escriba el comando siguiente:

   PowerShell
   ```
   Save-Script -Name New-AwsVM -Path <path>
   ```

2. En Azure Portal, abra la cuenta de Automation y seleccione **Runbooks** en **Automatización de procesos**.
3. En la página Runbooks, seleccione **Agregar un Runbook**.
4. En el panel Agregar un Runbook, seleccione **Creación rápida** para crear un nuevo runbook.
5. En el panel de propiedades del runbook, escriba un nombre para el runbook.

6. En la lista desplegable **Tipo de Runbook**, seleccione **PowerShell** y, a continuación, haga clic en **Crear**.



7. Cuando aparezca la página Editar Runbook de PowerShell, copie y pegue el script de PowerShell en el lienzo de creación de runbooks.

```
34 #>
35 #ToDO:
36 #Change the default values for the following parameters if they do not match with yours:
37 #AWSRegion, EC2ImageName, MinCount, MaxCount,InstanceType
38 #Create an Azure Automation Asset called "AwsCred"
39 #Turn on Log verbose records and optionally Log progress records under the runbook settings to see verbose
40
41 param (
42     [Parameter(Mandatory=$true)]
43     [string]$VMname,
44     [ValidateNotNullOrEmpty()]
45     [string]$AWSRegion = "us-west-2",
46     [ValidateNotNullOrEmpty()]
47     [string]$EC2ImageName = "WINDOWS_2012R2_BASE",
48     [ValidateNotNullOrEmpty()]
49     [string]$MinCount = 1,
50     [ValidateNotNullOrEmpty()]
51     [string]$MaxCount = 1,
52     [ValidateNotNullOrEmpty()]
53     [string]$InstanceType = "t2.micro"
54     )
55
56 # Get credentials to authenticate against AWS
57 $AwsCred = Get-AutomationPSCredential -Name "AwsCred"
58 $AwsAccessKeyId = $AwsCred.UserName
59 $AwsSecretKey = $AwsCred.GetNetworkCredential().Password
```

Tenga en cuenta lo siguiente al trabajar con el script de PowerShell de ejemplo:

- El Runbook contiene una serie de valores de parámetro predeterminados. Evalúe todos los valores predeterminados y actualícelos cuando sea necesario.
- Si ha almacenado las credenciales de AWS como un recurso de credencial con un nombre diferente de `AWScred`, debe actualizar el script en la línea 57 como corresponda.
- Cuando trabaja con los comandos de la CLI de AWS en PowerShell, especialmente con este manual de procedimientos de ejemplo, debe especificar la región de AWS. En caso contrario, se producirá un error en los cmdlets. Consulte el tema de AWS Specify AWS Region (Especificación de la región de AWS) en las herramientas de AWS para documentos de PowerShell para más información.

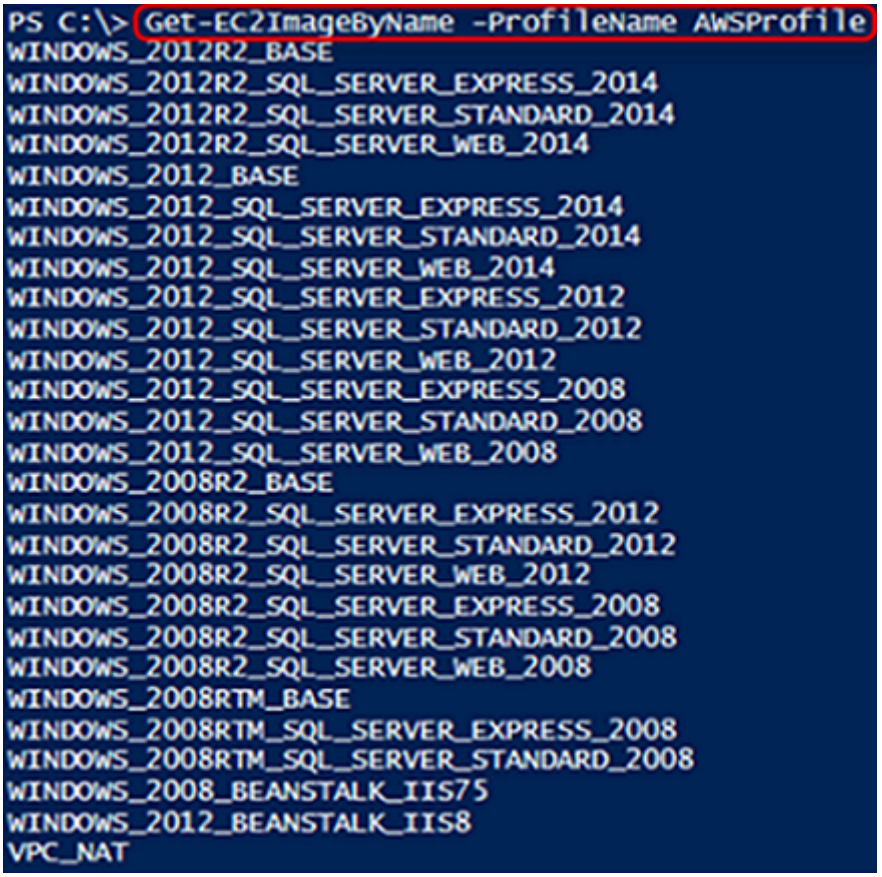
8. Para recuperar una lista de nombres de imagen de su suscripción de AWS, inicie PowerShell ISE e importe el módulo AWS de PowerShell. Para autenticar en AWS, reemplace `Get-AutomationPSCredential` en el entorno de ISE con `AWScred = Get-Credential`. Esta instrucción solicita las credenciales. Puede proporcionar el identificador de la clave de acceso como nombre de usuario y la clave de acceso secreta como contraseña.

PowerShell

```
#Sample to get the AWS VM available images
#Please provide the path where you have downloaded the AWS PowerShell module
Import-Module AWSPowerShell
$AwsRegion = "us-west-2"
```

```
  $AwsCred = Get-Credential
  $AwsAccessKeyId = $AwsCred.UserName
  $AwsSecretKey = $AwsCred.GetNetworkCredential().Password

  # Set up the environment to access AWS
  Set-AwsCredentials -AccessKey $AwsAccessKeyId -SecretKey $AwsSecretKey -StoreAs
AWSProfile
  Set-DefaultAWSRegion -Region $AwsRegion

  Get-EC2ImageByName -ProfileName AWSProfile
```

Se devuelve el siguiente resultado:



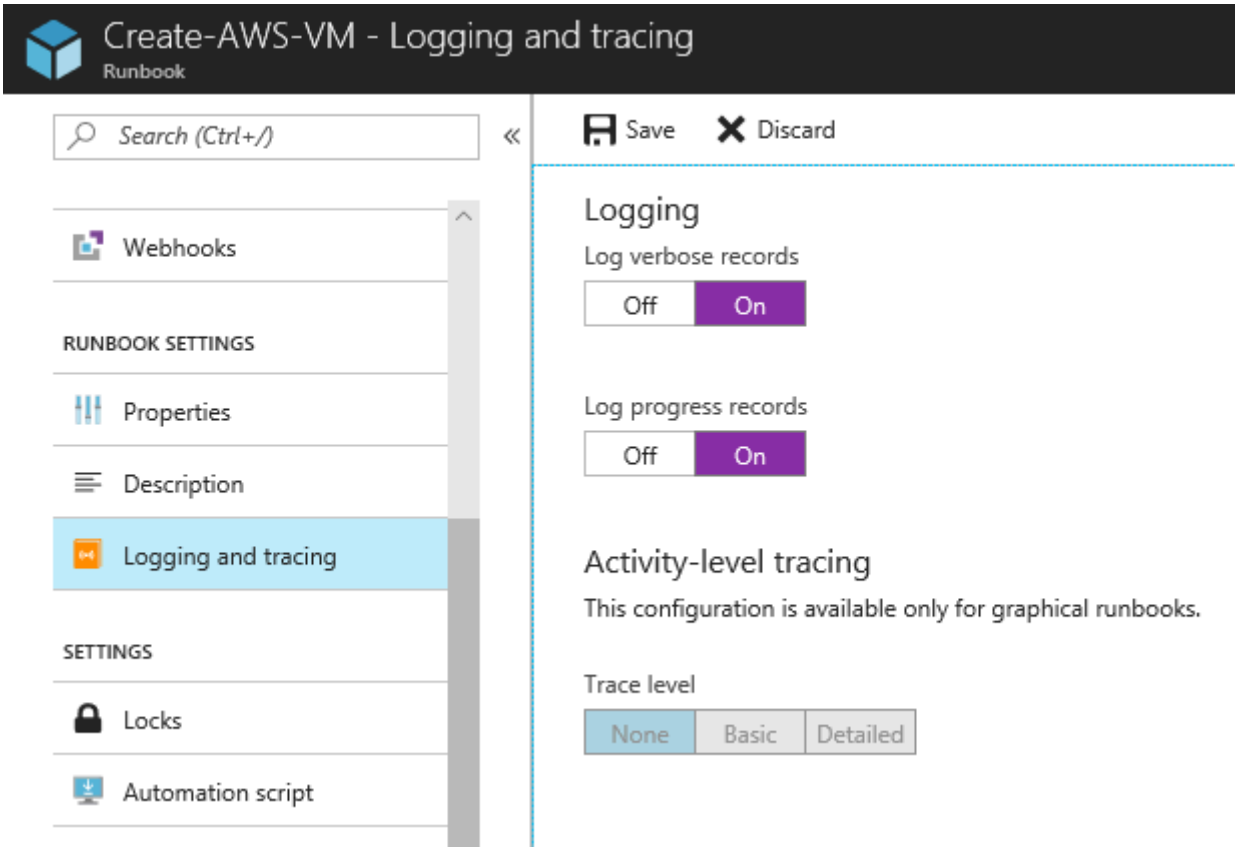
9. Copie y pegue el nombre de una de las imágenes en una variable de Automation que se indica en el runbook como `$InstanceType`. Dado que en este ejemplo utiliza la suscripción gratuita en capas de AWS, usará **t2.micro** en el ejemplo de runbook.

10. Guarde el runbook y, a continuación, haga clic en **Publish** para publicarlo y después en **Yes** cuando se le indique.

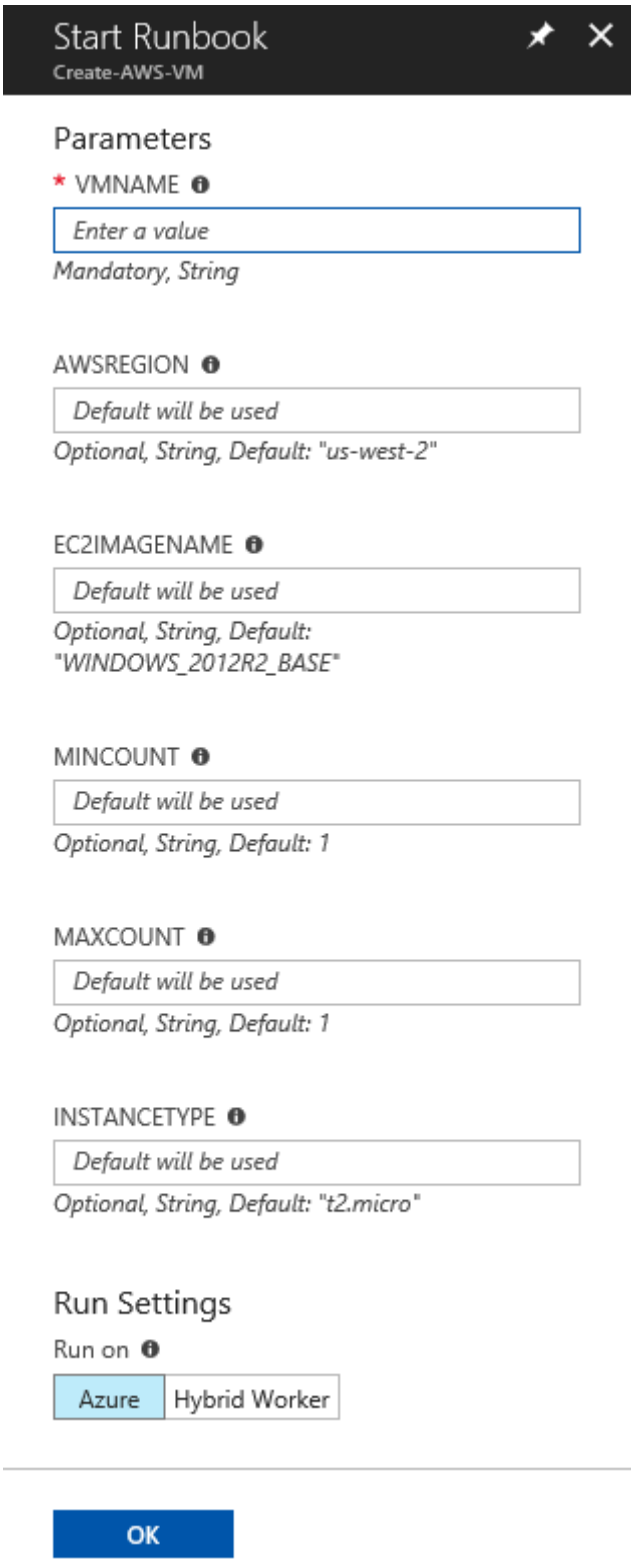
Prueba del runbook de la máquina virtual de AWS

1. Compruebe que el runbook crea un recurso denominado `AWScred` para su autenticación en AWS, o bien actualice el script para que haga referencia al nombre del recurso de credenciales.

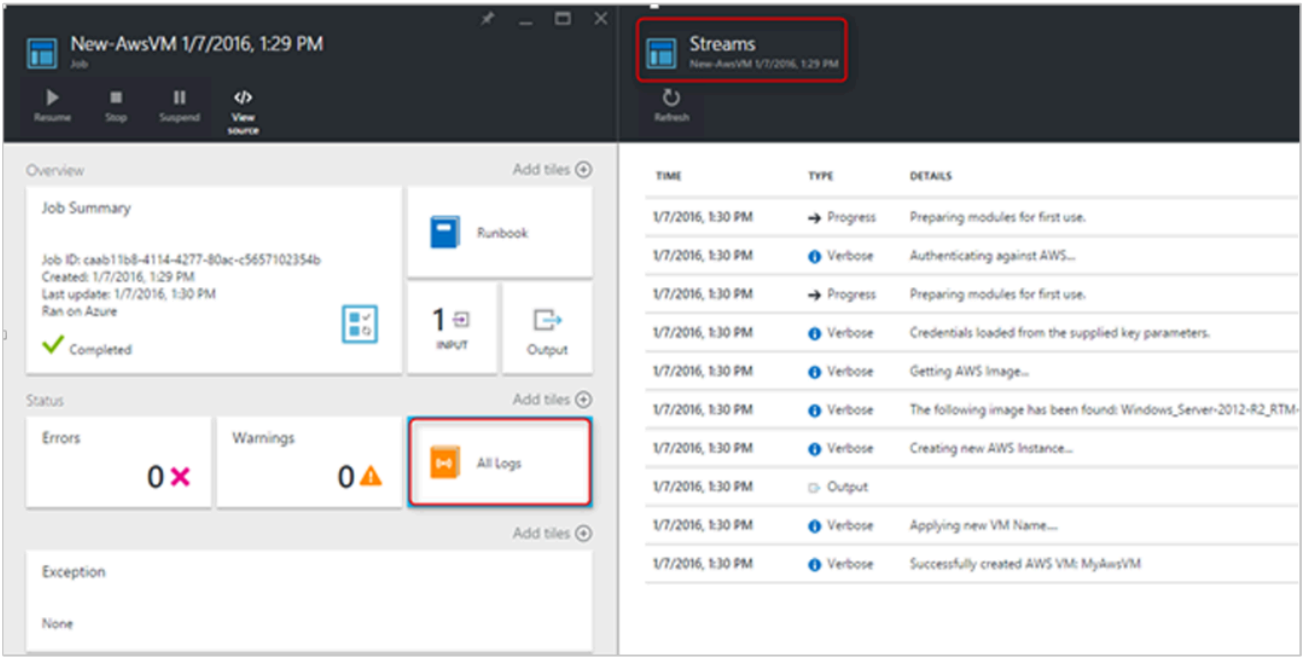
2. Verifique su nuevo runbook y asegúrese de que los valores de todos los parámetros se hayan actualizado según sea necesario. Asegúrese de que el módulo de AWS para PowerShell se haya importado en Azure Automation.
3. En Azure Automation, establezca **Registrar registros detallados** y, opcionalmente, **Registrar registros de progreso** de la operación **Registro y seguimiento** del runbook en **Activado**.



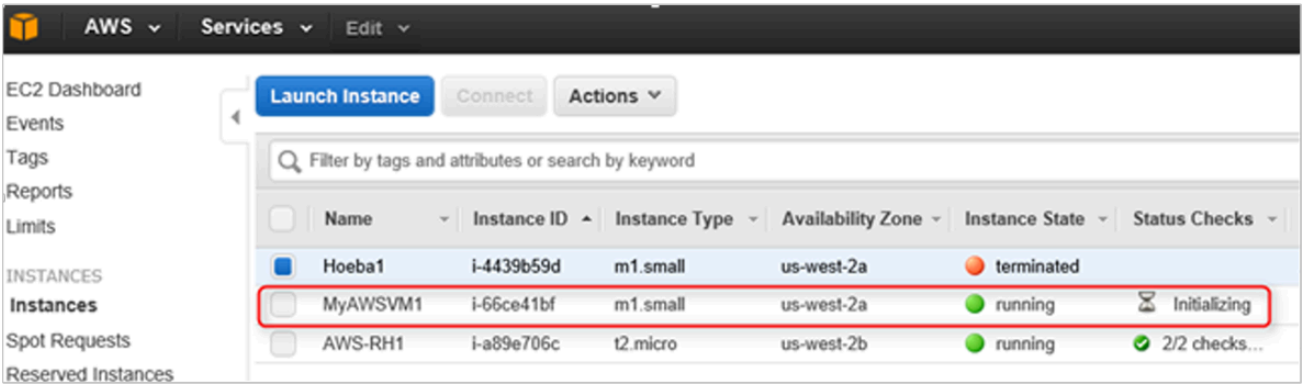
4. Para iniciar el runbook, haga clic en **Iniciar** y, después, en **Aceptar** cuando se abra el panel Iniciar runbook.
5. En el panel Iniciar runbook, especifique un nombre para la máquina virtual. Acepte los valores predeterminados de los demás parámetros que configuró previamente en el script. Haga clic en **Aceptar** para iniciar el trabajo del runbook.



6. Se abre un panel de trabajo para el trabajo del runbook que ha creado. Cierre este panel.
7. Puede ver el progreso del trabajo y los flujos de salida seleccionando **Todos los registros** en el panel Trabajo del runbook.



8. Para confirmar que la VM se está aprovisionando, inicie sesión en la Consola de administración de AWS si todavía no ha iniciado sesión.



Pasos siguientes

- Para averiguar qué runbooks se admiten, consulte Tipos de runbooks de Azure Automation
- Para trabajar con runbooks, consulte Administración de runbooks en Azure Automation.
- Para más información sobre PowerShell, consulte PowerShell Docs.
- Con respecto a la compatibilidad con scripts, consulte Compatibilidad nativa con scripts de PowerShell en Azure Automation.
- Para obtener una referencia de los cmdlets de PowerShell, consulte Az.Automation.

Last updated on 22/07/2026

Desplegar una plantilla de Azure Resource Manager en un libro de ejecución de PowerShell Automation

Puede escribir un runbook de PowerShell de Automation que implemente un recurso de Azure mediante una plantilla de Azure Resource Manager. Las plantillas le permiten usar Azure Automation para automatizar la implementación de los recursos de Azure. Puede mantener las plantillas de Resource Manager en una ubicación central segura como Azure Storage.

En este artículo, se crea un runbook de PowerShell que usa una plantilla de Resource Manager almacenada en Azure Storage para implementar una nueva cuenta de Azure Storage.

Si no tiene una suscripción a Azure, cree una cuenta gratuita antes de empezar.

Prerrequisitos

- Una cuenta de Azure Automation con al menos una identidad administrada asignada por el usuario. Para más información, consulte Uso de una identidad administrada asignada por el usuario para una cuenta de Azure Automation.
- Módulos de Az: `Az.Accounts`, `Az.ManagedServiceIdentity`, `Az.Resources` y `Az.Storage`. importados en la cuenta de Automation. Para más información, consulte Importación de módulos Az.
- Cuenta de Azure Storage donde se va a almacenar la plantilla de Resource Manager.
- Azure PowerShell instalado en una máquina local. Consulte Instalación del módulo de Azure PowerShell para información sobre cómo obtener Azure PowerShell. También necesitará el módulo Az.ManagedServiceIdentity. `Az.ManagedServiceIdentity` es un módulo en versión preliminar y no está instalado como parte del módulo Az. Para instalarlo, ejecute `Install-Module -Name Az.ManagedServiceIdentity`

Asignación de permisos a identidades administradas

Asigne permisos a las identidades administradas para realizar las tareas relacionadas con el almacenamiento en el Runbook.

1. Inicie sesión en Azure de forma interactiva mediante el cmdlet Connect-AzAccount y siga las instrucciones.

PowerShell

```
# Sign in to your Azure subscription
$sub = Get-AzSubscription -ErrorAction SilentlyContinue
if(-not($sub))
{
    Connect-AzAccount
}

# If you have multiple subscriptions, set the one to use
# Select-AzSubscription -SubscriptionId <SUBSCRIPTIONID>
```

2. Proporcione un valor adecuado para las variables siguientes y, a continuación, ejecute el script.

PowerShell

```
$resourceGroup = "resourceGroup"
$automationAccount = "automationAccount"
$storageAccount = "storageAccount"
$userAssignedManagedIdentity = "userAssignedManagedIdentity"
$storageTemplate = "path\storageTemplate.json"
$runbookScript = "path\runbookScript.ps1"
```

3. Asigne el rol `reader` a la identidad administrada asignada por el sistema para ejecutar el cmdlet `Get-AzUserAssignedIdentity`.

PowerShell

```
$SAMI = (Get-AzAutomationAccount -ResourceGroupName $resourceGroup -Name
$automationAccount).Identity.PrincipalId
New-AzRoleAssignment `
    -ObjectId $SAMI `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName "Reader"
```

4. Asigne el rol `Storage Account Contributor` a la identidad administrada asignada por el sistema para realizar acciones en la cuenta de almacenamiento.

PowerShell

```
$UAMI_ID = (Get-AzUserAssignedIdentity -ResourceGroupName $resourceGroup -Name
$userAssignedManagedIdentity).PrincipalId
New-AzRoleAssignment `
    -ObjectId $UAMI_ID `
    -ResourceGroupName $resourceGroup `
    -RoleDefinitionName "Storage Account Contributor"
```

Creación de la plantilla de Resource Manager

En este ejemplo, se usa una plantilla de Resource Manager que implementa una nueva cuenta de Azure Storage. Cree un archivo local llamado `storageTemplate.json` y pegue el código siguiente:

JSON

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "storageAccountType": {
      "type": "string",
      "defaultValue": "Standard_LRS",
      "allowedValues": [
        "Standard_LRS",
        "Standard_GRS",
        "Standard_ZRS",
        "Premium_LRS"
      ],
      "metadata": {
        "description": "Storage Account type"
      }
    },
    "location": {
      "type": "string",
      "defaultValue": "[resourceGroup().location]",
      "metadata": {
        "description": "Location for all resources."
      }
    }
  },
  "variables": {
    "storageAccountName": "[concat(uniquestring(resourceGroup().id), 'standardsa')]"
  },
  "resources": [
    {
      "type": "Microsoft.Storage/storageAccounts",
      "name": "[variables('storageAccountName')]",
      "apiVersion": "2018-02-01",
      "location": "[parameters('location')]",
      "sku": {
          "name": "[parameters('storageAccountType')]"
      },
      "kind": "Storage",
      "properties": {
      }
    }
  ],
  "outputs": {
```

```
        "storageAccountName": {
            "type": "string",
            "value": "[variables('storageAccountName')]"
        }
    }
}
```

Guardar la plantilla de Resource Manager en Azure Files

Usa PowerShell para crear una compartición de archivos en Azure y cargar `storageTemplate.json`. Para obtener instrucciones sobre cómo crear un recurso compartido de archivos y cargar un archivo en Azure Portal, consulte Introducción a Azure Files en Windows.

Ejecute los siguientes comandos para crear un recurso compartido de archivos y cargar la plantilla de Resource Manager en ese recurso compartido de archivos.

PowerShell

```
# Get the access key for your storage account
$key = Get-AzStorageAccountKey -ResourceGroupName $resourceGroup -Name
$storageAccount

# Create an Azure Storage context using the first access key
$context = New-AzStorageContext -StorageAccountName $storageAccount -
StorageAccountKey $key[0].value

# Create a file share named 'resource-templates' in your Azure Storage account
$fileShare = New-AzStorageShare -Name 'resource-templates' -Context $context

# Add the storageTemplate.json file to the new file share
Set-AzStorageFileContent -ShareName $fileShare.Name -Context $context -Source
$storageTemplate
```

Creación del script de runbook de PowerShell

Cree un script de PowerShell que obtenga el `storageTemplate.json` archivo de Azure Storage e implemente la plantilla para crear una nueva cuenta de Azure Storage. Cree un archivo local llamado `runbookScript.ps1` y pegue el código siguiente:

PowerShell

```
param (
    [Parameter(Mandatory=$true)]
    [string]
    $resourceGroup,
```

```
    [Parameter(Mandatory=$true)]
    [string]
    $storageAccount,

    [Parameter(Mandatory=$true)]
    [string]
    $storageAccountKey,

    [Parameter(Mandatory=$true)]
    [string]
    $storageFileName,

    [Parameter(Mandatory=$true)]
    [string]
    $userAssignedManagedIdentity
)

# Ensures you do not inherit an AzContext in your runbook
Disable-AzContextAutosave -Scope Process

# Connect to Azure with user-assigned managed identity
$AzureContext = (Connect-AzAccount -Identity).context
$identity = Get-AzUserAssignedIdentity -ResourceGroupName $resourceGroup `
    -Name $userAssignedManagedIdentity `
    -DefaultProfile $AzureContext
$AzureContext = (Connect-AzAccount -Identity -AccountId $identity.ClientId).context

# set and store context
$AzureContext = Set-AzContext -SubscriptionName $AzureContext.Subscription `
    -DefaultProfile $AzureContext

#Set the parameter values for the Resource Manager template
$Parameters = @{
    "storageAccountType"="Standard_LRS"
    }

# Create a new context
$Context = New-AzStorageContext -StorageAccountName $storageAccount -
StorageAccountKey $storageAccountKey

Get-AzStorageFileContent `
    -ShareName 'resource-templates' `
    -Context $Context `
    -path 'storageTemplate.json' `
    -Destination 'C:\Temp' -Force

$TemplateFile = Join-Path -Path 'C:\Temp' -ChildPath $storageFileName

# Deploy the storage account
New-AzResourceGroupDeployment `
    -ResourceGroupName $resourceGroup `
    -TemplateFile $TemplateFile `
    -TemplateParameterObject $Parameters
```

Importa y publica el runbook en tu cuenta de Azure Automation

Use PowerShell para importar el runbook en su cuenta de Azure Automation y, luego, publique el runbook. Para obtener información sobre cómo importar y publicar runbooks en Azure Portal, consulte Administración de runbooks en Azure Automation.

Para importar `runbookScript.ps1` en tu cuenta de Automation como un runbook de PowerShell, ejecute los siguientes comandos de PowerShell.

PowerShell

```
$importParams = @{
    Path = $runbookScript
    ResourceGroupName = $resourceGroup
    AutomationAccountName = $automationAccount
    Type = "PowerShell"
}
Import-AzAutomationRunbook @importParams

# Publish the runbook
$publishParams = @{
    ResourceGroupName = $resourceGroup
    AutomationAccountName = $automationAccount
    Name = "runbookScript"
}
Publish-AzAutomationRunbook @publishParams
```

Inicio del runbook

Ahora iniciaremos el runbook llamando al cmdlet Start-AzAutomationRunbook. Para obtener información sobre cómo iniciar un runbook en Azure Portal, consulte Inicio de un runbook en Azure Automation.

Ejecute los comandos siguientes en la consola de PowerShell:

PowerShell

```
# Set up the parameters for the runbook
$runbookParams = @{
    resourceGroup = $resourceGroup
    storageAccount = $storageAccount
    storageAccountKey = $key[0].Value # We got this key earlier
    storageFileName = "storageTemplate.json"
    userAssignedManagedIdentity = $userAssignedManagedIdentity
}
```

```
# Set up parameters for the Start-AzAutomationRunbook cmdlet
$startParams = @{
    resourceGroup = $resourceGroup
    AutomationAccountName = $automationAccount
    Name = "runbookScript"
    Parameters = $runbookParams
}

# Start the runbook
$job = Start-AzAutomationRunbook @startParams
```

Después de ejecutar el runbook, puede comprobar su estado recuperando el valor de propiedad del objeto de trabajo `$job.Status`.

El runbook obtiene la plantilla de Resource Manager y la usa para implementar una nueva cuenta de Azure Storage. Para ver que se creó la nueva cuenta de almacenamiento, ejecute el siguiente comando:

PowerShell

```
Get-AzStorageAccount
```

Pasos siguientes

- Para más información sobre las plantillas de Resource Manager, consulte Introducción a Azure Resource Manager.
- Para empezar a trabajar con Azure Storage, consulte Introducción a Azure Storage.
- Para encontrar otros runbooks útiles de Azure Automation, vea Utilizar runbooks y módulos en Azure Automation.

Last updated on 17/11/2025

Tutorial: Integración de Azure Automation con Event Grid y Microsoft Teams

Artículo • 16/03/2023

En este tutorial, aprenderá a:

- ✓ Importe un runbook de ejemplo de Event Grid.
- ✓ Cree un webhook de Microsoft Teams opcional.
- ✓ Cree un webhook para el runbook.
- ✓ Cree una suscripción de Event Grid.
- ✓ Cree una máquina virtual que desencadene el runbook.

Si no tiene una suscripción a Azure, cree una cuenta gratuita antes de empezar.

Requisitos previos

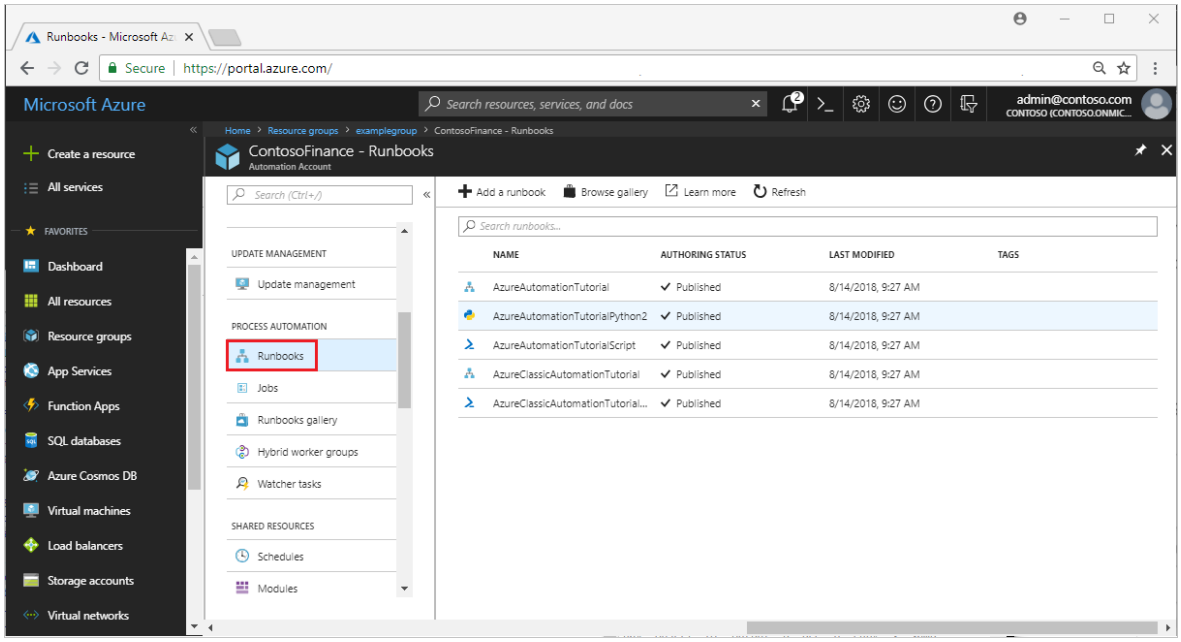
> **Importante**
>
> El uso de esta característica de Azure desde PowerShell requiere que tenga el módulo `AzureRM` instalado. Se trata de un módulo anterior que solo está disponible para Windows PowerShell 5.1 que ya no obtiene nuevas características. Los módulos `Az` y `AzureRM` **no** son compatibles cuando se instalan para las mismas versiones de PowerShell. Si necesita ambas versiones:
>
> 1. **Desinstale el módulo Az** desde una sesión de PowerShell 5.1.
> 2. **Instale el módulo AzureRM** desde una sesión de PowerShell 5.1.
> 3. **Descargue e instale PowerShell Core 6.x o posterior**.
> 4. **Instale el módulo Az** en una sesión de PowerShell Core.

Para completar este tutorial, se requiere una cuenta de Azure Automation para contener el runbook que se desencadena desde la suscripción de Azure Event Grid.

- El módulo `AzureRM.Tags` tiene que cargarse en su cuenta de Automation, consulte Importación de módulos en Azure Automation para aprender a importar módulos en Azure Automation.

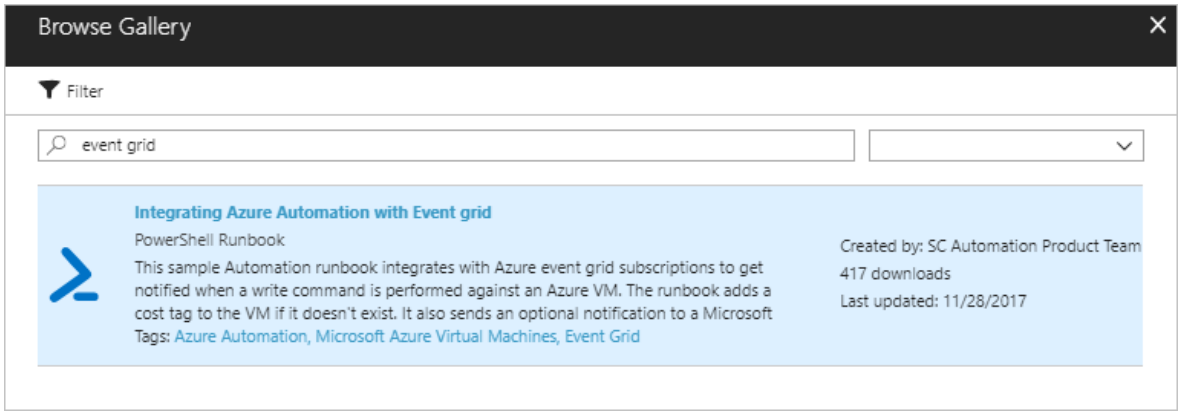
Importación de un runbook de ejemplo de Event Grid

1. Seleccione la cuenta de Automation y la página **Runbooks**.



2. Seleccione el botón **Examinar galería**.

3. Busque **Event Grid** y seleccione **Integración de Azure Automation con Event Grid**.



4. Seleccione **Importar** y asígnele el nombre **Watch-VMWrite**.

5. Después de la importación, seleccione **Editar** para ver el origen de runbook.

6. Actualice la línea 74 en el script para que utilice `Tag` en lugar de `Tags`.

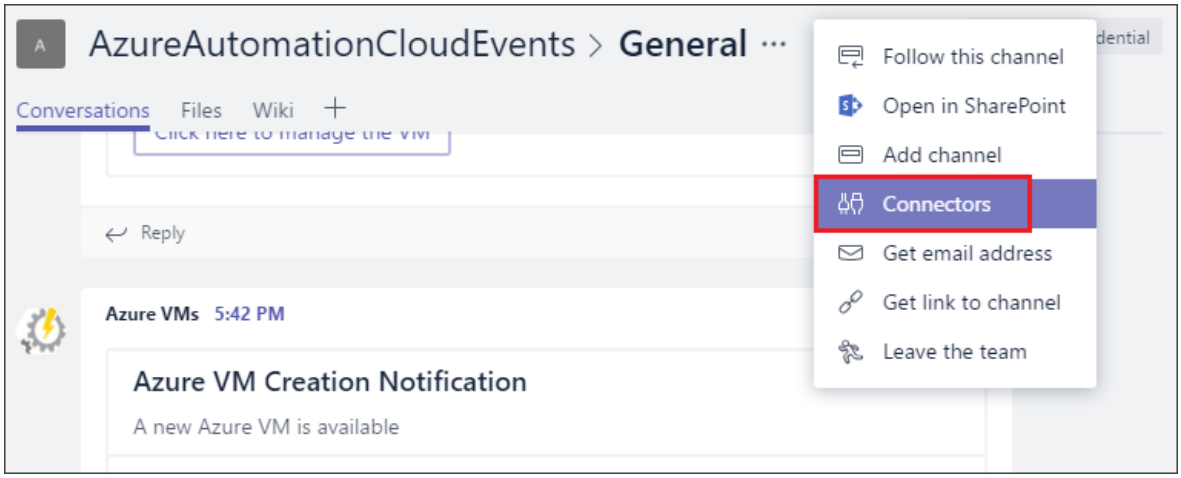
PowerShell

```
Update-AzureRmVM -ResourceGroupName $VMResourceGroup -VM $VM -Tag $Tag
| Write-Verbose
```

7. Seleccione el botón **Publicar**.

Creación de un webhook de Microsoft Teams opcional

1. En Microsoft Teams, seleccione **Más opciones** junto al nombre del canal y, luego, seleccione **Conectores**.



2. Desplácese por la lista de conectores hasta **Webhook de entrada** y seleccione **Agregar**.

3. Escriba **AzureAutomationIntegration** como el nombre y seleccione **Crear**.

4. Copie la dirección URL del webhook en el Portapapeles y guárdela. La dirección URL del webhook se usa para enviar información a Microsoft Teams.

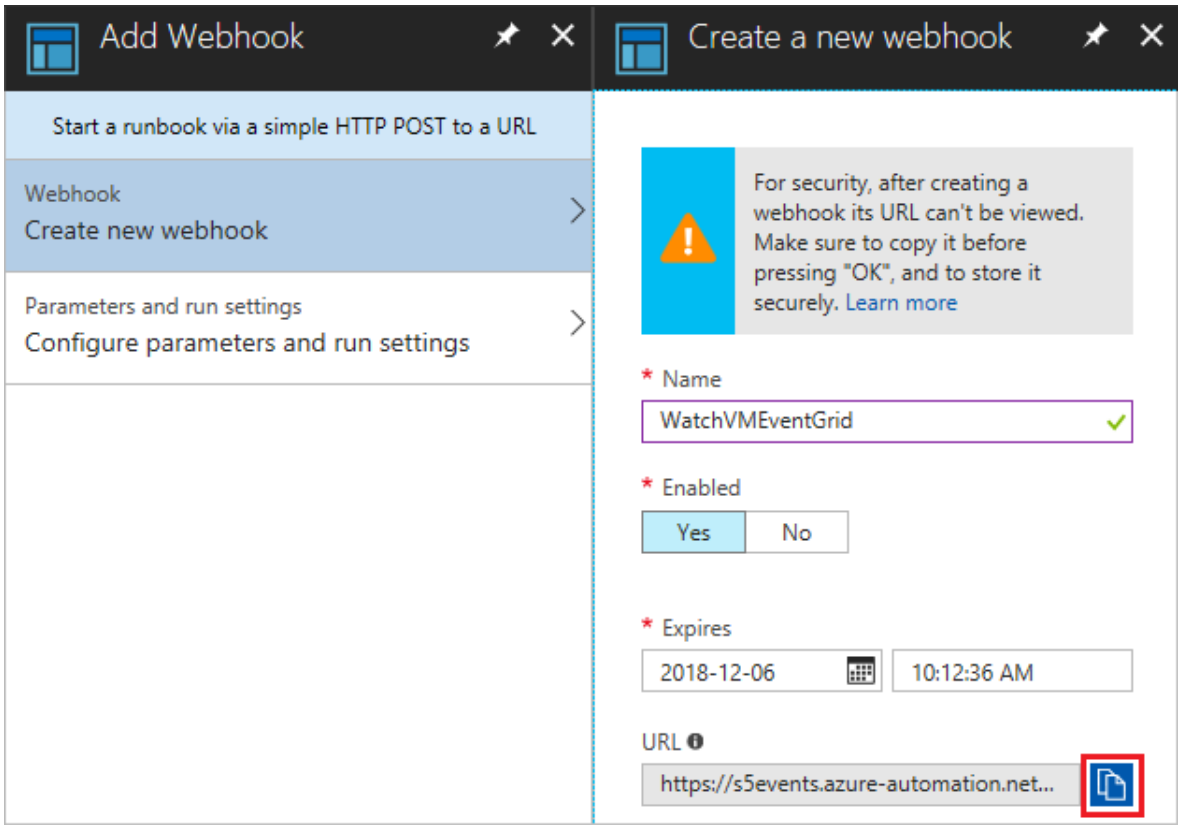
5. Seleccione **Hecho** para guardar el webhook.

Creación de un webhook para el runbook

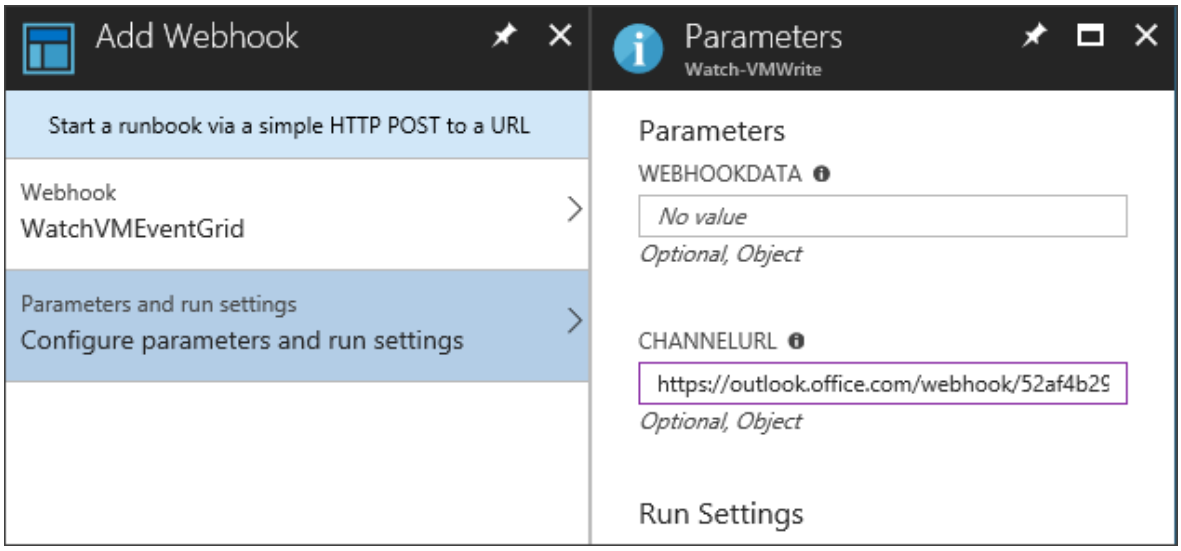
1. Abra el runbook Watch-VMWrite.

2. Seleccione **Webhooks** y el botón **Agregar webhook**.

3. Escriba **WatchVMEventGrid** como el nombre. Copie la dirección URL en el Portapapeles y guárdelo.



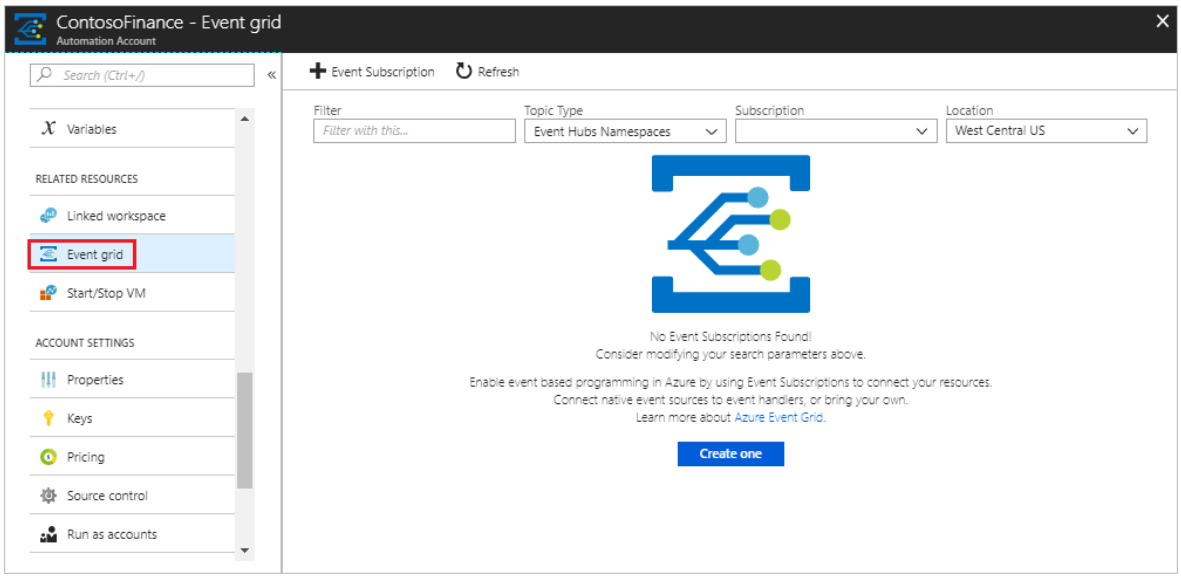
4. Seleccione **Configurar parámetros y ejecutar configuraciones** y escriba la dirección URL del webhook de Microsoft Teams para **CHANNELURL**. Deje en blanco **WEBHOOKDATA**.



5. Seleccione **Crear** para crear el webhook del runbook de Automation.

Creación de una suscripción de Event Grid

1. En la página de información general de la **cuenta de Automation**, seleccione **Event Grid**.



2. Haga clic en **+Suscripción de eventos**.

3. Configure la suscripción con la información siguiente:

   a. En **Tipo de tema**, seleccione **Suscripciones de Azure**.

   b. Desactive la casilla **Suscribirse a todos los tipos de evento**.

   c. Escriba **AzureAutomation** como el nombre.

   d. En la lista desplegable **Tipos de evento definidos**, desactive todas las opciones excepto **Resource Write Success** (Escritura de recursos correcta).

   > ⓘ **Nota**
   >
   > Azure Resource Manager no distingue actualmente entre Crear y Actualizar, por lo que la implementación de este tutorial para todos los eventos Microsoft.Resources.ResourceWriteSuccess en su suscripción de Azure podría dar lugar a un gran volumen de llamadas.

   e. En **Tipo de punto de conexión**, seleccione **Webhook**.

   f. Haga clic en **Seleccionar un punto de conexión**. En la página **Seleccionar webhook** que se abre, pegue la dirección URL del webhook que creó para el runbook Watch-VMWrite.

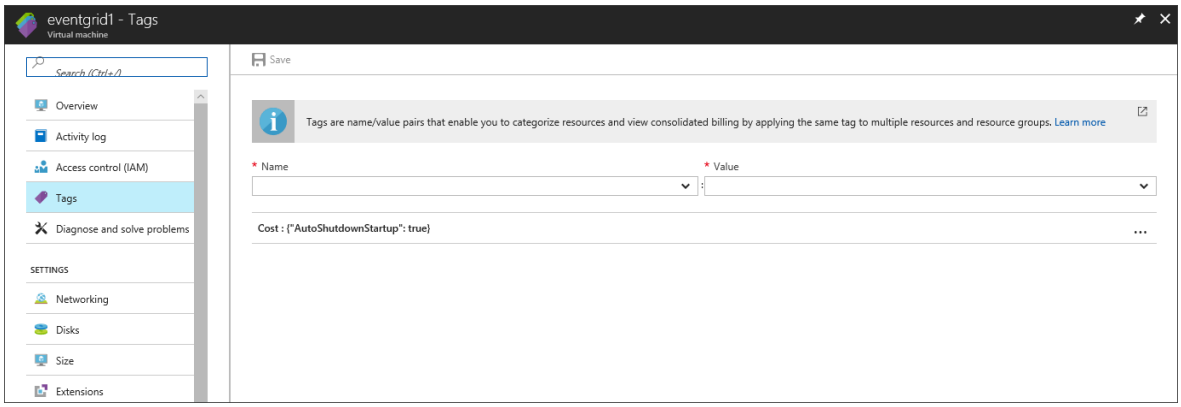
   g. En **FILTROS**, escriba la suscripción y el grupo de recursos en los que desea buscar las nuevas máquinas virtuales creadas. Debe verse de una manera similar a: `/subscriptions/<subscription-id>/resourcegroups/<resource-group-name>/providers/Microsoft.Compute/virtualMachines`

4. Seleccione **Crear** para guardar la suscripción de Event Grid.

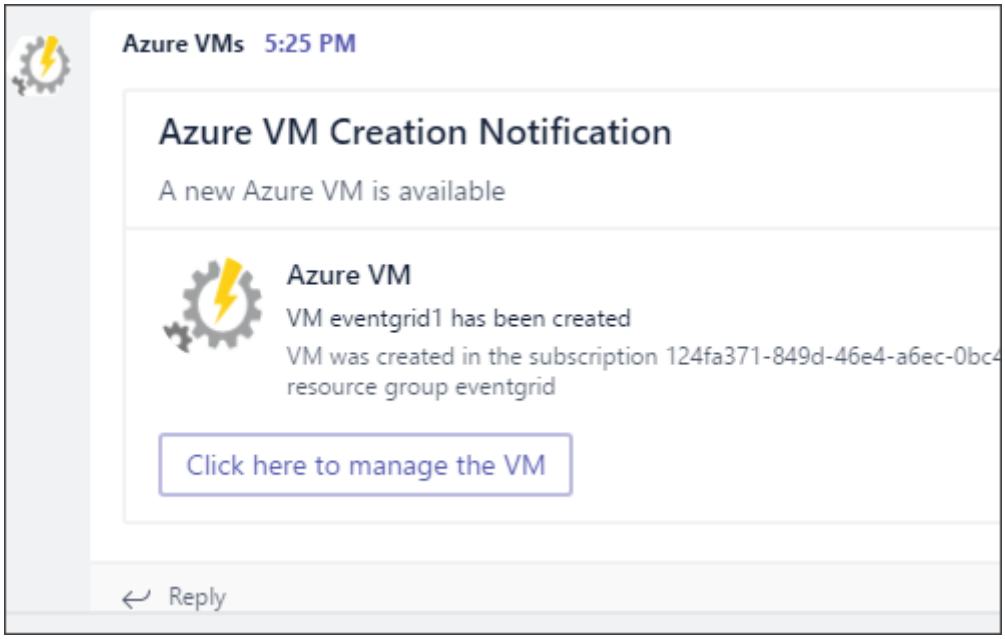
Creación de una máquina virtual que desencadena el runbook

1. Cree una máquina virtual nueva en el grupo de recursos que especificó en el filtro de prefijo de la suscripción de Event Grid.

2. Se debe llamar al runbook Watch-VMWrite y se debe agregar una etiqueta nueva a la máquina virtual.



3. Se envía un mensaje nuevo al canal de Microsoft Teams.



Pasos siguientes

En este tutorial, se configura la integración entre Event Grid y Automation. Ha aprendido a:

- ✔ Importe un runbook de ejemplo de Event Grid.

- ✓ Cree un webhook de Microsoft Teams opcional.
- ✓ Cree un webhook para el runbook.
- ✓ Cree una suscripción de Event Grid.
- ✓ Cree una máquina virtual que desencadene el runbook.

Creación y enrutamiento de eventos personalizados con Event Grid

Inicio y detención de entorno de ejecución de integración de Azure-SSIS mediante una programación

SE APLICA A: Azure Data Factory Azure Synapse Analytics

> **Sugerencia**
>
> Data Factory en Microsoft Fabric es la próxima generación de Azure Data Factory, con una arquitectura más sencilla, inteligencia artificial integrada y nuevas características. Si no está familiarizado con la integración de datos, comience con Fabric Data Factory. Las cargas de trabajo de ADF existentes pueden actualizarse a Fabric para acceder a nuevas funcionalidades en ciencia de datos, análisis en tiempo real e informes.
>
> - Iniciar una prueba gratuita de Fabric.
> - Actualización de Azure Data Factory a Data Factory en Microsoft Fabric

En este artículo se describe cómo programar el inicio y la detención de un entorno de ejecución de integración (IR) de Azure-SQL Server Integration Services (SSIS) mediante Azure Data Factory y Azure Synapse Analytics. Un Azure-SSIS IR es un recurso de proceso dedicado para ejecutar paquetes SSIS.

Un costo se asocia con la ejecución de un Azure-SSIS IR. Normalmente es preferible ejecutar su IR solo cuando haya que ejecutar paquetes de SSIS en Azure y detenerlo cuando ya no se necesite. Puede usar Data Factory, la página del Azure Portal para las canalizaciones de Azure Synapse Analytics o Azure PowerShell para iniciar o detener manualmente el IR.

Como alternativa, puede crear actividades web en Data Factory o Canalizaciones de Azure Synapse Analytics para iniciar y detener su entorno de ejecución de integración según una programación. Por ejemplo, puede iniciarla por la mañana antes de ejecutar las cargas de trabajo de extracción, transformación y carga de datos diarias y detenerla por la tarde después de que se realicen las cargas de trabajo.

También puede encadenar una actividad de Ejecutar paquete SSIS entre dos actividades web que inicien y detengan su IR. Después, el entorno de ejecución de integración se iniciará y se detendrá a petición, antes o después de la ejecución del paquete. Para obtener más información

sobre la actividad Ejecutar paquete SSIS, consulte Ejecución de un paquete SSIS con la actividad Ejecutar paquete SSIS en Azure Portal.

> ⓘ **Nota:**
>
> Se recomienda usar el módulo de PowerShell de Azure Az para interactuar con Azure. Para comenzar, consulte **Instalar Azure PowerShell**. Para más información sobre cómo migrar al módulo Az PowerShell, consulte **Migración de Azure PowerShell de AzureRM a Az**.

Requisitos previos

Para implementar este tutorial, necesita:

- Una instancia de Azure Data Factory. Si no tiene uno aprovisionado, siga los pasos descritos en Inicio rápido: Creación de una factoría de datos mediante Azure Portal y Azure Data Factory Studio.
- Un Azure-SSIS IR. Si no tiene uno aprovisionado, siga las instrucciones de Aprovisionamiento del entorno de ejecución de integración de SSIS de Azure en Azure Data Factory.

Creación y programación de canalizaciones de Azure Data Factory que inician o detienen un Azure-SSIS IR

> ⓘ **Nota:**
>
> Esta sección no es compatible con Azure-SSIS en Azure Synapse Analytics con la**protección contra la filtración de datos habilitada**.

En esta sección se muestra cómo usar las actividades web en las canalizaciones de Data Factory para iniciar y detener Azure-SSIS IR al programar o iniciar y detener a petición. Creará tres canalizaciones:

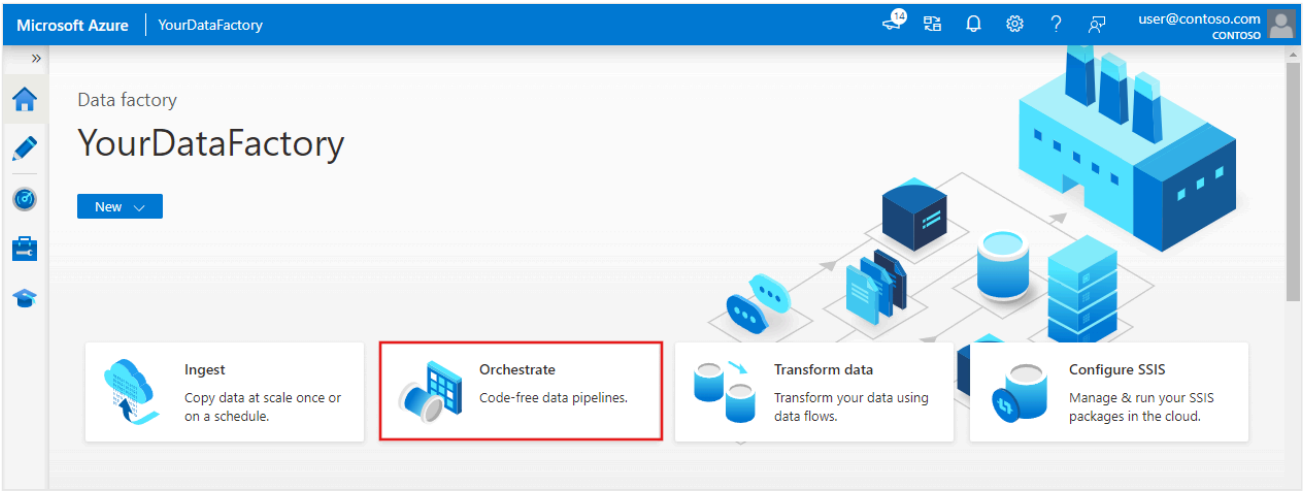
- La primera canalización contiene una actividad web que inicia su Azure-SSIS IR.
- La segunda canalización contiene una actividad web que detiene su Azure-SSIS IR.
- La tercera canalización contiene una actividad Ejecutar paquete SSIS encadenada entre dos actividades web que inician y detienen su Azure-SSIS IR.

Después de crear y probar esas canalizaciones, puede crear un desencadenador que defina una programación para ejecutar una canalización. Por ejemplo, puede crear dos desencadenadores. La primera está programada para ejecutarse diariamente a las 6:00 a. m. y está asociada a la primera canalización. La segunda está programada para ejecutarse diariamente a las 6 p. m. y está asociada a la segunda canalización. De este modo dispone de un período entre las 6 a. m. y las 6 p. m. todos los días en el que su IR se ejecuta, listo para ejecutar sus cargas de trabajo diarias de ETL.

Si crea un tercer desencadenador que está programado para ejecutarse diariamente a medianoche y está asociado a la tercera canalización, esa canalización se ejecutará a medianoche todos los días. Iniciará el entorno de ejecución de integración justo antes de la ejecución del paquete y, a continuación, ejecutará el paquete. Detendrá inmediatamente el entorno de ejecución de integración justo después de la ejecución del paquete, por lo que el entorno de ejecución de integración no se ejecutará de forma inactiva.

Creación de las canalizaciones

1. En la página principal de Azure Data Factory, seleccione **Orchestrate** (Organizar).



2. En el cuadro de herramientas **Actividades**, expanda el menú **General**, arrastre una actividad web y colóquela en la superficie del diseñador de canalizaciones. En la pestaña **General** de la ventana de propiedades, cambie el nombre de la actividad a**startMyIR**. Cambie a la pestaña **Configuración** y realice las acciones siguientes.

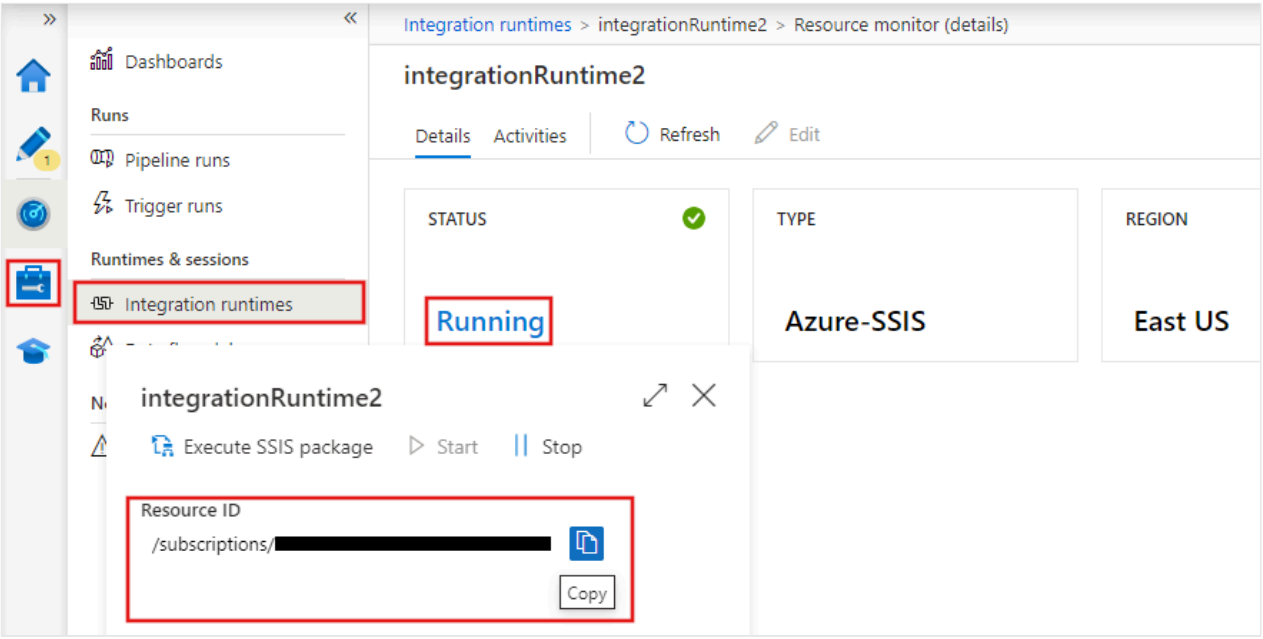
> ⓘ **Nota:**
>
> En el caso de Azure-SSIS en Azure Synapse Analytics, use la API de REST de Azure Synapse Analytics correspondiente para obtener el estado del entorno de ejecución

de integración, iniciar el entorno de ejecución de integración, y detener el entorno de ejecución de integración.

a. En **DIRECCIÓN URL**, escriba la siguiente dirección URL para la API de REST que inicia el Azure-SSIS IR. Reemplace `{subscriptionId}`, `{resourceGroupName}`, `{factoryName}` y `{integrationRuntimeName}` por los valores reales del entorno de ejecución de integración.

`https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.DataFactory/factories/{factoryName}/integrationRuntimes/{integrationRuntimeName}/start?api-version=2018-06-01`

De forma alternativa, puede copiar y pegar el identificador de recurso del IR de su página de supervisión en la interfaz de usuario o la aplicación de Data Factory para remplazar el siguiente elemento de la dirección URL anterior: `/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.DataFactory/factories/{factoryName}/integrationRuntimes/{integrationRuntimeName}`.

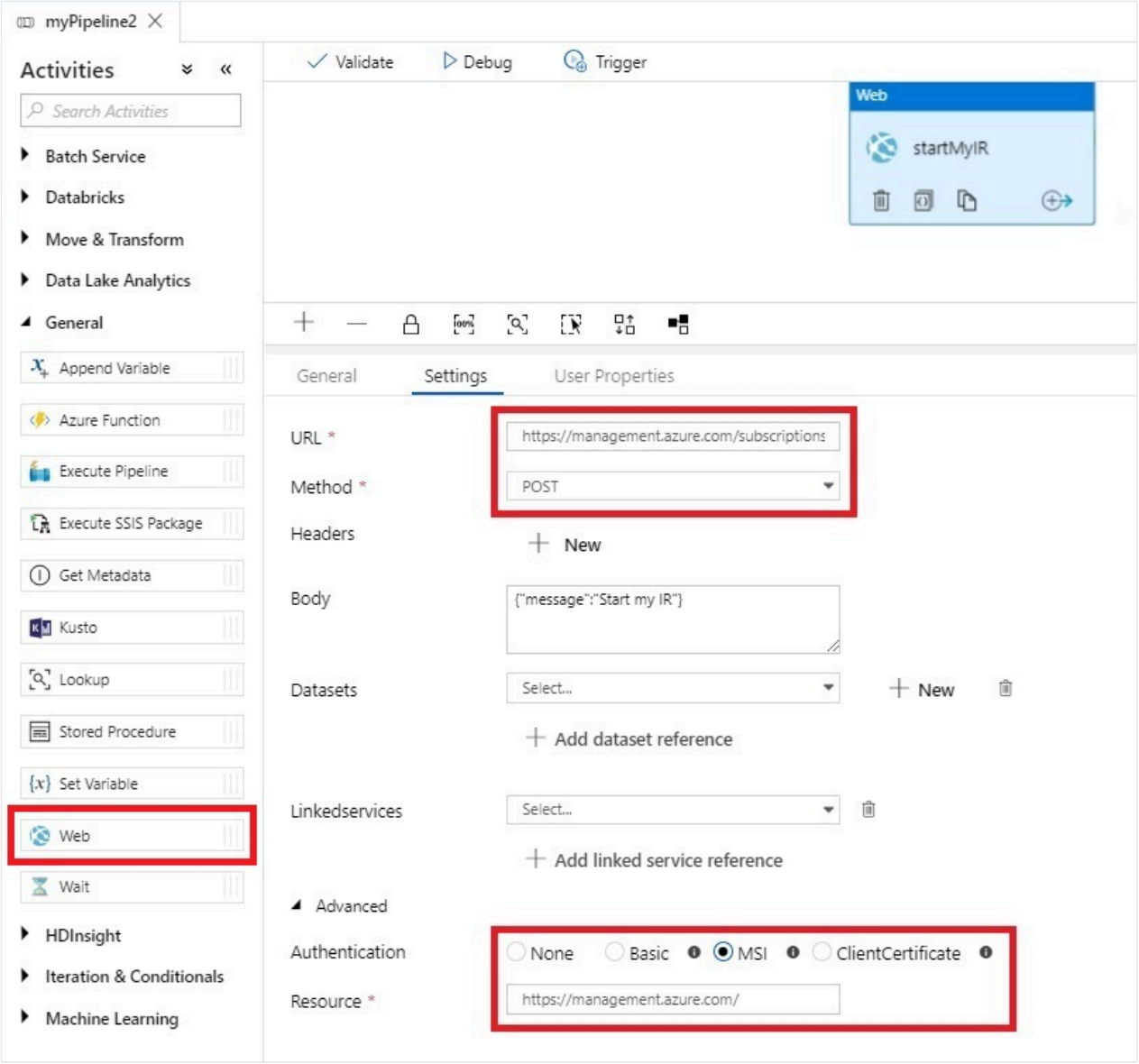


b. En **Method** (Método) seleccione **POST**.

c. En **Body** (Cuerpo), especifique `{"message":"Start my IR"}`.

d. En **Autenticación**, seleccione **Identidad administrada** para usar la identidad administrada por el sistema especificada para la factoría de datos. Para más información, consulte Identidad administrada para Azure Data Factory.

e. En **Recurso**, escriba `https://management.azure.com/`.



3. Clone la primera canalización para crear una segunda. Cambie el nombre de la actividad a **stopMyIR** y reemplace las siguientes propiedades:

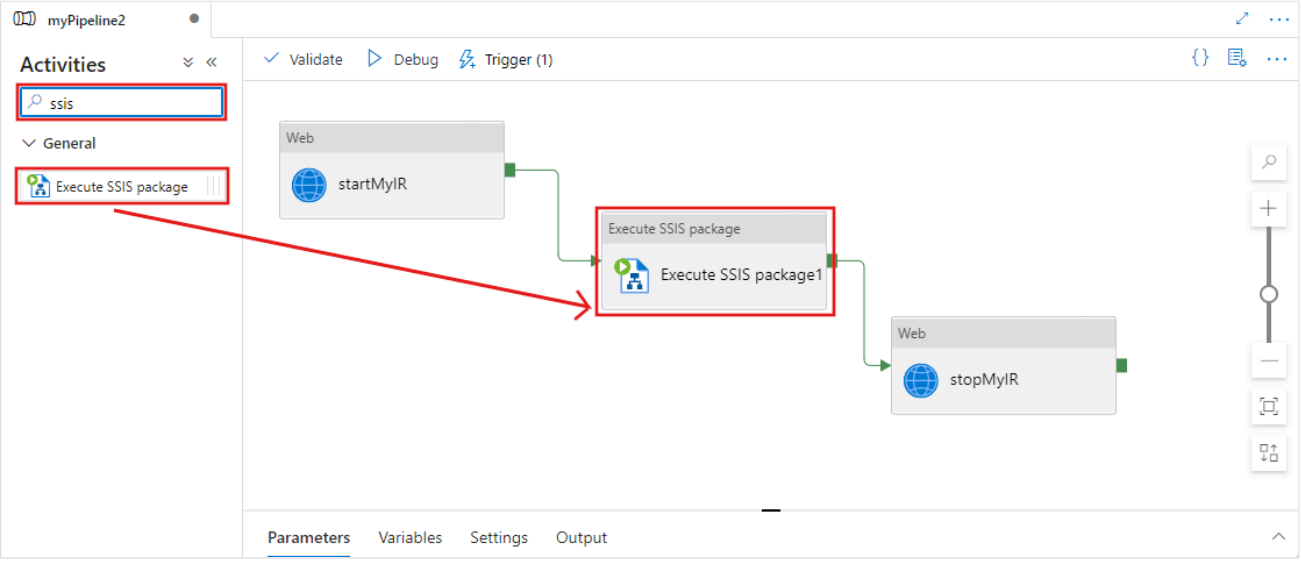
   a. En **DIRECCIÓN URL**, escriba la siguiente dirección URL para la API de REST que detiene el Azure-SSIS IR. Reemplace `{subscriptionId}`, `{resourceGroupName}`, `{factoryName}` y `{integrationRuntimeName}` por los valores reales del entorno de ejecución de integración.

   `https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.DataFactory/factories/{factoryName}/integrationRuntimes/{integrationRuntimeName}/stop?api-version=2018-06-01`.

   b. En **Body** (Cuerpo), especifique `{"message":"Stop my IR"}`.

4. Cree una tercera canalización. Arrastre una actividad **Ejecutar paquete SSIS** desde el cuadro de herramientas **Actividades** a la superficie del diseñador de canalizaciones. A continuación, configure la actividad siguiendo las instrucciones de Ejecución de un paquete SSIS con la actividad Ejecutar paquete SSIS en Azure Portal.

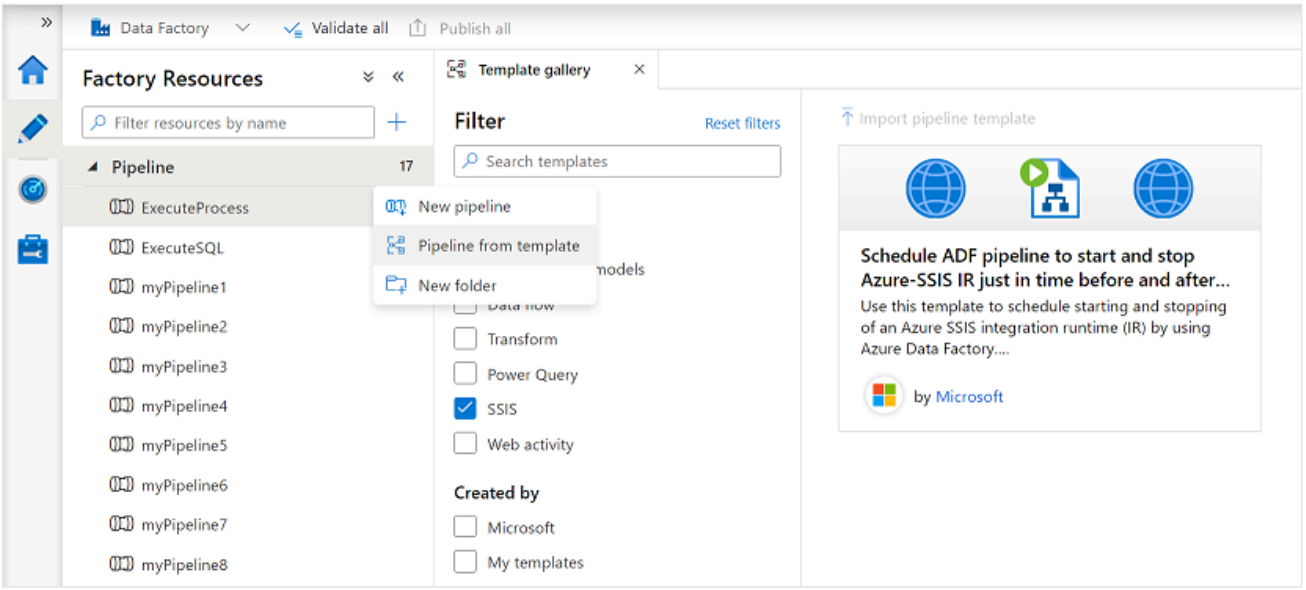
Encadene la actividad Ejecutar paquete SSIS entre dos actividades web que inicien y detengan el entorno de ejecución de integración, de forma similar a las actividades web de la primera y segunda canalización.



En lugar de crear manualmente la tercera canalización, también puede crearla automáticamente a partir de una plantilla:

a. Seleccione los puntos suspensivos (...) junto a **Canalización** para abrir un menú desplegable de acciones de canalización. A continuación, seleccione la acción **Canalización de la plantilla**.
b. Active la casilla **SSIS** en **Categoría**.
c. Seleccione la plantilla **Programación de la canalización de ADF para iniciar y detener Azure-SSIS IR en el momento preciso antes y después de ejecutar el paquete SSIS**.
d. En el menú desplegable **Azure-SSIS Integration Runtime**, seleccione el entorno de ejecución de integración.
e. Seleccione el botón **Usar esta plantilla**.

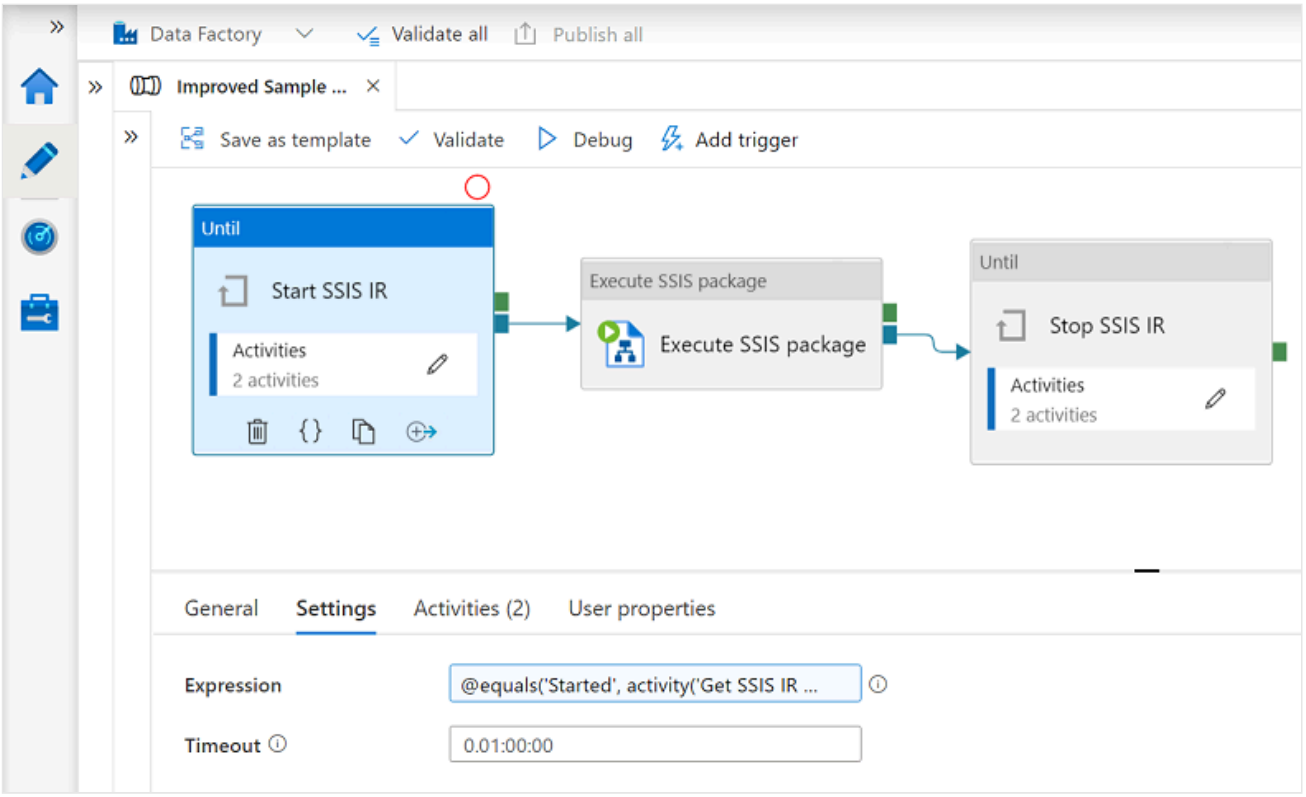
Después de crear la canalización automáticamente, solo queda el paquete SSIS para asignarlo a la actividad Ejecutar paquete SSIS.



5. Para que la tercera canalización sea más sólida, puede asegurarse de que se reintenten las actividades web para iniciar o detener el entorno de ejecución de integración si hay errores transitorios (por ejemplo, debido a la conectividad). También puede asegurarse de que esas actividades web solo se completen cuando el entorno de ejecución de integración se inicie o detenga realmente.

   Para ello, puede reemplazar cada actividad web por una actividad Until. La actividad Until contiene dos actividades web: una para iniciar y detener el IR y otra para comprobar el estado del IR. Vamos a llamar a las actividades Until *Start SSIS IR* (Iniciar SSIS IR) y *Stop SSIS IR* (Detener SSIS IR). La actividad Until *Iniciar SSIS IR* contiene las actividades web *Probar inicio de SSIS IR* y *Obtener estado de SSIS IR*. La actividad Until *Detener SSIS IR* contiene las actividades web *Probar detención de SSIS IR* y *Obtener estado de SSIS IR*.

   En la pestaña **Configuración** de las actividad Until *Iniciar SSIS IR*, en **Expresión**, escriba `@equals('Started', activity('Get SSIS IR Status').output.properties.state)`. En la pestaña **Configuración** de las actividad Until *Detener SSIS IR*, en **Expresión**, escriba `@equals('Stopped', activity('Get SSIS IR Status').output.properties.state)`.



Dentro de ambas actividades Until, las actividades web *Probar inicio de SSIS IR* y *Probar detención de SSIS IR* son similares a las actividades web de la primera y la segunda canalización. En la pestaña **Configuración** de las actividades web *Obtener estado de SSIS IR*, realice las siguientes acciones:

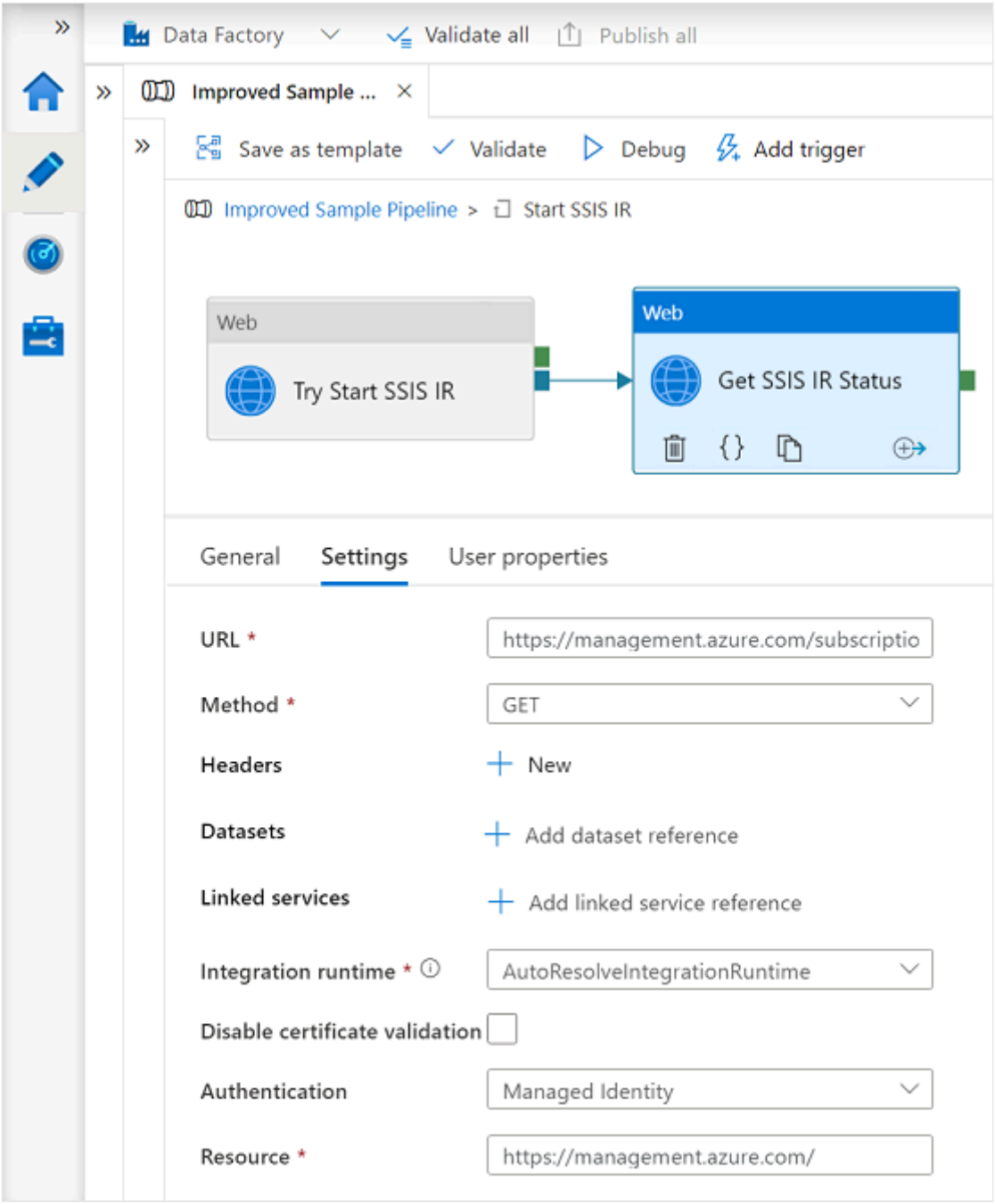
a. En **DIRECCIÓN URL**, escriba la siguiente dirección URL para la API de REST que obtiene el estado de Azure-SSIS IR. Reemplace `{subscriptionId}`, `{resourceGroupName}`, `{factoryName}` y `{integrationRuntimeName}` por los valores reales del entorno de ejecución de integración.

   `https://management.azure.com/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}/providers/Microsoft.DataFactory/factories/{factoryName}/integrationRuntimes/{integrationRuntimeName}?api-version=2018-06-01`

b. En **Método**, seleccione **GET**.

c. En **Autenticación**, seleccione **Identidad administrada** para usar la identidad administrada por el sistema especificada para la factoría de datos. Para más información, consulte Identidad administrada para Azure Data Factory.

d. En **Recurso**, escriba `https://management.azure.com/`.



6. Asigne a la identidad administrada para factoría de datos un rol de **Colaborador** para sí misma, de manera que las actividades web de las canalizaciones puedan llamar a la API de REST para iniciar o detener las instancias de Azure-SSIS IR aprovisionadas en ella:

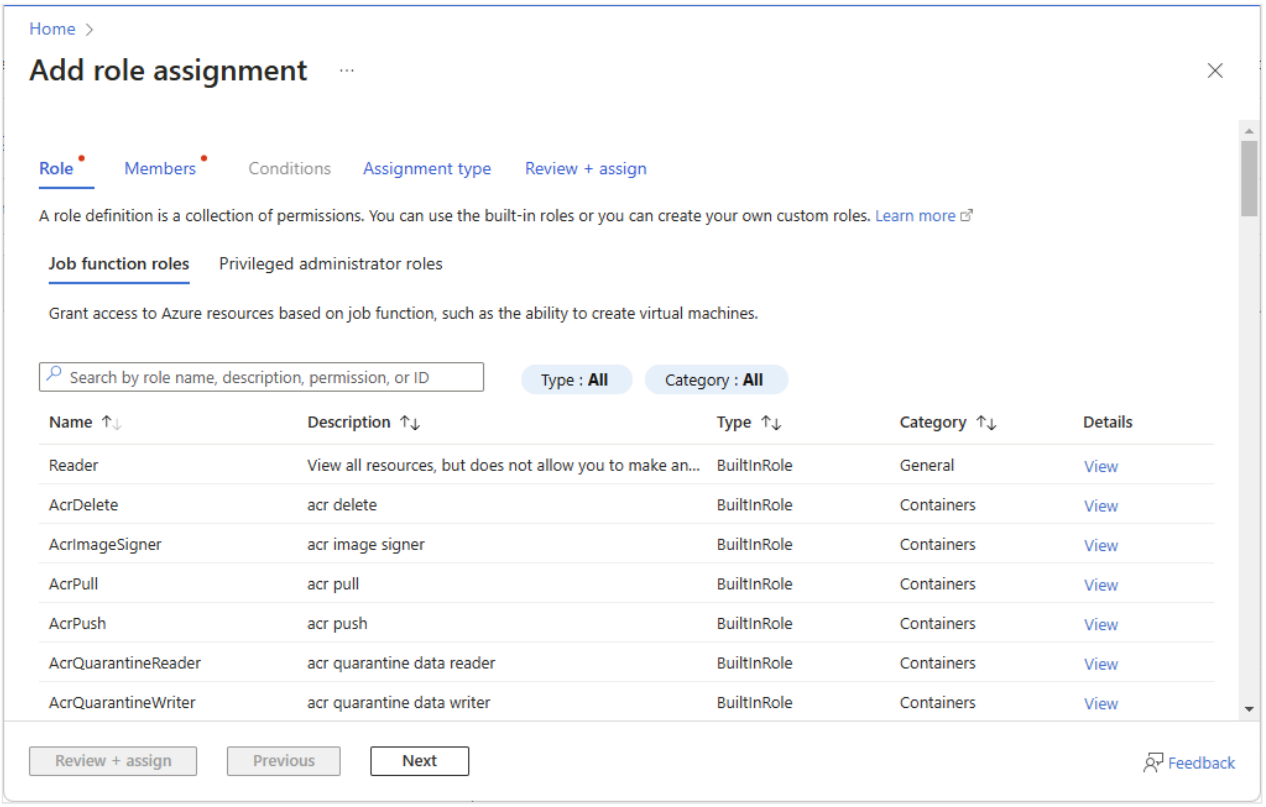
   a. En la página de Data Factory del Azure Portal, seleccione **Control de acceso (IAM)**.

   b. Seleccione **Agregar**>**Agregar asignación de roles** para abrir la página **Agregar asignación de roles**.

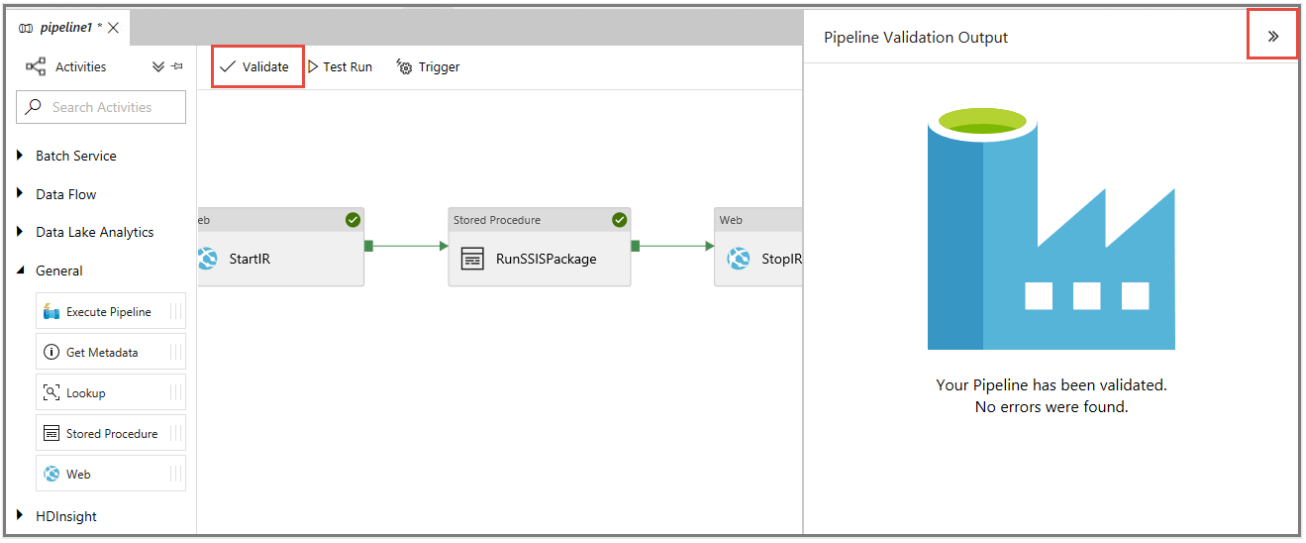
   c. Asigne el siguiente rol. Para acceder a los pasos detallados, consulte Asignación de roles de Azure mediante Azure Portal.

Expandir tabla

Configuración	Value
Role	Colaborador
Asignar acceso a	Usuario, grupo o entidad de servicio
Members	Nombre de usuario de Data Factory

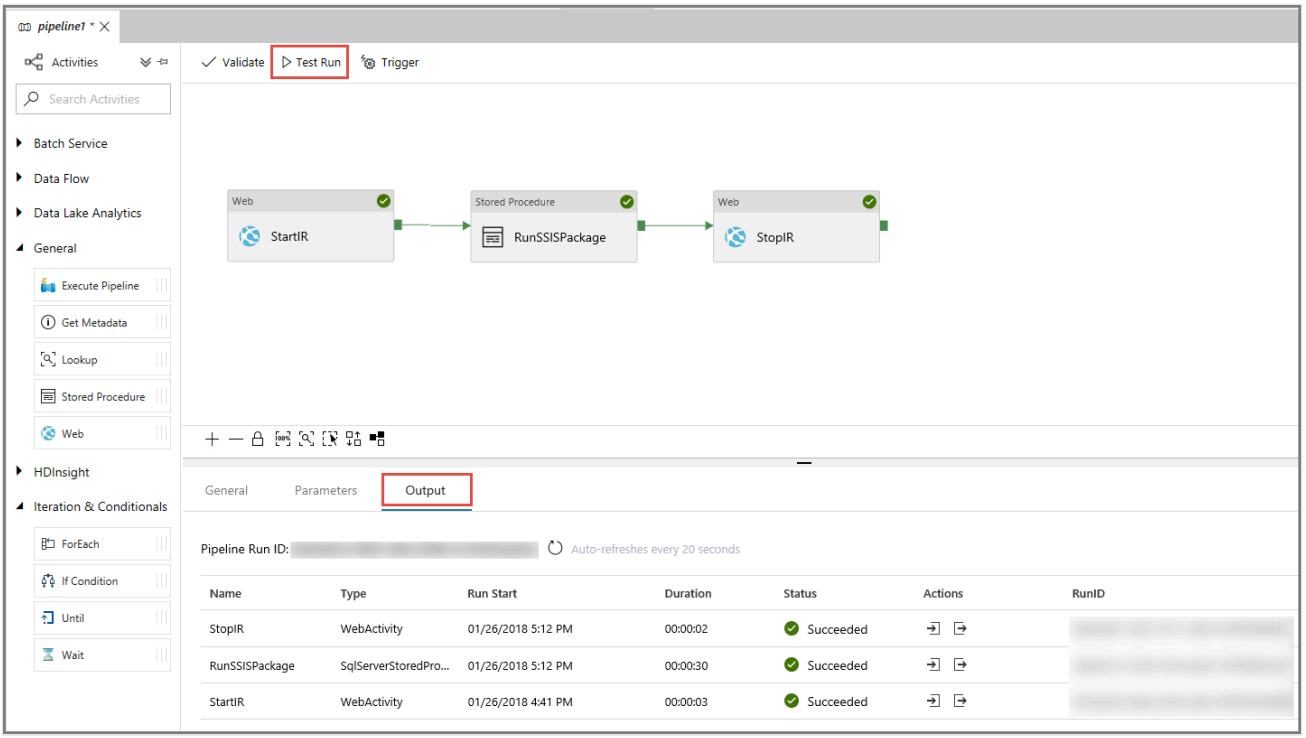


7. Valide la factoría de datos y toda la configuración de canalización seleccionando **Validar todo** o **Validar** en la barra de herramientas de fábrica o canalización. Cierre la **Salida de validación de fábrica** o la **Salida de validación de canalización** seleccionando el botón de flecha doble (>>).

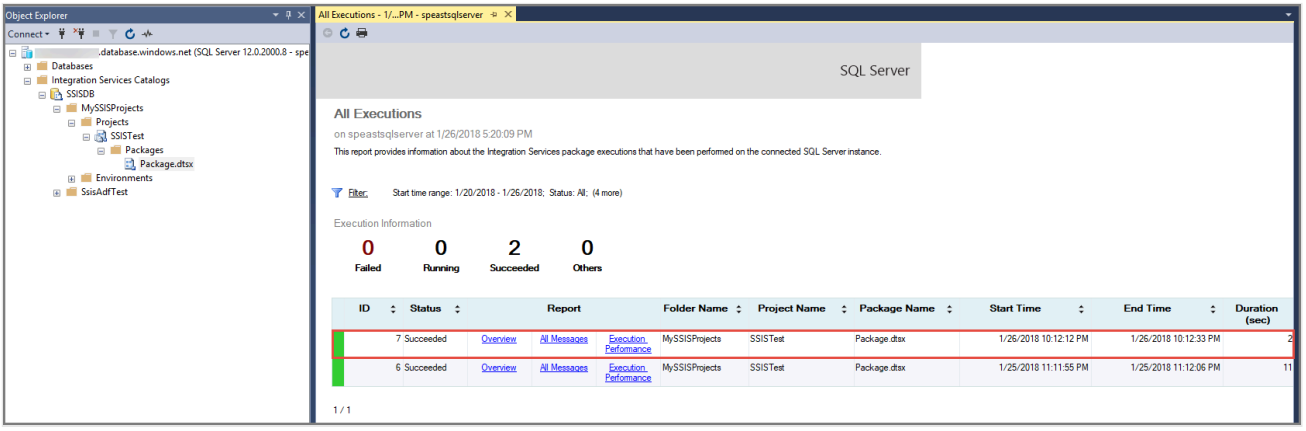


Serie de pruebas de las canalizaciones

1. Seleccione **Serie de pruebas** en la barra de herramientas para cada canalización. En el panel inferior, en la pestaña **Salida** se enumeran las ejecuciones de canalización.



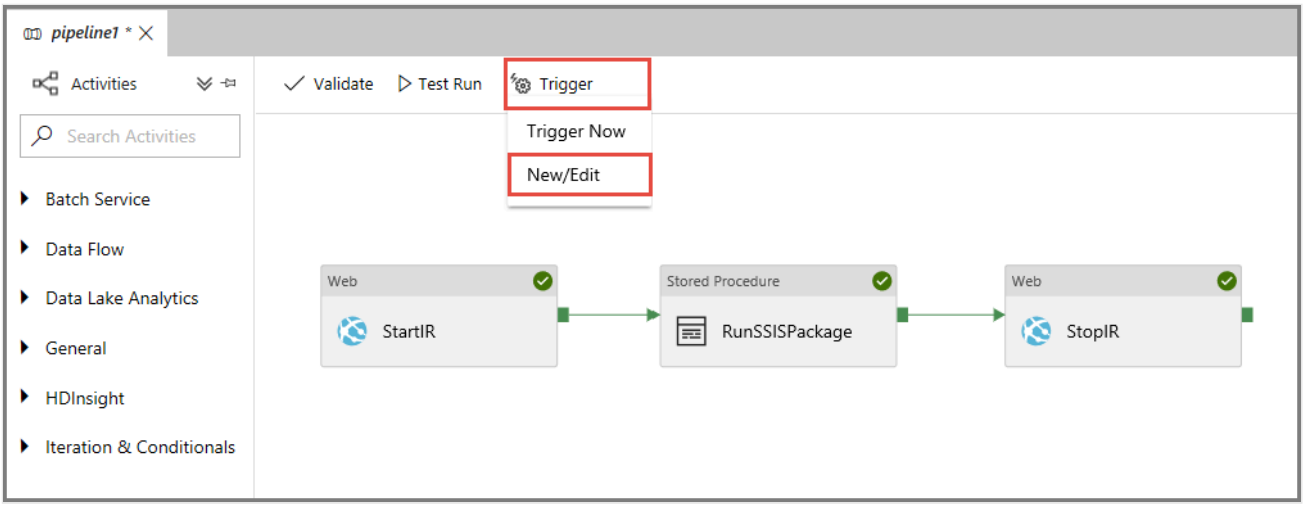
2. Para probar la tercera canalización, puede usar SQL Server Management Studio si almacena el paquete de SSIS en el catálogo de SSIS (SSISDB). En la ventana **Conectar al servidor**, realice las acciones siguientes:
   a. En **Nombre del servidor**, escriba **<el nombre del servidor>.database.windows.net**.
   b. Seleccione **Opciones >>**.
   c. En **Conectar una base de datos**, seleccione **SSISDB**.
   d. Seleccione **Conectar**.
   e. Expanda **Catálogos de Integration Services**>**SSISDB**> su carpeta >**Proyectos**> su proyecto de SSIS >**Paquetes**.
   f. Haga clic con el botón derecho en el paquete de SSIS especificado a ejecutar y luego seleccione **Informes**>**Informes estándar**>**Todas las ejecuciones**.
   g. Compruebe que se ejecuta el paquete.



Programación de las canalizaciones

Ahora que las canalizaciones funcionan tal como esperaba, puede crear desencadenadores para ejecutarlas a las cadencias especificadas. Para más información sobre cómo asociar desencadenadores con canalizaciones, consulte Configuración de programaciones para canalizaciones.

1. En la barra de herramientas de la canalización, seleccione **Desencadenador** y luego, seleccione **Nuevo/Editar**.



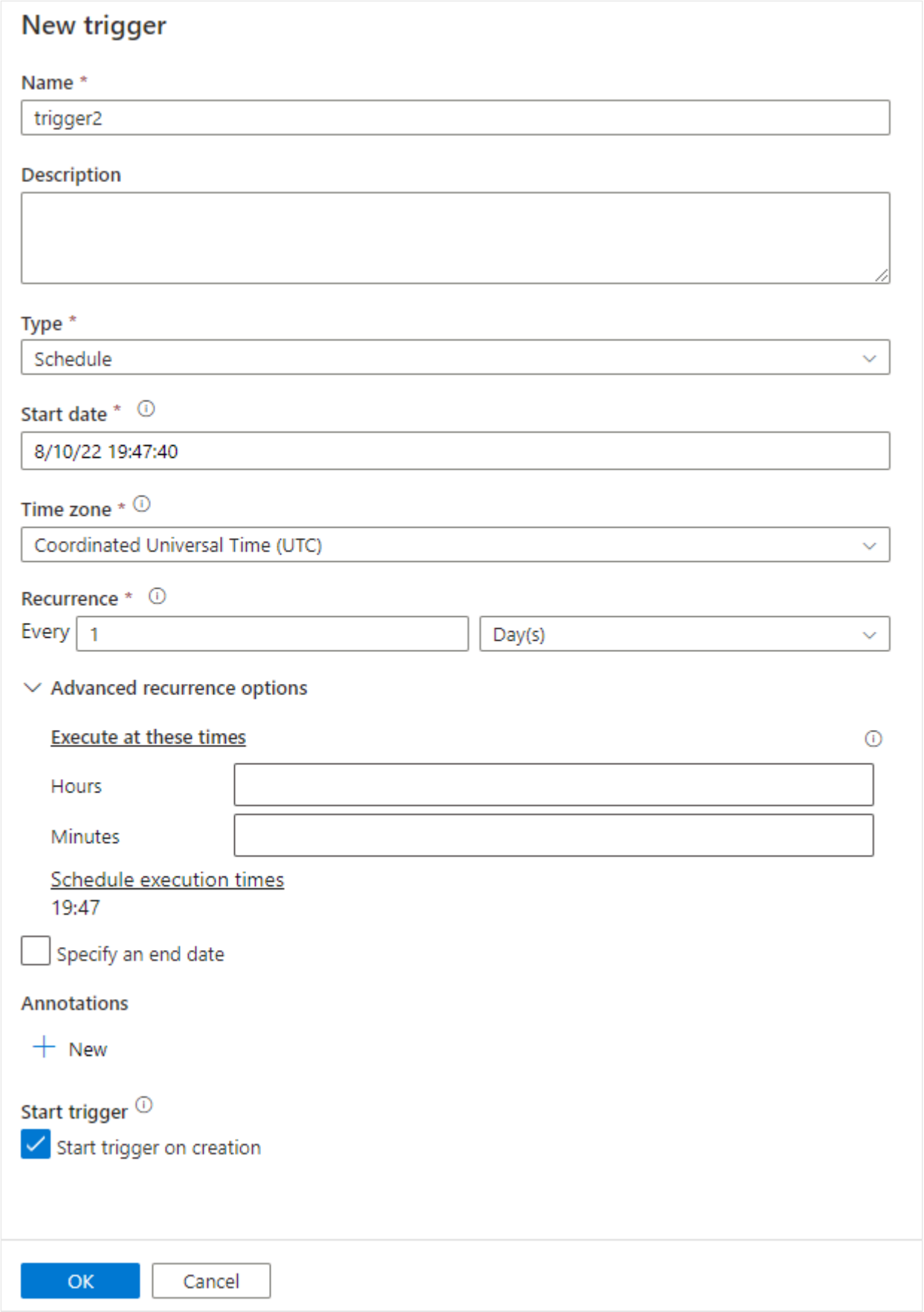
2. En el panel **Agregar desencadenadores**, seleccione **+ Nuevo**.

3. En el **Nuevo desencadenador**, lleve a cabo las siguientes acciones:
   a. En **Nombre**, escriba el nombre del desencadenador. En el ejemplo siguiente, **desencadenador2** es el nombre del desencadenador.
   b. En **Tipo**, seleccione **Programación**.
   c. En **Fecha de inicio**, seleccione una fecha y hora de inicio en UTC.
   d. En **Periodicidad**, especifique la cadencia para el desencadenador. En el ejemplo siguiente, es una vez al día.

e. Si desea que el desencadenador tenga una fecha de finalización, seleccione **Especificar una fecha de finalización** y, a continuación, seleccione una fecha y hora.

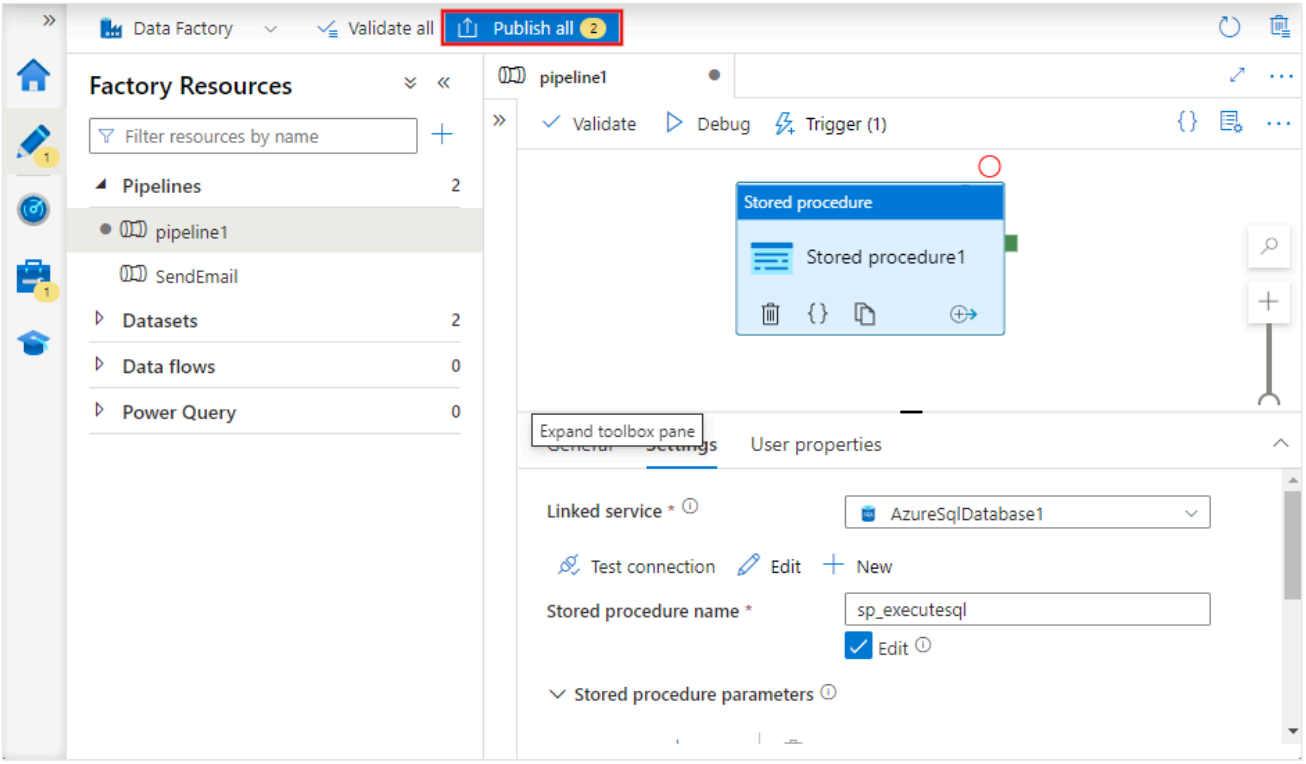
f. Seleccione **Iniciar desencadenador al crear** para activar el desencadenador inmediatamente después de publicar toda la configuración de Data Factory.

g. Seleccione **Aceptar**.



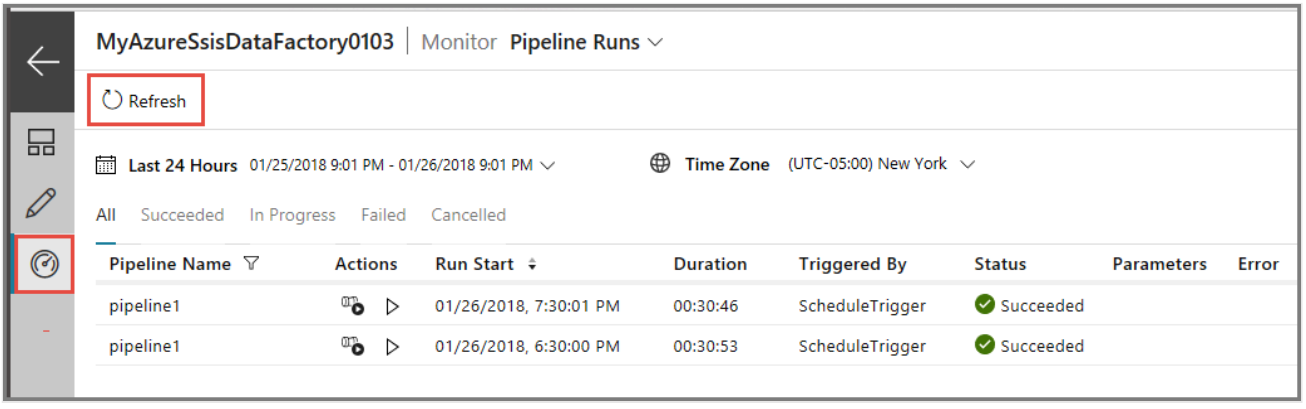
4. En la página **Parámetros de ejecución de desencadenador**, revise si hay advertencias y después seleccione **Finalizar**.

5. Para publicar toda la configuración de Data Factory, seleccione **Publicar todo** en la barra de herramientas de fábrica.



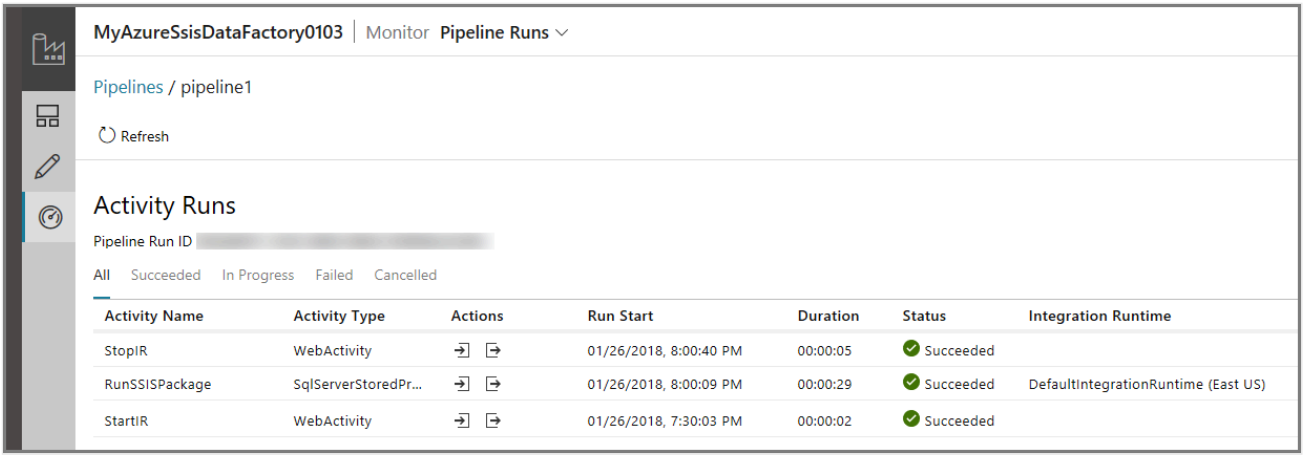
Supervisión de las canalizaciones y los desencadenadores en Azure Portal

- Para supervisar las ejecuciones de desencadenador y las ejecuciones de canalización, use la pestaña **Supervisión** en el lado izquierdo de la interfaz de usuario o la aplicación de Data Factory. Para ver los pasos detallados, consulte Supervisión visual de Azure Data Factory.

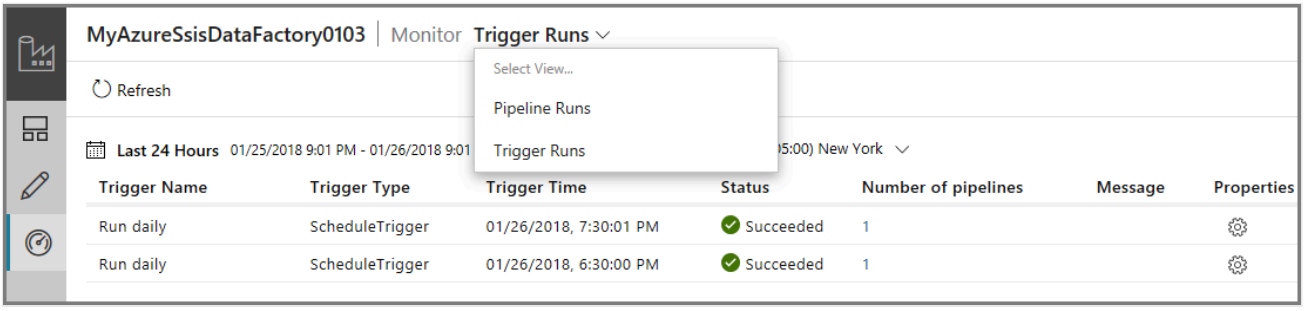


- Para ver las ejecuciones de actividad asociadas con la de una canalización, seleccione el primer vínculo (**View Activity Runs** [Ver ejecuciones de actividad]) de la columna **Actions** (Acciones). En el caso de la tercera canalización, aparecen tres ejecuciones de actividad: una

por cada actividad encadenada en la canalización (actividad web para iniciar el IR, actividad Ejecución de paquetes de SSIS y actividad web para detener el IR). Para ver de nuevo las ejecuciones de canalización, seleccione el vínculo **Canalizaciones** de la parte superior.



- Para ver las ejecuciones de desencadenador, seleccione **Ejecuciones de desencadenador** en la lista desplegable **Ejecuciones de canalización** de la parte superior.



Supervisión de las canalizaciones y los desencadenadores mediante PowerShell

Utilice scripts como los de los ejemplos siguientes para supervisar las canalizaciones y los desencadenadores:

- Obtener el estado de una ejecución de canalización:

PowerShell

```
Get-AzDataFactoryV2PipelineRun -ResourceGroupName $ResourceGroupName -
DataFactoryName $DataFactoryName -PipelineRunId $myPipelineRun
```

- Obtener información sobre un desencadenador:

PowerShell

```
Get-AzDataFactoryV2Trigger -ResourceGroupName $ResourceGroupName -DataFactoryName
$DataFactoryName -Name  "myTrigger"
```

- Obtener el estado de una ejecución de desencadenador:

PowerShell

```
Get-AzDataFactoryV2TriggerRun -ResourceGroupName $ResourceGroupName -
DataFactoryName $DataFactoryName -TriggerName "myTrigger" -TriggerRunStartedAfter
"2018-07-15" -TriggerRunStartedBefore "2018-07-16"
```

Creación y programación de un runbook de Azure Automation que inicia o detiene un Azure-SSIS IR

En esta sección, aprende cómo crear un runbook de Azure Automation que ejecuta el script de PowerShell, para iniciar y detener Azure-SSIS IR mediante una programación. Esta información es útil cuando desea ejecutar scripts adicionales antes o después de iniciar y detener el entorno de ejecución de integración para el procesamiento previo y posterior.

Creación de la cuenta de Azure Automation

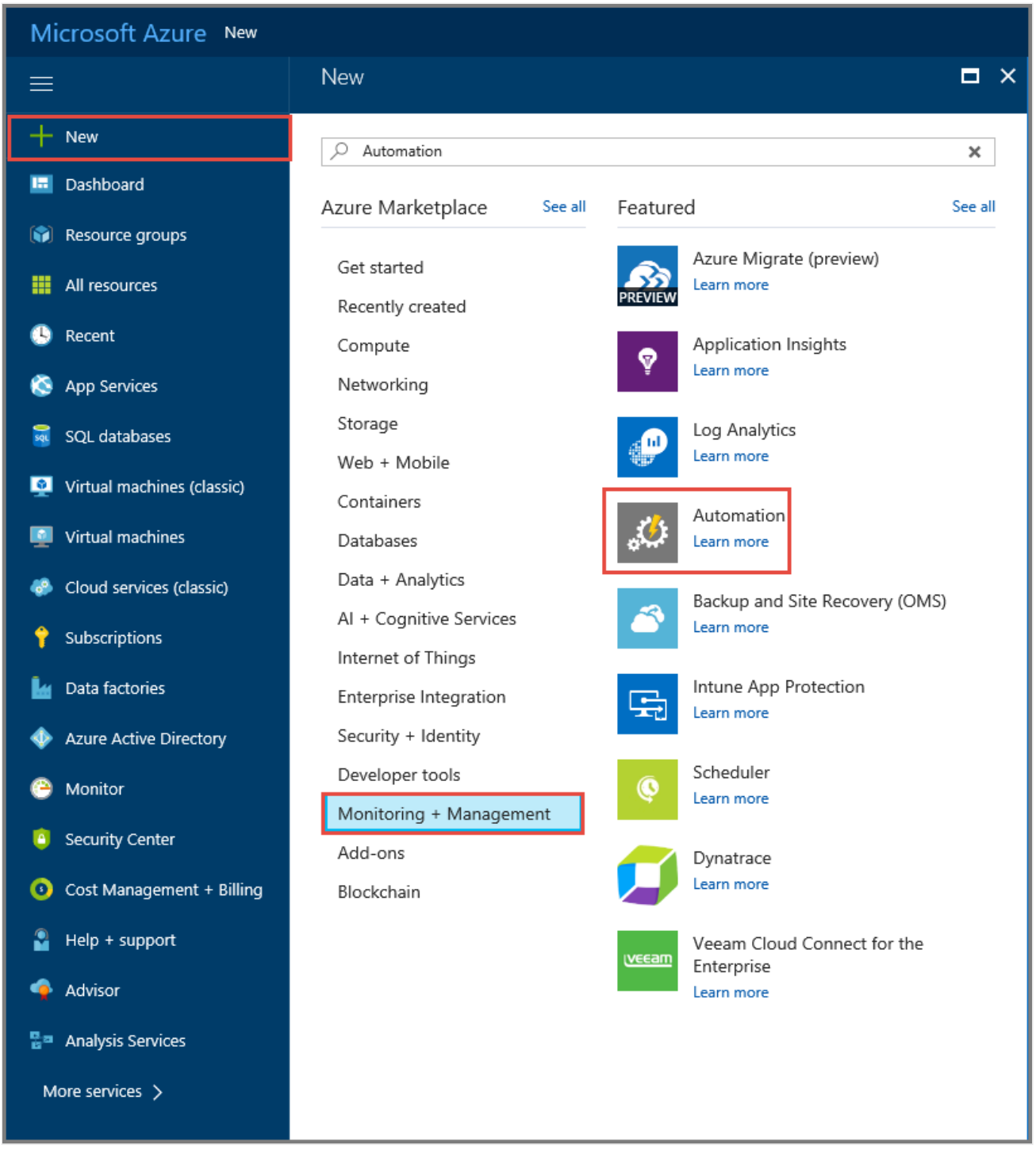
Si no tiene una cuenta de Azure Automation, cree una siguiendo las instrucciones de esta sección. Para obtener instrucciones detalladas, consulte Creación de una cuenta de Azure Automation.

Como parte de este proceso, va a crear una cuenta de **Ejecutar Azure como** (una entidad de servicio en su instancia de Microsoft Entra ID) y le va a asignar el rol de **Colaborador** en su suscripción a Azure. Asegúrese de que sea la misma suscripción que contiene la factoría de datos con el Azure-SSIS IR. Azure Automation utilizará esta cuenta para autenticarse en Azure Resource Manager y trabajar con sus recursos.

1. Abra el explorador web Microsoft Edge o Google Chrome. Actualmente, la interfaz de usuario de Data Factory solo se admite en estos exploradores.

2. Inicie sesión en Azure Portal .

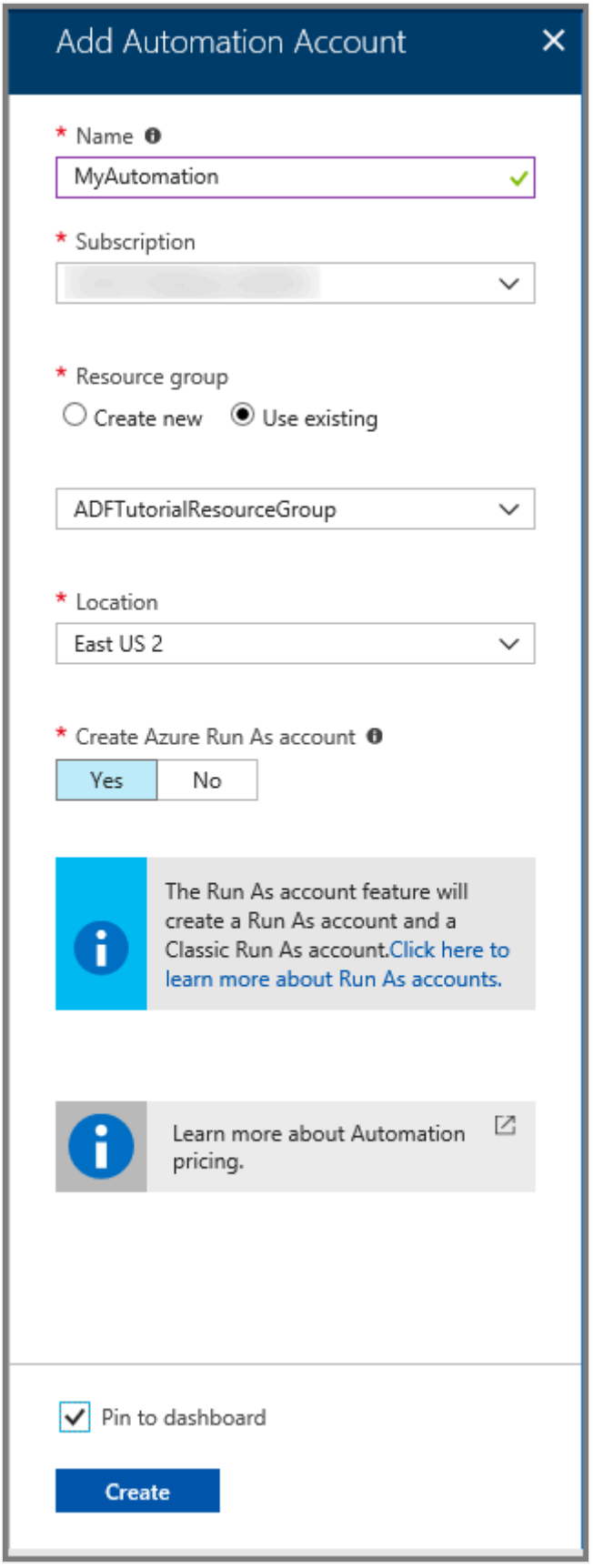
3. Seleccione **Nuevo** en el menú de la izquierda, seleccione **Supervisión y administración** y, después, seleccione **Automation**.



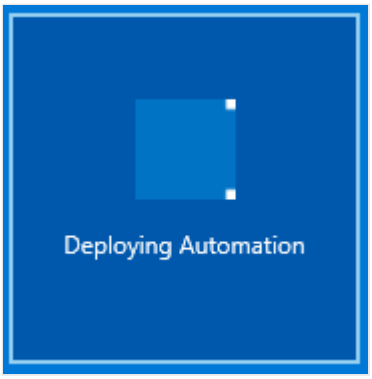
4. En el panel **Agregar cuenta de Automation**, lleve a cabo las siguientes acciones:
   a. En **Nombre**, escriba un nombre para la cuenta de Azure Automation.
   b. En **Suscripción**, seleccione la suscripción que tiene la instancia de factoría de datos con Azure-SSIS IR.
   c. En **Grupo de recursos**, seleccione **Crear nuevo** para crear un grupo de recursos o seleccione **Usar existente** para usar un grupo de recursos existente.
   d. En **Ubicación**, seleccione la ubicación para la cuenta de Azure Automation.
   e. Para **Crear la cuenta de ejecución de Azure**, seleccione **Sí**. Se creará una entidad de servicio en la instancia de Microsoft Entra y se le asignará un rol de **Colaborador** en la suscripción de Azure.

f. Seleccione **Anclar al panel** para mostrar la cuenta de forma permanente en el panel de Azure.

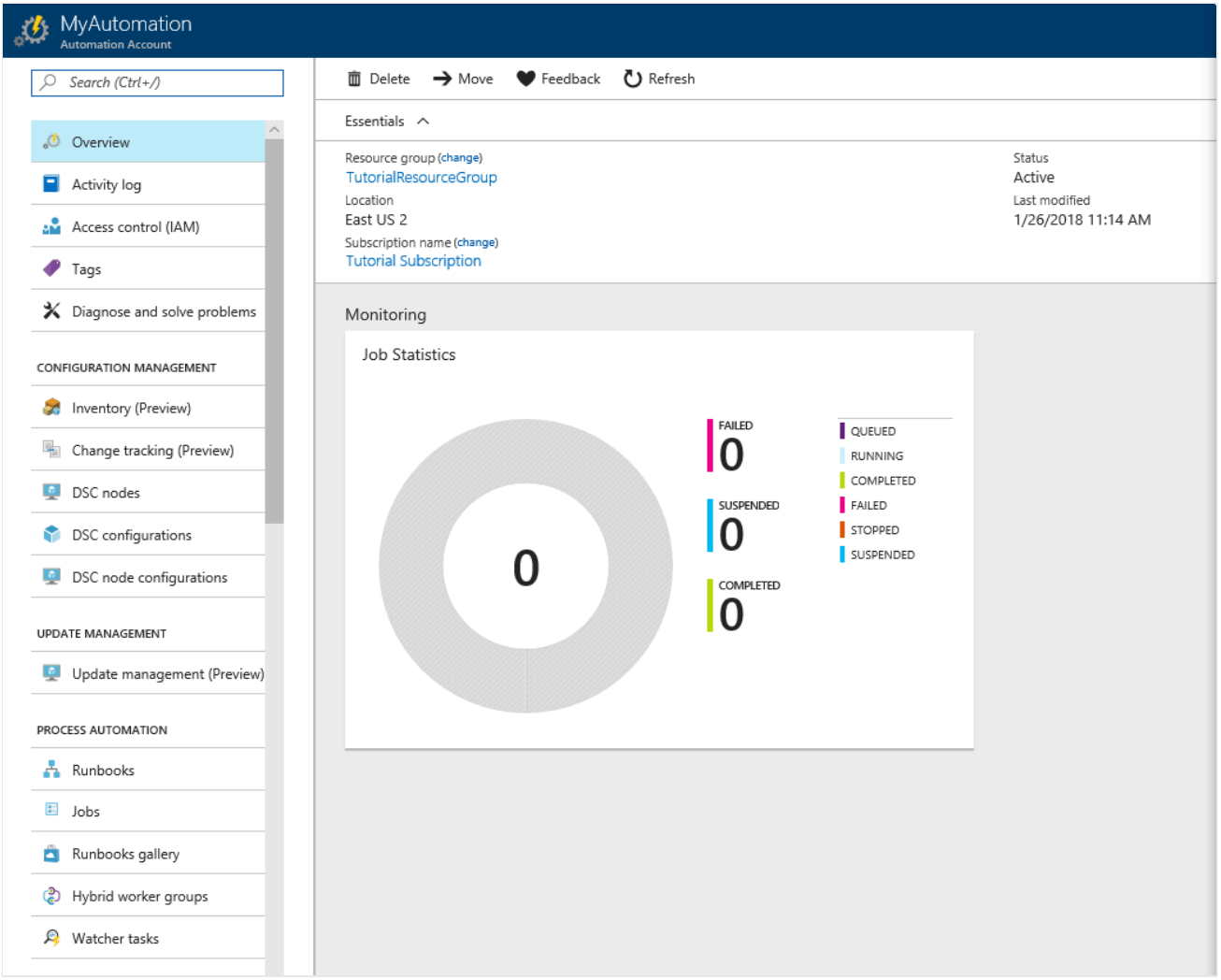
g. Seleccione **Crear**.



5. Supervise el estado de implementación de la cuenta de Azure Automation en el panel de Azure y las notificaciones.

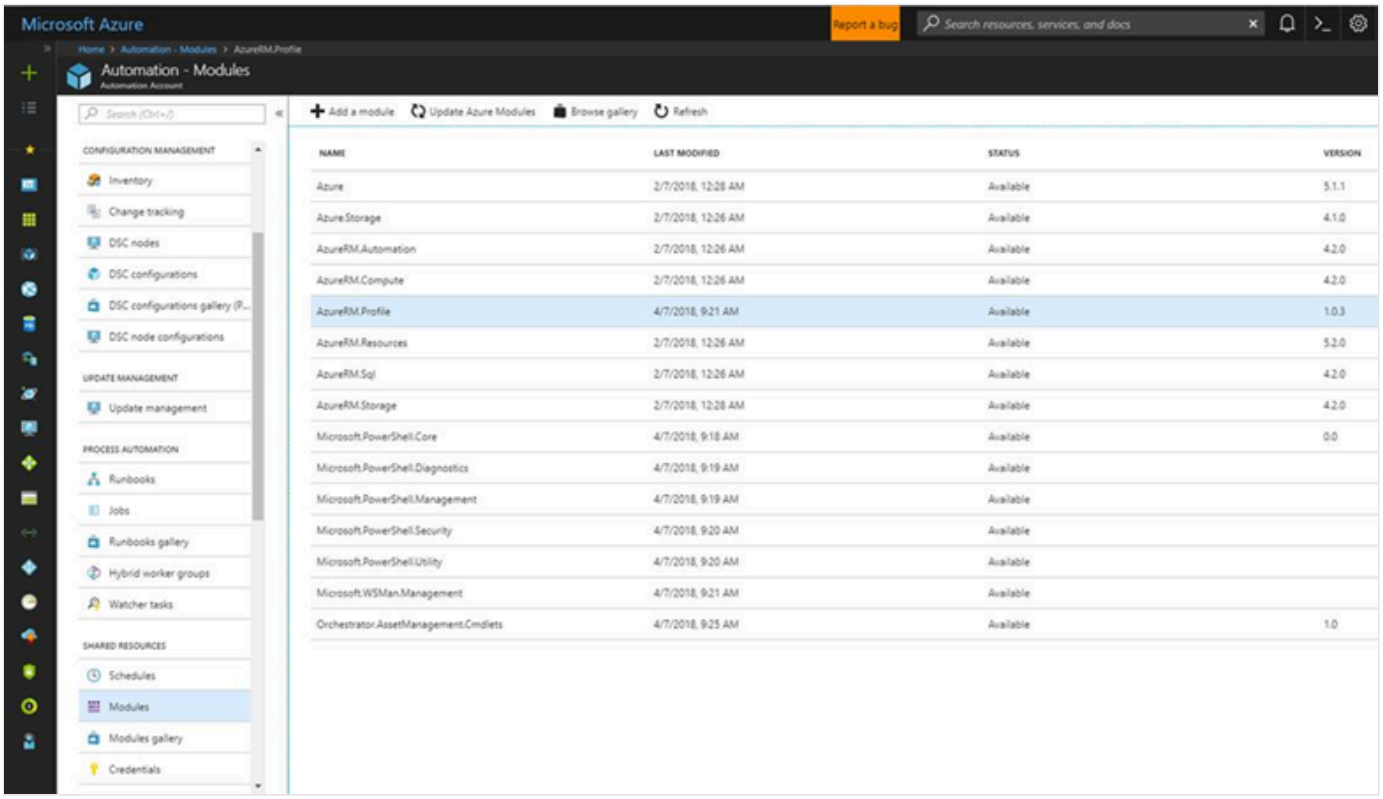


6. Confirme que aparece la página principal de la cuenta de Azure Automation. Significa que ha creado la cuenta correctamente.



Importación de módulos de Data Factory

En el menú de la izquierda, en la sección **RECURSOS COMPARTIDOS**, seleccione **Módulos**. Compruebe que tiene **Az.DataFactory** y **Az.Profile** en la lista de módulos. Ambos son obligatorios.

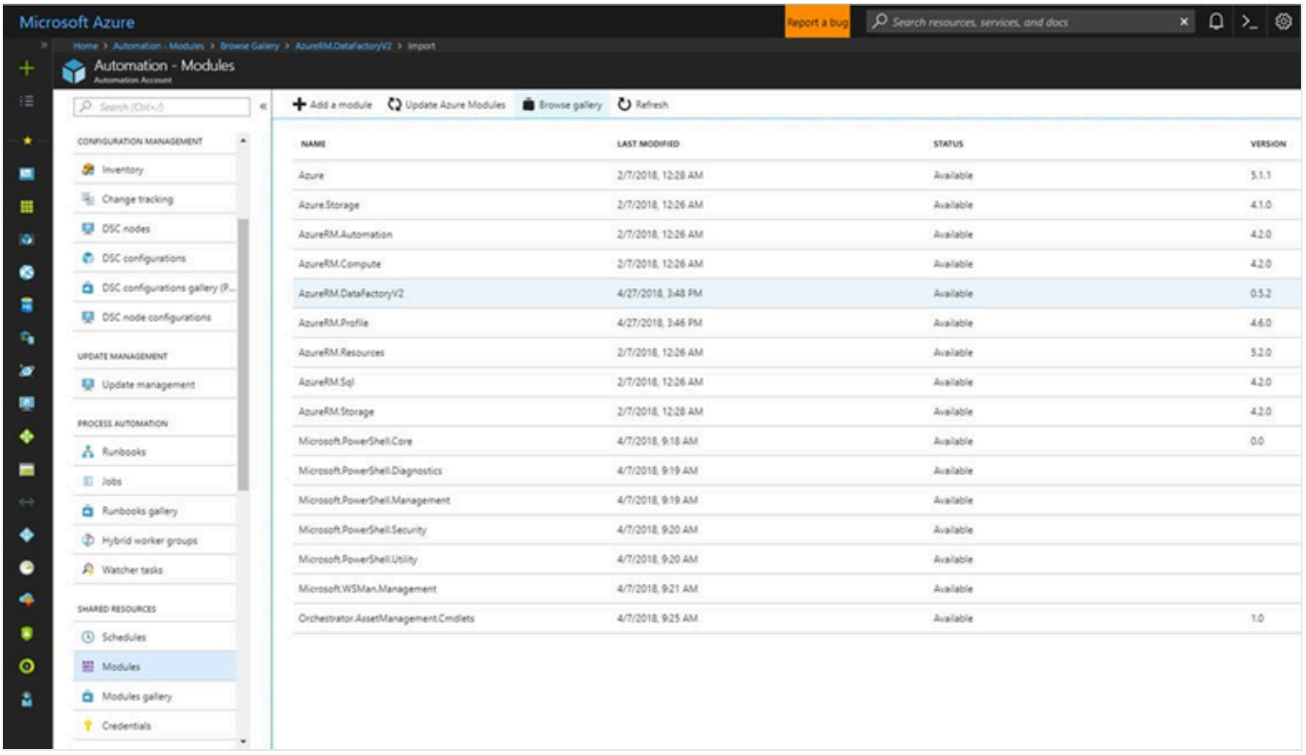


Si no tiene **Az.DataFactory**:

1. Vaya al módulo de Az.DataFactory en el Galería de PowerShell.

2. Seleccione **Implementar en Azure Automation**, seleccione la cuenta de Azure Automation y, a continuación, seleccione **Aceptar**.

3. Vuelva a la vista **Módulos** en la sección **RECURSOS COMPARTIDOS** del menú de la izquierda. Espere hasta que el **ESTADO** del módulo **Az.DataFactory** cambie a **Disponible**.

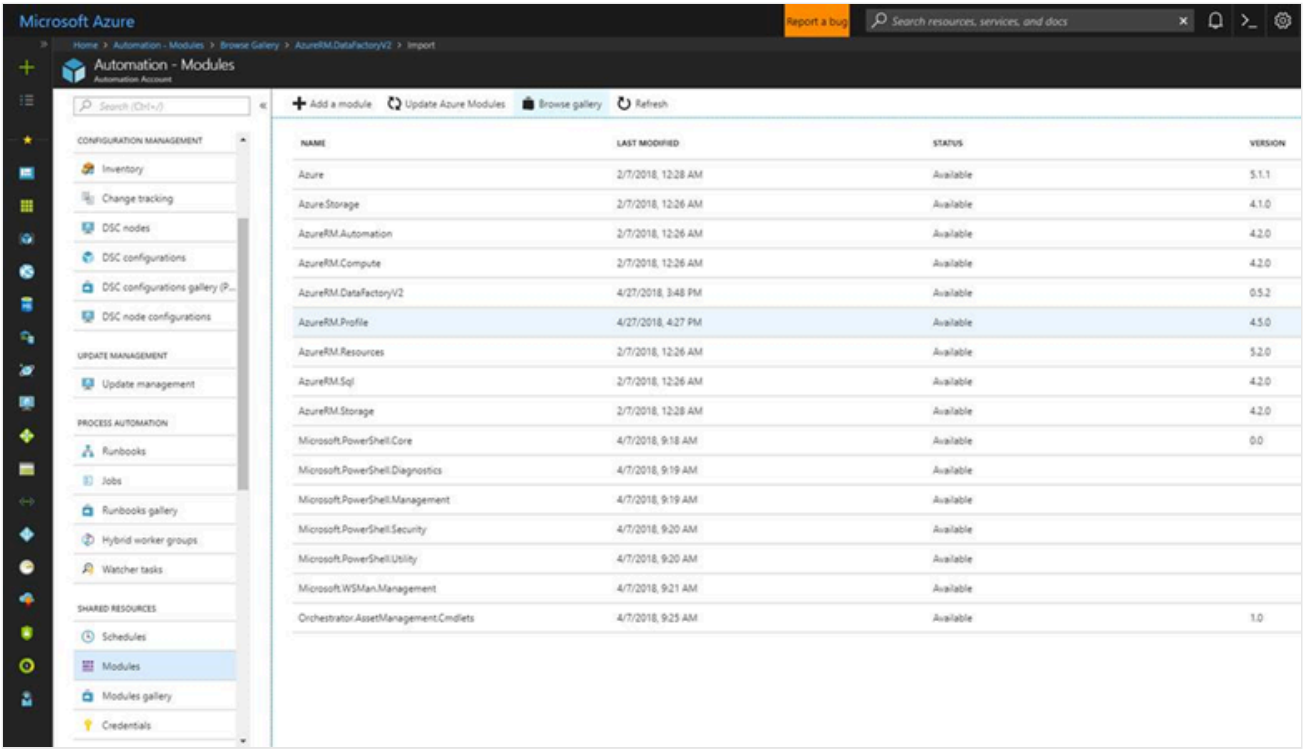


Si no tiene **Az.Profile**:

1. Vaya al módulo Az.Profile de la Galería de PowerShell.

2. Seleccione **Implementar en Azure Automation**, seleccione la cuenta de Azure Automation y, a continuación, seleccione **Aceptar**.

3. Vuelva a la vista **Módulos** en la sección **RECURSOS COMPARTIDOS** del menú de la izquierda. Espere hasta que el **ESTADO** del módulo **Az.Profile** cambie a **Disponible**.

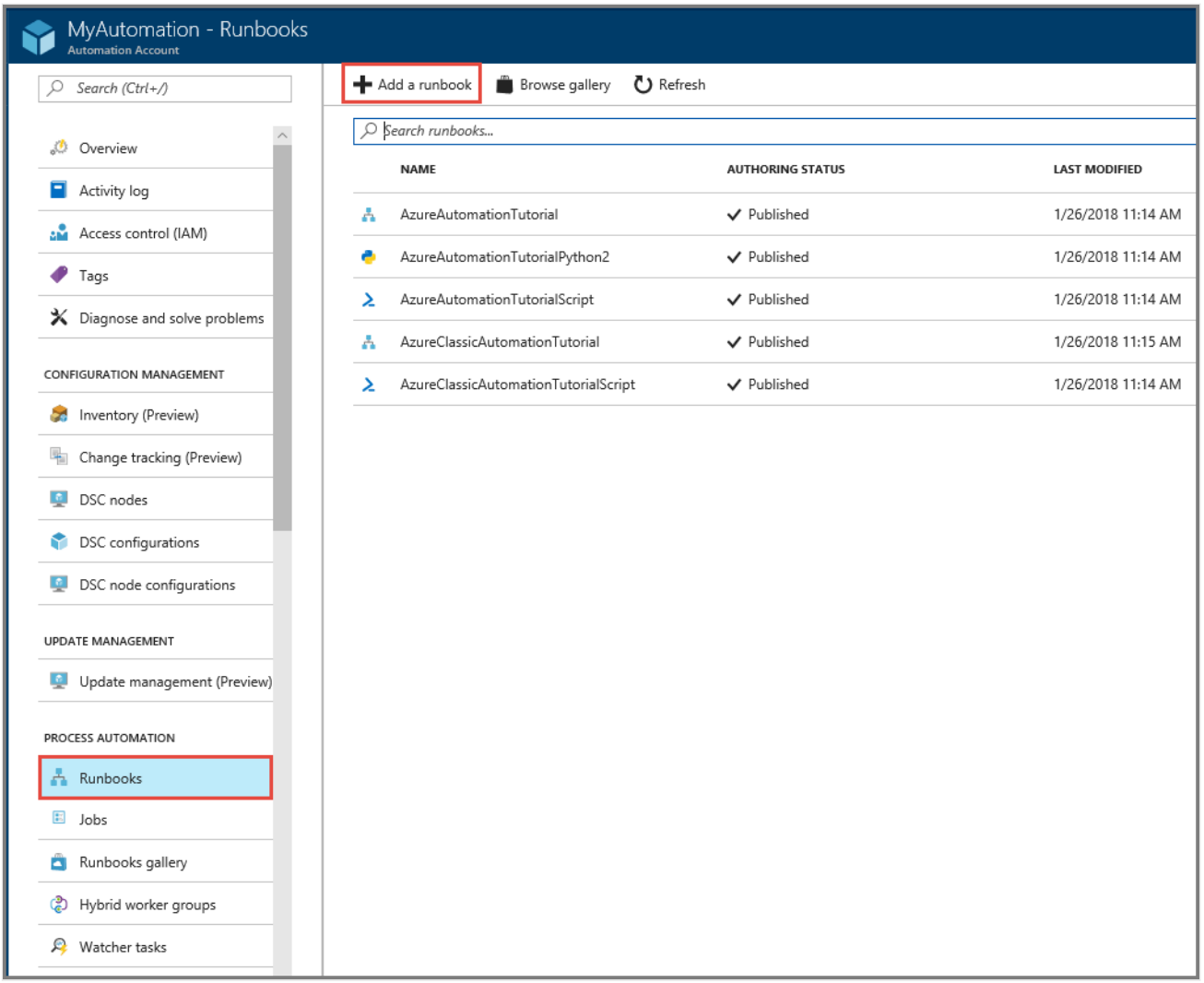


Creación del runbook de PowerShell

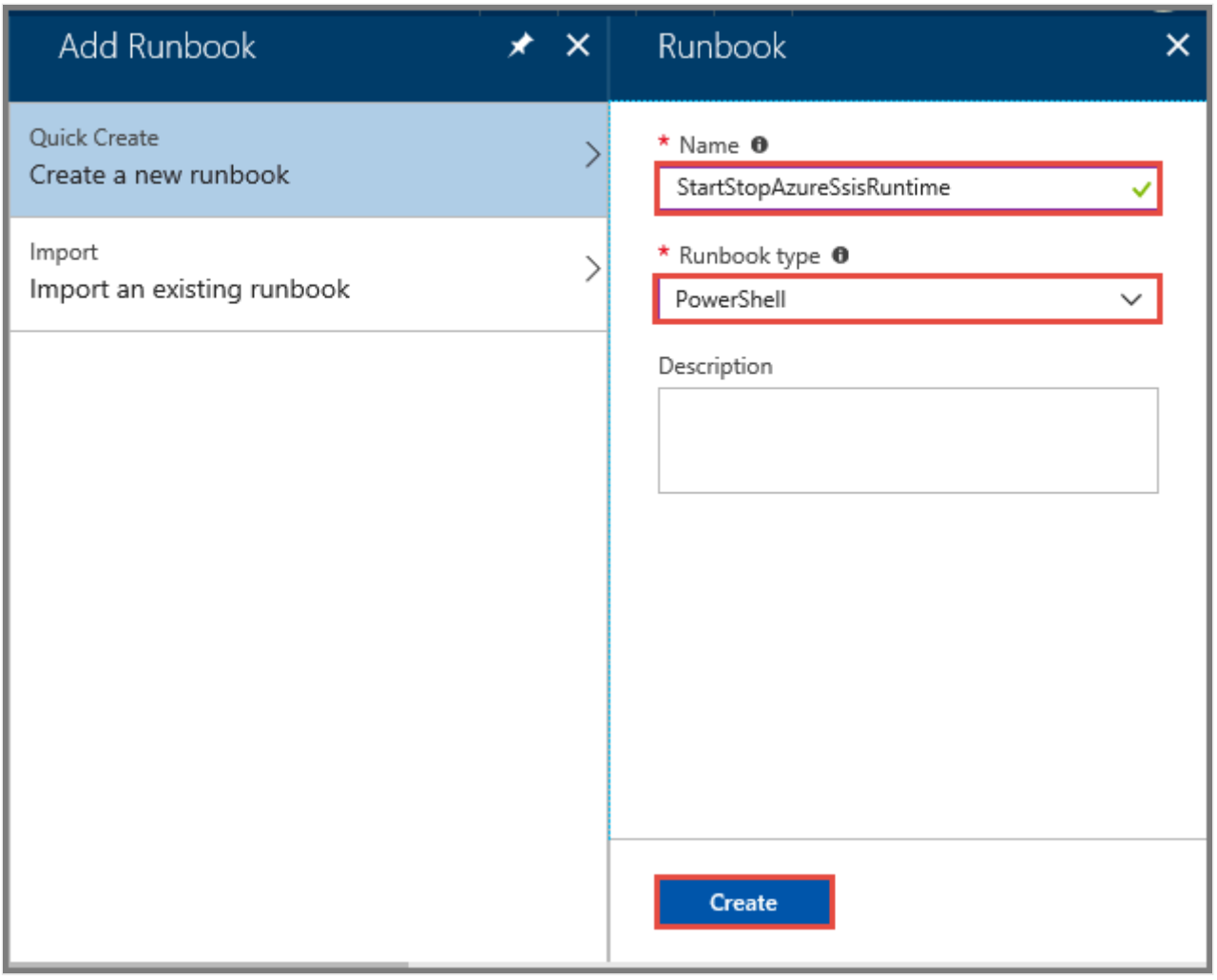
Esta sección proporciona los pasos para la creación de un runbook de PowerShell. El script asociado al runbook inicia o detiene Azure-SSIS IR en función del comando que especifique para el parámetro **OPERATION**.

Los pasos siguientes no proporcionan los detalles completos para crear un runbook. Para obtener más información, consulte Creación de un runbook.

1. Cambie a la pestaña **Runbooks** y seleccione **+ Agregar un runbook** en la barra de herramientas.



2. Seleccione **Crear un runbook nuevo** y después realice los pasos siguientes:
   a. En **Nombre**, escriba **StartStopAzureSsisRuntime**.
   b. En **Tipo de runbook**, seleccione **PowerShell**.
   c. Seleccione **Crear**.



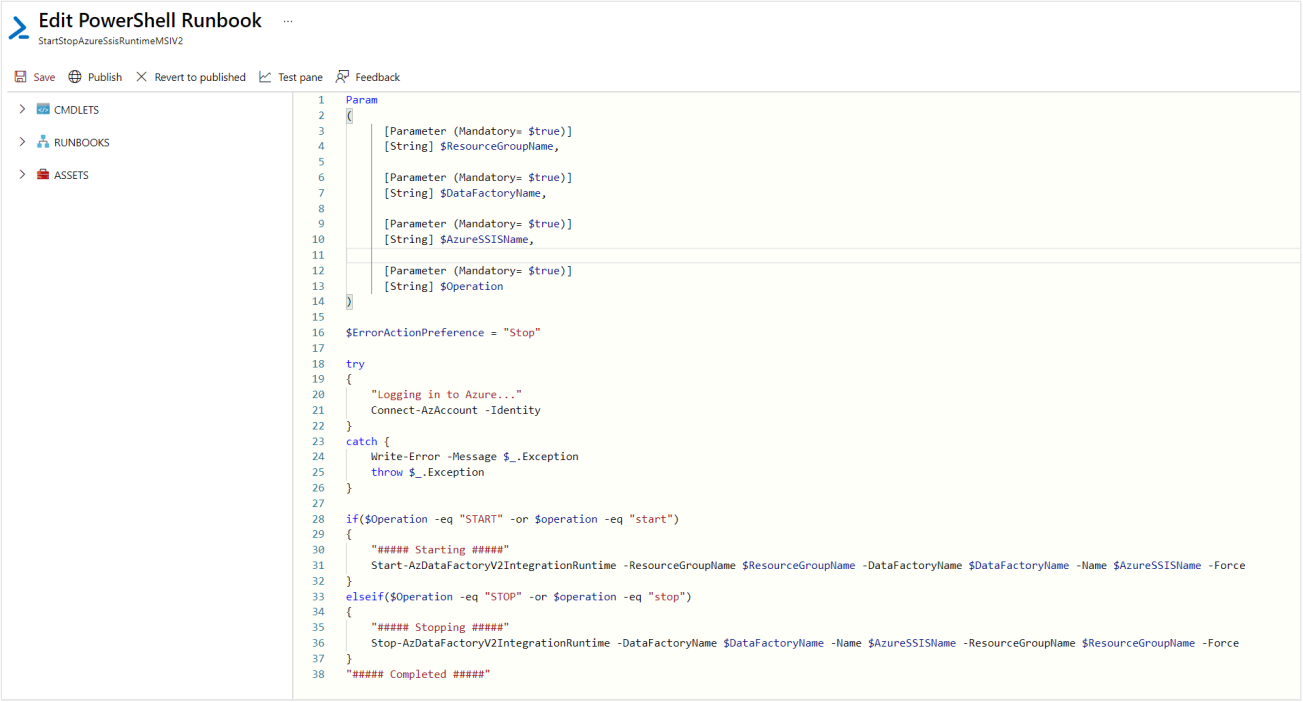
3. Copie y pegue el siguiente script de PowerShell en la ventana de script del runbook. Guarde y luego publique el runbook con los botones **Guardar** y **Publicar** de la barra de herramientas.

> ⓘ **Nota:**
>
> En ese ejemplo se usa una identidad administrada asignada por el sistema. Si usa una cuenta de ejecución (entidad de servicio) o la identidad administrada asignada por el usuario, consulte Scripts de ejemplo de Azure Automation para la parte de inicio de sesión.
>
> Habilite los permisos de control de acceso basado en roles (RBAC) adecuados para la identidad administrada de esta cuenta de Automation. Para más información, consulte Roles y permisos para Azure Data Factory.

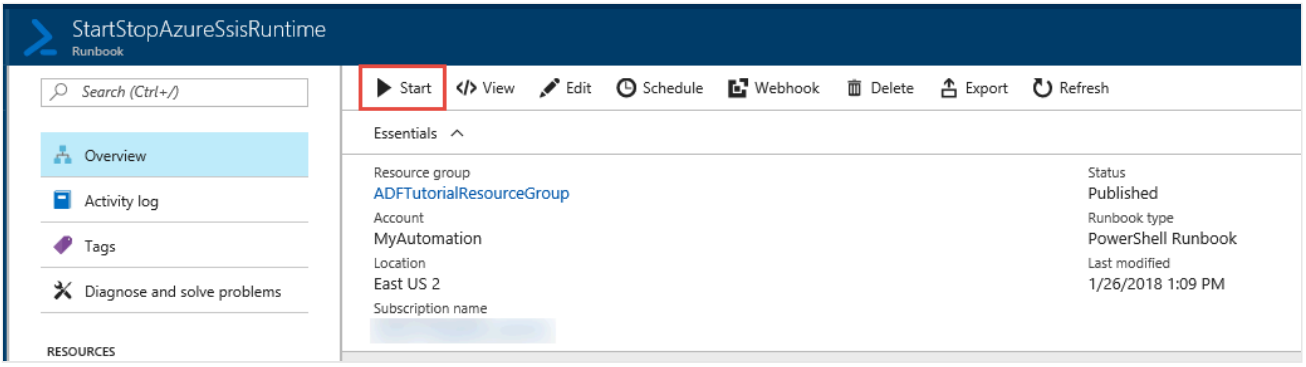
PowerShell

```
Param
(
      [Parameter (Mandatory= $true)]
```

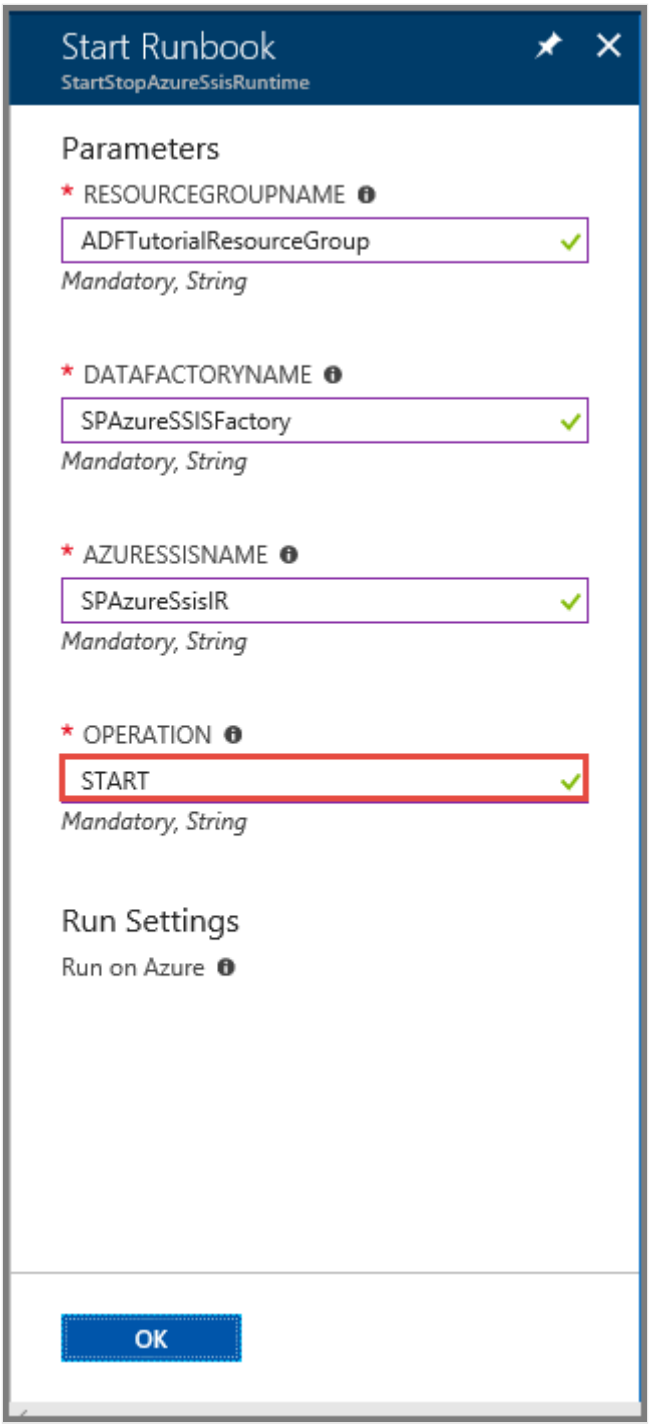
```
      [String] $ResourceGroupName,

      [Parameter (Mandatory= $true)]
      [String] $DataFactoryName,

      [Parameter (Mandatory= $true)]
      [String] $AzureSSISName,

      [Parameter (Mandatory= $true)]
      [String] $Operation
)

$ErrorActionPreference = "Stop"

try
{
    "Logging in to Azure..."
    Connect-AzAccount -Identity
}
catch {
    Write-Error -Message $_.Exception
    throw $_.Exception
}

if($Operation -eq "START" -or $operation -eq "start")
{
    "##### Starting #####"
    Start-AzDataFactoryV2IntegrationRuntime -ResourceGroupName $ResourceGroupName
-DataFactoryName $DataFactoryName -Name $AzureSSISName -Force
}
elseif($Operation -eq "STOP" -or $operation -eq "stop")
{
    "##### Stopping #####"
    Stop-AzDataFactoryV2IntegrationRuntime -DataFactoryName $DataFactoryName -
Name $AzureSSISName -ResourceGroupName $ResourceGroupName -Force
}
"##### Completed #####"
```



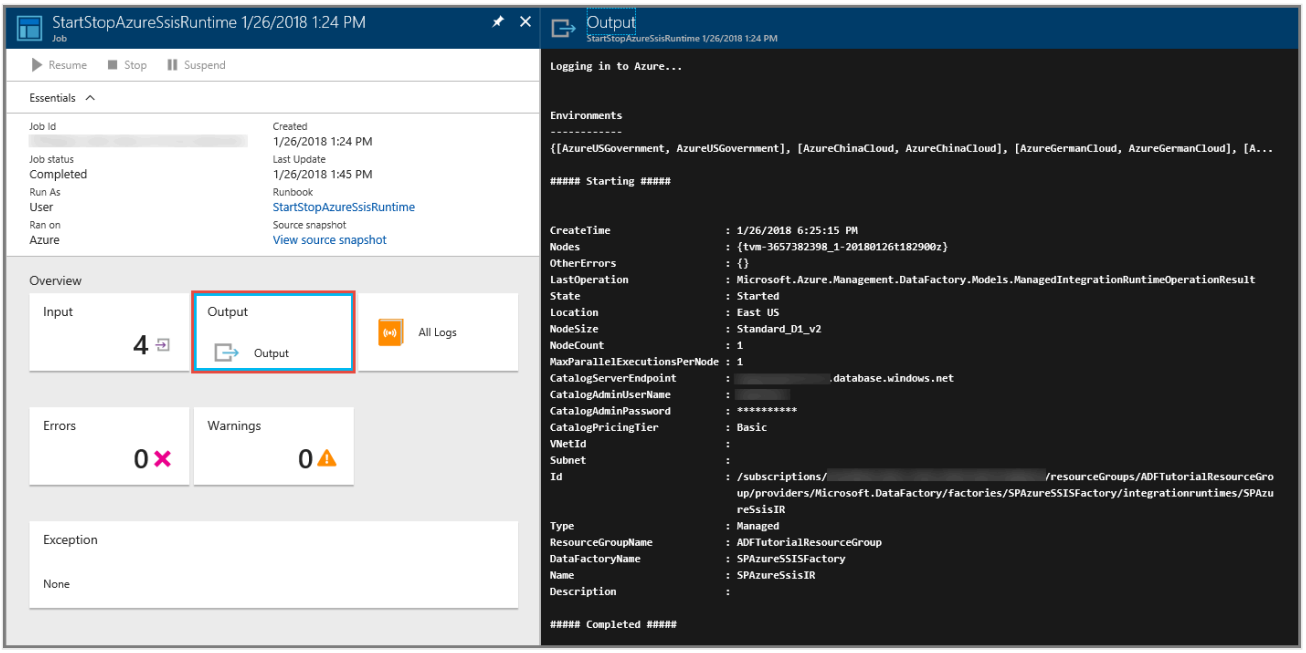
4. Pruebe el runbook seleccionando el botón **Iniciar** en la barra de herramientas.



5. En el panel **Iniciar runbook**, haga las siguientes acciones:
   a. En **RESOURCEGROUPNAME**, escriba el nombre del grupo de recursos que tenga la factoría de datos con el Azure-SSIS IR.
   b. En **DATAFACTORYNAME**, escriba el nombre de la factoría de datos con el Azure-SSIS IR.
   c. En **AZURESSISNAME**, escriba el nombre de Azure-SSIS IR.
   d. En **OPERATION** (Operación), escriba **START** (Iniciar).
   e. Seleccione **Aceptar**.



6. En el panel **Trabajo**, seleccione el icono **Salida** . En la ventana de **Salida**, espere hasta ver el mensaje #### **Completado** #### después de ver #### **Iniciando** ####. Cierre el panel **Trabajo** y vuelva a la página **Runbook**.



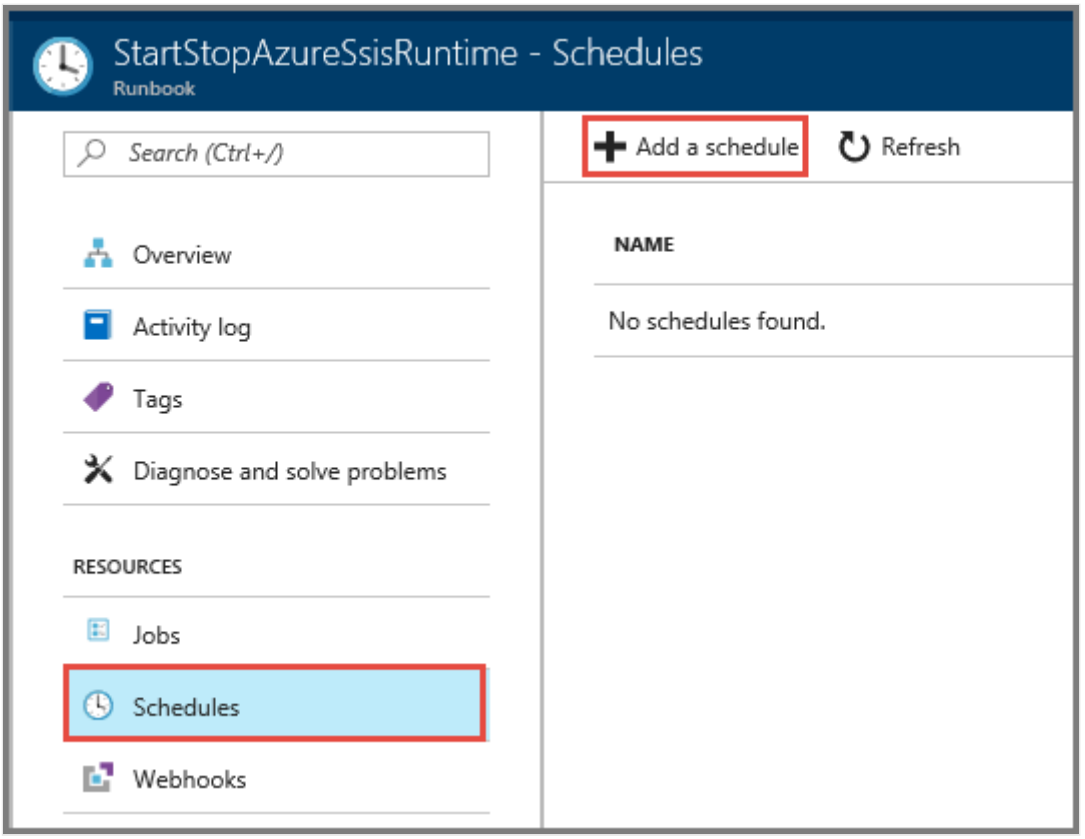
7. Repita los dos pasos anteriores mediante **STOP** como valor de **OPERATION**. Inicie de nuevo el runbook seleccionando el botón **Iniciar** en la barra de herramientas. Escriba los nombres del grupo de recursos, la factoría de datos y el Azure-SSIS IR. En **OPERATION** (Operación), escriba **STOP** (Detener). En el panel de **Salida**, espere hasta ver el mensaje **#### Completado ####** después de ver **#### Deteniendo ####**. No se tarda tanto en detener un Azure-SSIS IR como en iniciarlo. Cierre el panel **Trabajo** y vuelva a la página **Runbook**.

8. También puede desencadenar el runbook a través de un webhook. Para crear un webhook, seleccione el elemento de menú **Webhooks**. O bien, puede crear el webhook según una programación seleccionando el elemento de menú **Programaciones**, como se especifica en la sección siguiente.

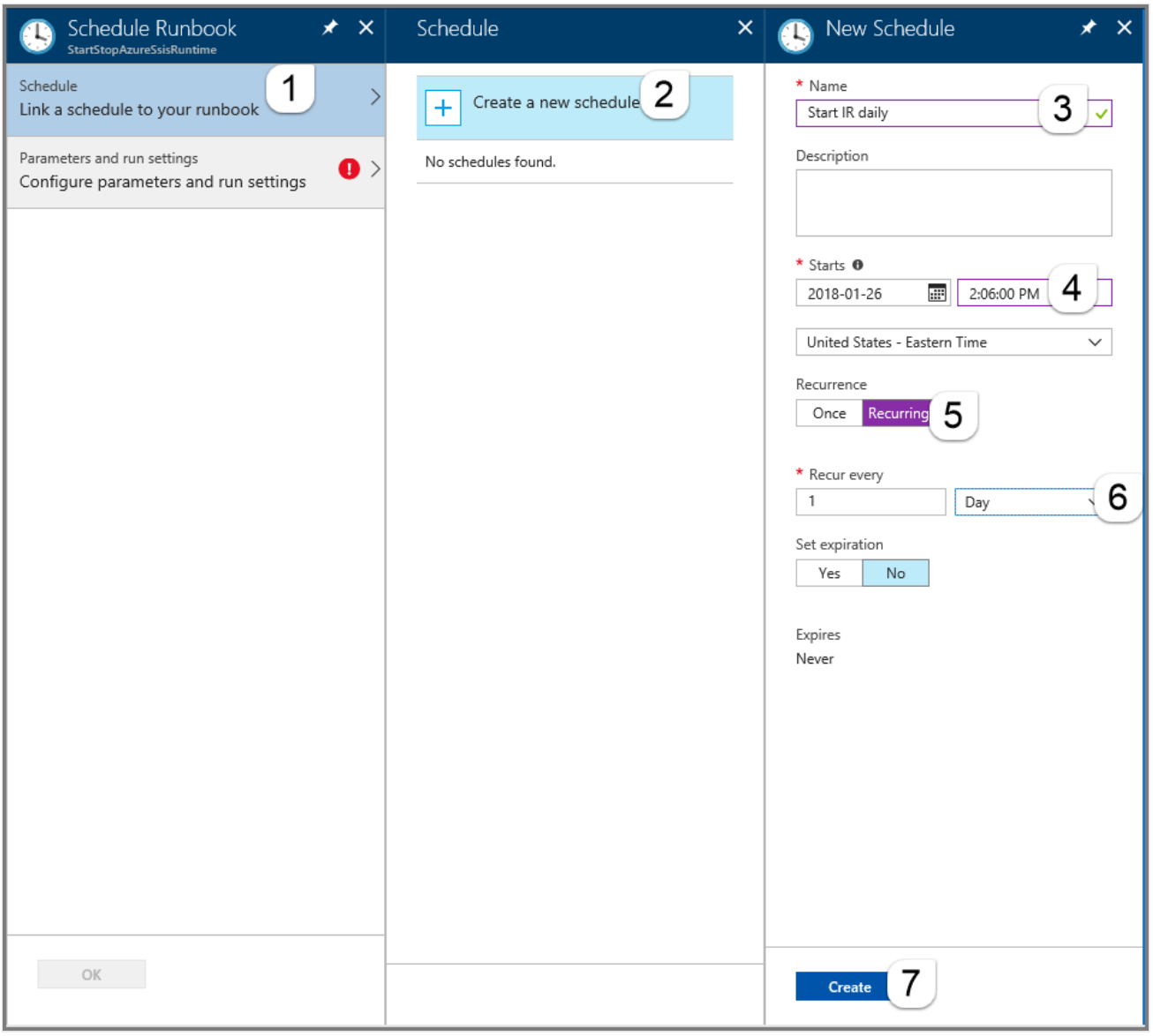
Creación de programaciones para que el runbook inicie o detenga un Azure-SSIS IR

En la sección anterior, ha creado un runbook de Azure Automation que puede iniciar o detener un Azure-SSIS IR. En esta sección, creará dos programaciones para el runbook. Al configurar la primera programación, especifique **INICIAR** para **OPERACIÓN**. Al configurar la segunda programación, especifique **DETENER** para **OPERACIÓN**. Para ver el procedimiento detallado para crear programaciones, consulte Creación de una programación.

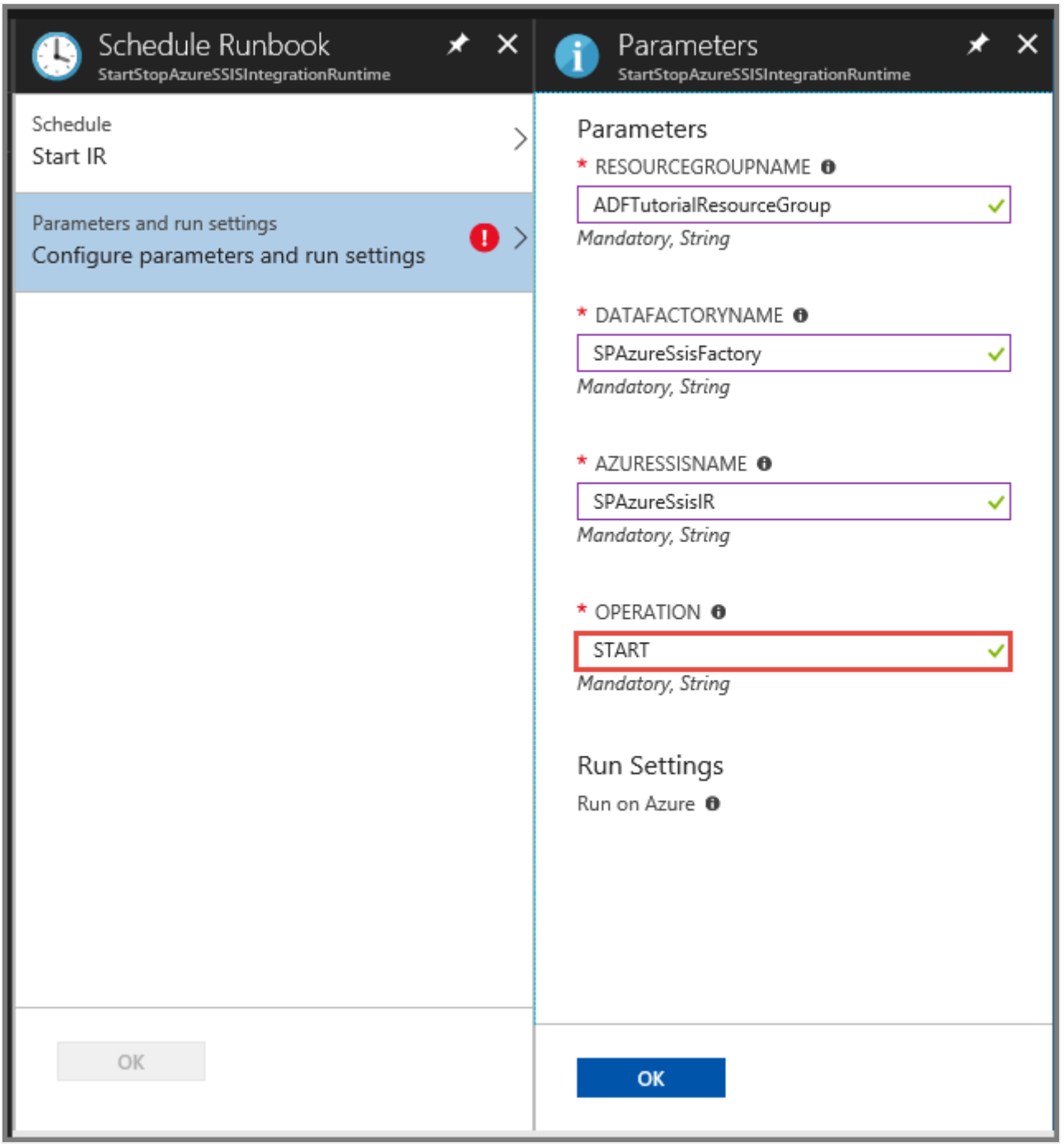
1. En la página **Runbook**, seleccione **Programaciones** y, después, seleccione **+ Agregar una programación** en la barra de herramientas.



2. En el panel **Programar Runbook**, lleve a cabo las acciones siguientes:
   a. Seleccione **Vincular una programación a su Runbook**.
   b. Seleccione **Crear una programación nueva**.
   c. En el panel **Nueva programación**, escriba **Iniciar IR a diario** en **Nombre**.
   d. En **Se inicia el**, especifique una hora que sea unos minutos posterior a la hora actual.
   e. En **Periodicidad**, seleccione **Periódico**.
   f. En **Repetir cada**, escriba **1** y seleccione **Día**.
   g. Seleccione **Crear**.

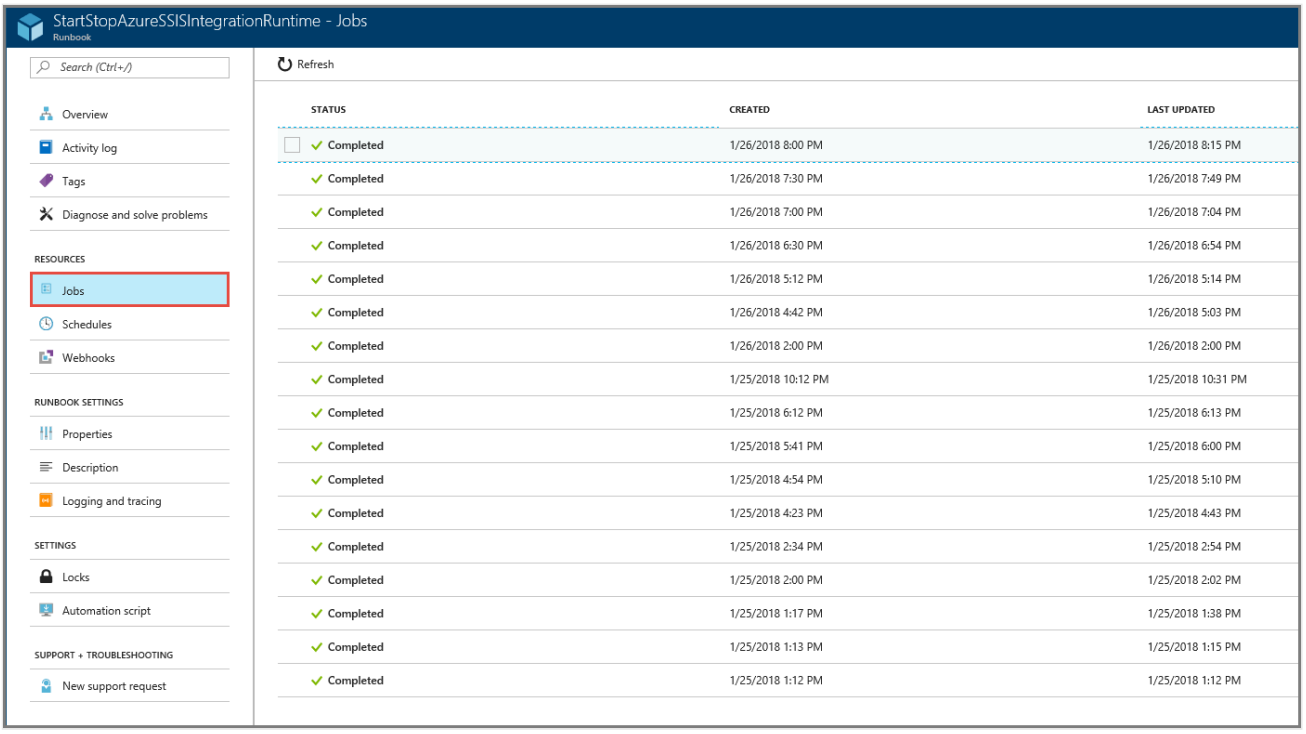


3. Cambie a la pestaña **Configuración de ejecución y parámetros**. Especifique los nombres del grupo de recursos, la factoría de datos y Azure-SSIS IR. Para **OPERACIÓN**, especifique **INICIAR** y, después, seleccione **Aceptar**. Vuelva a seleccionar **Aceptar** para ver la programación en la página **Programaciones** del runbook.



4. Repita los dos pasos anteriores para crear una programación denominada **Detener IR a diario**. Especifique una hora que sea al menos 30 minutos posterior a la hora que ha especificado en la programación **Iniciar IR a diario**. Para **OPERACIÓN**, especifique **DETENER** y, después, seleccione **Aceptar**. Vuelva a seleccionar **Aceptar** para ver la programación en la página **Programaciones** del runbook.

5. En la página **Runbook**, seleccione **Trabajos** en el menú de la izquierda. En la página que se abre se enumeran los trabajos creados por las programaciones en los momentos especificados, junto con sus estados. Puede ver los detalles del trabajo, como su salida, similares a los que aparecieron después de probar el runbook.



6. Cuando termine las pruebas, edite las programaciones para deshabilitarlas. Seleccione **Programaciones** en el menú de la izquierda, seleccione **Iniciar IR a diario o Detener IR a diario** y, después, seleccione **No** para **Habilitado**.

Contenido relacionado

Vea la siguiente entrada de blog:

- Modernización y ampliación de los flujos de trabajo ETL/ETL con actividades de SSIS en las canalizaciones de Azure Data Factory

Consulte los siguientes artículos de la documentación de SSIS:

- Deploy, run, and monitor an SSIS package on Azure (Implementación, ejecución y supervisión de un paquete SSIS en Azure)
- Conexión al Catálogo de SSIS en Azure
- Programación de la ejecución de un paquete en Azure
- Connect to on-premises data sources with Windows authentication (Conexión a orígenes de datos locales con autenticación de Windows)

Last updated on 22/04/2026

Solución de problemas de identidades gestionadas de Azure Automation

En este artículo se describen soluciones para los problemas que pueden surgir al usar una identidad administrada con la cuenta de Automation. Para obtener información general sobre el uso de la identidad administrada con cuentas de Automation, consulte Azure Automation información general sobre la autenticación de cuentas.

Escenario: Se produce un error en el runbook con una identidad administrada asignada por el sistema con un mensaje de error 400

Problema

El runbook con la identidad administrada asignada por el sistema produce un error como `unable to acquire for tenant organizations with error ManagedIdentityCredential authentication failed. Managed System Identity not found! Status 400 (Bad Request)`

Causa

No ha asignado permisos después de crear la identidad administrada asignada por el sistema.

Solución

Asegúrese de asignar los permisos adecuados para la identidad administrada asignada por el sistema. Uso de una identidad administrada asignada por el sistema para una cuenta de Azure Automation

Escenario: La Identidad Administrada en un Runbook no puede autenticarse en Azure

Problema

Al usar una identidad administrada en el runbook, recibirá un error similar a: `connect-azaccount : ManagedIdentityCredential authentication failed: Failed to get MSI token for account`

`d94c0db6-5540-438c-9eb3-aa20e02e1226 and resource https://management.core.windows.net/. Status: 500 (Internal Server Error)`

Causa

Se puede producir en los casos siguientes:

- **Causa 1**: usa la identidad administrada del sistema de la cuenta de Automation, que todavía no ha sido creada, y `Code Connect-AzAccount -Identity` intenta autenticarse en Azure y ejecutar un runbook en Azure o en un trabajo híbrido de runbook.
- **Causa 2**: la cuenta de Automation tiene asignada una identidad administrada por el usuario y no una identidad administrada por el sistema, y `Code Connect-AzAccount -Identity` intenta autenticarse en Azure y ejecutar un runbook en un trabajo híbrido de runbook de la máquina virtual de Azure mediante la identidad administrada por el sistema de máquina virtual de Azure.

Solución

- **Resolution 1**: debe crear la identidad administrada del sistema de cuentas de Automation y concederle acceso a los recursos de Azure.
- **Resolución 2**: según sean sus requisitos, puede:
  - Crear la identidad administrada por el sistema de la cuenta de Automation y usarla para autenticarse.
    O bien
  - Eliminar la identidad administrada asignada por el usuario para una cuenta de Azure Automation.

Escenario: No se puede encontrar la identidad administrada asignada por el usuario para agregarla a la cuenta de Automation

Problema

Desea agregar una identidad administrada asignada por el usuario a la cuenta de Automation. Sin embargo, no puede encontrar la cuenta en la hoja de Automation.

Causa

Este problema se produce cuando no tiene los permisos siguientes para la identidad administrada asignada por el usuario para poder verla en la hoja de Automation.

- `Microsoft.ManagedIdentity/userAssignedIdentities/*/read`
- `Microsoft.ManagedIdentity/userAssignedIdentities/*/assign/action`

> **Nota:**
>
> Los permisos anteriores se conceden de forma predeterminada al operador de identidad administrada y al colaborador de identidad administrada.

Solución

Asegúrese de que tiene el permiso de rol operador de identidad para agregar la identidad administrada asignada por el usuario a la cuenta de Automation.

Escenario: se produce un error en el runbook con el mensaje de error "this.Client.SubscriptionId no puede ser null".

Problema

Su runbook mediante una identidad administrada Connect-AzAccount -Identity que intenta administrar objetos de Azure, no funciona correctamente y registra el siguiente error: `this.Client.SubscriptionId cannot be null.`

error

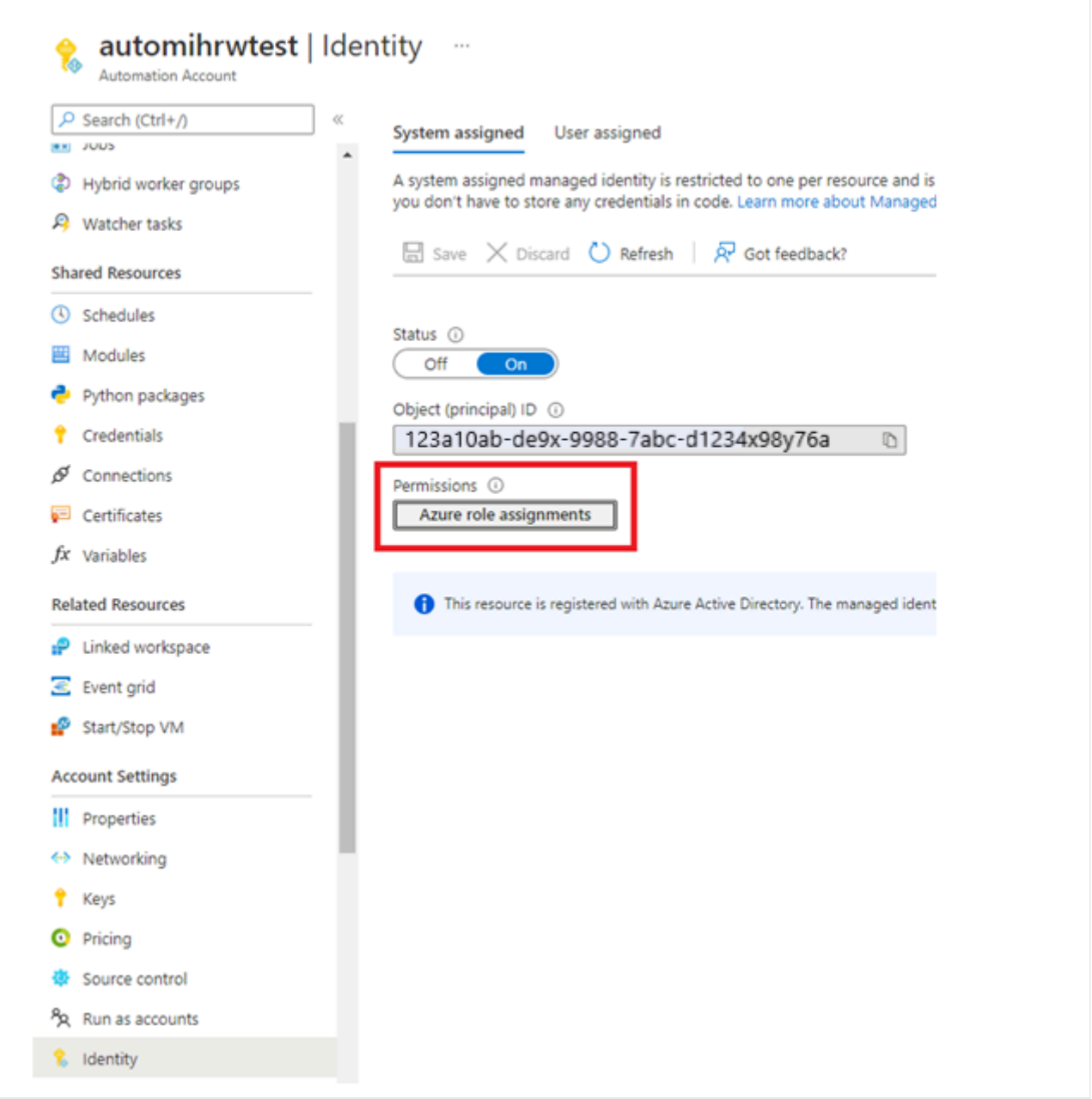
```
get-azvm : 'this.Client.SubscriptionId' cannot be null. At line:5 char:1 + get-azvm +
~~~~~~~~ + CategoryInfo : CloseError: (:) [Get-AzVM], ValidationException +
FullyQualifiedErrorId : Microsoft.Azure.Commands.Compute.GetAzureVMCommand
```

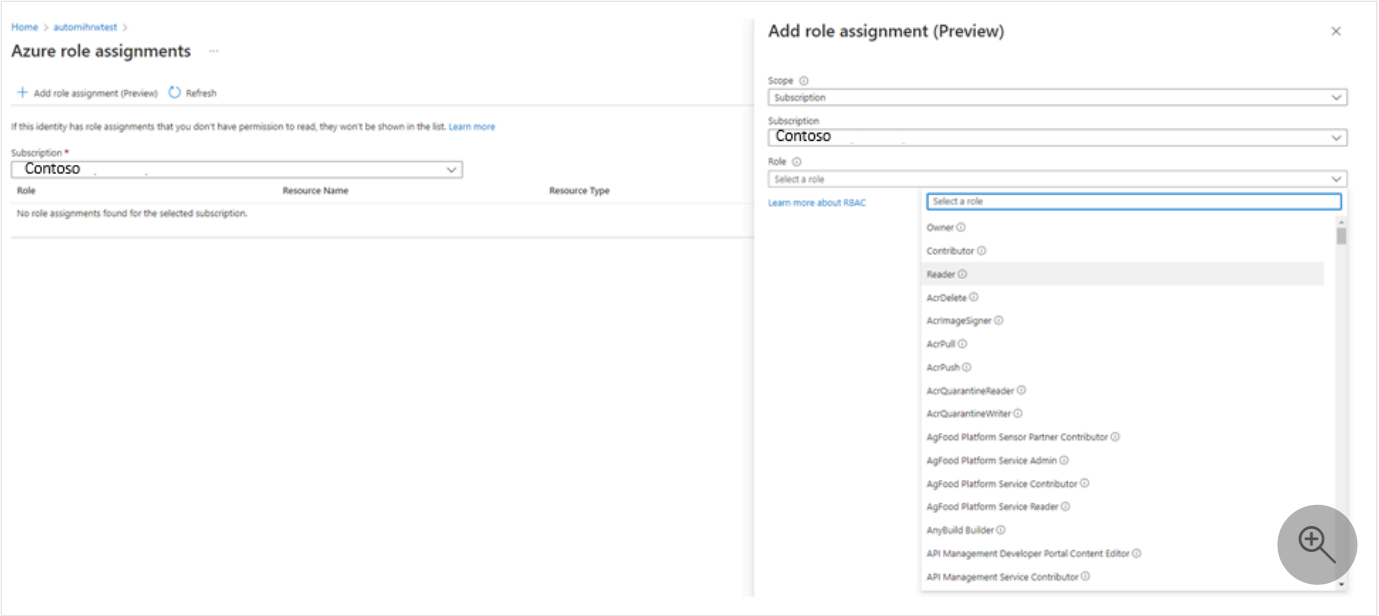
Causa

Esto puede ocurrir cuando no se ha concedido ningún permiso a la identidad administrada (u otra cuenta usada en el runbook) para acceder a la suscripción.

Solución

Conceda a la identidad administrada (u otra cuenta usada en el runbook) una pertenencia a roles adecuada en la suscripción. Más información





Escenario: no se puede obtener el token de MSI para la cuenta

Problema

Al trabajar con una identidad administrada asignada por el usuario en la cuenta de Automation, recibe un error similar al siguiente: `Failed to get MSI token for account a123456b-1234-12a3-123a-aa123456aa0b`.

Causa

Usar una identidad administrada asignada por el usuario antes de habilitar una identidad administrada asignada por el sistema para la cuenta de Automation.

Solución

Habilite una identidad administrada asignada por el sistema para la cuenta de Automation. A continuación, utilice la identidad administrada asignada por el usuario.

Escenario: Se produce un error al intentar usar la identidad administrada con la cuenta de Automation

Problema

Al intentar trabajar con identidades administradas en la cuenta de Automation, se produce un error como el siguiente:

error

```
Connect-AzureRMAccount : An error occurred while sending the request. At line:2 char:1
+ Connect-AzureRMAccount -Identity +
CategoryInfo : CloseError: (:) [Connect-AzureRmAccount], HttpRequestException +
FullyQualifiedErrorId : Microsoft.Azure.Commands.Profile.ConnectAzureRmAccountCommand
```

Causa

La causa más común es que no haya habilitado la identidad antes de intentar usarla. Para comprobarlo, ejecute el siguiente runbook de PowerShell en la cuenta de Automation afectada.

PowerShell

```
$resource= "?resource=https://management.azure.com/"
$url = $env:IDENTITY_ENDPOINT + $resource
$Headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"
$Headers.Add("X-IDENTITY-HEADER", $env:IDENTITY_HEADER)
$Headers.Add("Metadata", "True")

try
{
    $Response = Invoke-RestMethod -Uri $url -Method 'GET' -Headers $Headers
}
catch
{
    $StatusCode = $_.Exception.Response.StatusCode.value__
    $stream = $_.Exception.Response.GetResponseStream()
    $reader = New-Object System.IO.StreamReader($stream)
    $responseBody = $reader.ReadToEnd()

    Write-Output "Request Failed with Status: $StatusCode, Message: $responseBody"
}
```

Si el problema es que no habilitó la identidad antes de intentar usarla, debería ver un resultado similar a este:

`Request Failed with Status: 400, Message: {"Message":"No managed identity was found for Automation account xxxxxxxxxxxx"}`

Solución

Antes de poder usar el servicio de identidad administrada, debe habilitar una identidad para la cuenta de Automation. Vea Habilitar una identidad gestionada para su cuenta de Azure Automation

Pasos siguientes

Si en este artículo no se resuelve la incidencia, pruebe uno de los canales siguientes para obtener ayuda adicional:

- Recibe respuestas de expertos de Azure a través de Azure Forums.
- Conéctese con @AzureSupport. Esta es la cuenta oficial Microsoft Azure para conectar la comunidad de Azure a los recursos adecuados: respuestas, apoyo y expertos.
- Si tiene previsto abrir un caso de soporte técnico para un problema sin resolver, recopile los datos de diagnóstico necesarios antes de abrirlo. Consulte Datos para recopilar al abrir un caso para Microsoft Azure Automation.
- Registrar un incidente de soporte de Azure. Vaya al sitio Soporte técnico de Azure y seleccione **Get Support**.

Last updated on 11/06/2026

Solución de problemas de recursos compartidos

En este artículo se describen los problemas que pueden surgir al usar recursos compartidos en Azure Automation.

Módulos

Escenario: Un módulo se bloquea durante la importación

Problema

Un módulo está bloqueado en el estado *Importing* al importar o actualizar los módulos de Azure Automation.

Causa

Dado que la importación de módulos de PowerShell es un proceso complejo y de varios pasos, es posible que un módulo no se importe correctamente y se pueda bloquear en un estado transitorio. Para más información sobre el proceso de importación, consulte Importación de un módulo de PowerShell.

Solución

Para resolver este problema, debe eliminar el módulo que está bloqueado mediante el cmdlet Remove-AzAutomationModule. A continuación, puede volver a intentar importar el módulo.

Azure PowerShell

```
Remove-AzAutomationModule -Name ModuleName -ResourceGroupName ExampleResourceGroup -
AutomationAccountName ExampleAutomationAccount -Force
```

Escenario: Los módulos de AzureRM se bloquean durante la importación después de un intento de actualización

Problema

Después de intentar actualizar los módulos de AzureRM, en la cuenta permanece un banner con el siguiente mensaje:

```
error
Azure modules are being updated
```

Causa

Hay un problema conocido con la actualización de los módulos de AzureRM en una cuenta de Automation. En concreto, el problema se produce si los módulos se encuentran en un grupo de recursos con un nombre numérico que empieza por 0.

Solución

Para actualizar los módulos de AzureRM en la cuenta de Automation, la cuenta debe estar en un grupo de recursos con un nombre alfanumérico. En este momento, los grupos de recursos con nombres numéricos que empiezan por 0 no pueden actualizar módulos de AzureRM.

Escenario: No se puede importar el módulo o no se puede ejecutar cmdlets después de la importación

Problema

No se puede importar un módulo, o bien se importa correctamente, pero no se extrae ningún cmdlet.

Causa

Algunas razones comunes por las que es posible que un módulo no se importe correctamente a Azure Automation son:

- La estructura no coincide con la que necesita Automation.
- El módulo depende de otro módulo que no se ha implementado en su cuenta de Automation.
- Al módulo le faltan sus dependencias en la carpeta.
- El cmdlet New-AzAutomationModule se usa para cargar el módulo y no se ha proporcionado la ruta de acceso de almacenamiento completa o no se ha cargado el módulo mediante una dirección URL de acceso público.

Solución

Use cualquiera de estas soluciones para corregir el problema:

- Asegúrese de que el módulo sigue el formato siguiente: NombreMódulo.Zip -> NombreMódulo o número de versión -> (NombreMódulo.psm1, NombreMódulo.psd1).
- Abra el archivo **.psd1** y compruebe si el módulo tiene dependencias. Si es así, cargue estos módulos en la cuenta de Automation.
- Asegúrese de que todos los archivos **.dll** a los que se hace referencia están presentes en la carpeta del módulo.

Escenario: Update-AzureModule.ps1 se suspende durante la actualización de módulos

Problema

Al usar el runbook Update-AzureModule.ps1 para actualizar los módulos de Azure, se suspende el proceso de actualización del módulo.

Causa

En este runbook, el valor predeterminado para determinar cuántos módulos se actualizan simultáneamente es de 10. El proceso de actualización es propenso a errores cuando se actualizan demasiados módulos al mismo tiempo.

Solución

No es habitual que se necesiten todos los módulos de AzureRM o Az en la misma cuenta de Automation. Se recomienda importar únicamente los módulos específicos que necesite.

> **Nota:**
>
> Evite importar todo el módulo `Az.Automation` o `AzureRM.Automation`, que importa todos los módulos contenidos.

Si se suspende el proceso de actualización, agregue el parámetro `SimultaneousModuleImportJobCount` al script **Update-AzureModules.ps1** y proporcione un valor menor que el predeterminado de 10. Se recomienda que, si implementa esta lógica, comience por un valor de 3 o 5. `SimultaneousModuleImportJobCount` es un parámetro del runbook del sistema **Update-AutomationAzureModulesForAccount** que se usa para actualizar los módulos de Azure. Si realiza este ajuste, el proceso de actualización se ejecuta durante más tiempo, pero

tiene más probabilidades de completarse. En el ejemplo siguiente se muestra el parámetro y su ubicación en el runbook:

PowerShell

```
$Body = @"
   {
      "properties":{
      "runbook":{
            "name":"Update-AutomationAzureModulesForAccount"
      },
      "parameters":{
            ...
            "SimultaneousModuleImportJobCount":"3",
            ...
      }
      }
   }
"@
```

Cuentas de ejecución

> ⓘ **Nota:**
>
> Las cuentas de ejecución de Azure Automation, incluidas las cuentas de ejecución clásicas, se retiraron el **30 de septiembre de 2023** y se reemplazaron por Identidades Administradas. Ya no podrá crear ni renovar cuentas de ejecución a través del portal de Azure. Para obtener más información, consulta cómo migrar de cuentas 'Run As' existentes a una identidad administrada.

Escenario: Aparece el error "No se puede encontrar el punto de entrada denominado 'GetPerAdapterInfo' en el archivo DLL 'iplpapi.dll'" al ejecutar un runbook.

Problema

Al ejecutar un runbook aparece la siguiente excepción:

error

```
Unable to find an entry point named 'GetPerAdapterInfo' in DLL 'iplpapi.dll'
```

Causa

Este error suele deberse a una configuración incorrecta de la cuenta de ejecución.

Solución

Asegúrese de que su cuenta de 'Ejecutar como' está configurada correctamente. A continuación, compruebe que tiene el código adecuado en el runbook para autenticarse con Azure. En el ejemplo siguiente se muestra un fragmento de código para realizar la autenticación en Azure en un libro de ejecución mediante una cuenta Run As.

PowerShell

```
$connection = Get-AutomationConnection -Name AzureRunAsConnection
Connect-AzAccount -ServicePrincipal -Tenant $connection.TenantID `
-ApplicationID $connection.ApplicationID -CertificateThumbprint
$connection.CertificateThumbprint
```

Pasos siguientes

Si en este artículo no se resuelve la incidencia, pruebe uno de los canales siguientes para obtener ayuda adicional:

- Recibe respuestas de expertos de Azure a través de Azure Forums.
- Conéctese con @AzureSupport. Esta es la cuenta oficial Microsoft Azure para conectar la comunidad de Azure a los recursos adecuados: respuestas, apoyo y expertos.
- Si tiene previsto abrir un caso de soporte técnico para un problema sin resolver, recopile los datos de diagnóstico necesarios antes de abrirlo. Consulte Datos para recopilar al abrir un caso para Microsoft Azure Automation.
- Registrar un incidente de soporte de Azure. Vaya al sitio Azure support y seleccione **Get Support**.

Last updated on 20/04/2026

Solución de incidencias de runbooks

En este artículo se describen los errores de runbook que pueden producirse y cómo resolverlos. Para obtener información general, consulte ejecución de Runbook en Azure Automation.

Los runbooks gráficos de flujo de trabajo de PowerShell con runbooks secundarios no se pueden ejecutar

Incidencia

Los runbooks gráficos de flujo de trabajo de PowerShell con runbooks secundarios no se pueden ejecutar según lo previsto.

Causa

Para mejorar la posición de seguridad de los runbooks gráficos de flujo de trabajo de PowerShell, el servicio ya no ejecuta runbooks gráficos de flujo de trabajo de PowerShell con scripts secundarios.

Resolución

La solución consiste en usar Start-AzAutomationRunbook (desde el módulo Az.Automation) desde el runbook primario para iniciar el runbook secundario. Por ejemplo, use el bloque InlineScript:

```
$job = Start-AzAutomationRunbook `
    -AutomationAccountName "MyAccount" `
    -ResourceGroupName "MyRG" `
    -Name "ReusableTaskRunbook" `
    -Parameters @{ TaskId = '1234' }


#Optional: Wait for job completion
```

```
do {

    Start-Sleep -Seconds 5

    $jobStatus = Get-AzAutomationJob -Id $job.Id -AutomationAccountName "MyAccount" -
ResourceGroupName "MyRG"} while ($jobStatus.Status -ne "Completed")
```

Ya no se pueden usar cmdlets de módulos no predeterminados importados en runbooks gráficos de PowerShell.

Incidencia

Al importar un módulo de PowerShell, no podrá usar sus cmdlets en runbooks gráficos de PowerShell.

Causa

Para mejorar la posición de seguridad de los runbooks de PowerShell, el servicio ha dejado de procesar el archivo de manifiesto del módulo para exportar los cmdlets y las funciones, Esto significa que no se pueden usar al crear runbooks gráficos de PowerShell.

Resolución

No hay ningún impacto en la ejecución de runbooks existentes. En el caso de los nuevos runbooks que usen módulos de PowerShell no predeterminados, para solucionar este problema se recomienda usar runbooks textuales, en lugar de runbooks gráficos de PowerShell. Puede usar la extensión Azure Automation para VS Code para crear y editar runbooks de PowerShell, que aprovecha GitHub Copilot para simplificar la experiencia de creación de runbook.

Se produce un error en Start-AzAutomationRunbook con el mensaje de error "runbookName no coincide con el patrón esperado"

Incidencia

Cuando se ejecuta `Start-AzAutomationRunbook` para iniciar runbooks específicos:

PowerShell

```
start-azautomationRunbook -Name "Test_2" -AutomationAccountName "AutomationParent" -
ResourceGroupName "AutomationAccount"
```

Se produce el siguiente error:

`Start-AzAutomationRunbook: "runbookname" does not match expected pattern '^[a-zA-Z]*-*[a-zA-Z0-9]*$'`

Causa

El código que se introdujo en la versión 1.9.0 del módulo Az.Automation comprueba los nombres de los runbooks para iniciar y marcar incorrectamente runbooks con varios caracteres "-" o con un carácter "_" en el nombre como no válido.

Solución alternativa

Se recomienda revertir a la versión 1.8.0 del módulo.

Resolución

Actualmente, estamos trabajando para implementar una corrección para solucionar este problema.

Diagnóstico de problemas de runbooks

Cuando reciba errores durante la ejecución del runbook en Azure Automation, puede usar los pasos siguientes para ayudar a diagnosticar los problemas:

1. Asegúrese de que el script de runbook se ejecuta correctamente en la máquina local.

   Para ver los módulos de referencia y aprendizaje de lenguaje, consulte la PowerShell Docs o Python Docs. La ejecución local del script puede detectar y resolver errores comunes, como:

   - Módulos que faltan
   - Errores de sintaxis
   - Errores lógicos

2. Investigue los flujos de errores del runbook.

Examine estos flujos en busca de mensajes específicos y compárelos con los errores documentados en este artículo.

3. Asegúrese de que los nodos y el área de trabajo de Automation tengan los módulos necesarios.

   Si el runbook importa módulos, compruebe que están disponibles para su cuenta de Automation mediante los pasos descritos en Importación de módulos. Actualice los módulos de PowerShell a la versión más reciente siguiendo las instrucciones en Actualizar los módulos de Azure PowerShell en Azure Automation. Para obtener más información de solución de problemas, consulte Solución de problemas de módulos.

4. Haga lo siguiente si el runbook se suspende o registra un error inesperado:

   - Renueve el webhook si está intentando usar un webhook expirado para iniciar el runbook.
   - Compruebe los estados de los trabajos para determinar los estados actuales del runbook y algunas posibles causas del problema.
   - Agregue datos de salida adicionales al runbook para identificar lo que sucede antes de que se suspenda el runbook.
   - Controle las excepciones producidas por su trabajo.

5. Realice este paso si el trabajo del runbook o el entorno de Hybrid Runbook Worker no responden.

   Si ejecuta los runbooks en una instancia de Hybrid Runbook Worker en lugar de Azure Automation, puede que necesite solucionar los propios problemas de Hybrid Worker.

Escenario: No se puede crear ningún trabajo de Automation en la región Oeste de Europa

Incidencia

Al crear nuevos trabajos de Automation, es posible que se experimente un retraso o un error en la creación de trabajos. Los trabajos programados se retirarán automáticamente y los trabajos ejecutados a través del portal se pueden retirar si se detecta un error.

Causa

Esto se debe a la carga elevada de los runbooks de los clientes mediante el servicio Automation en la región Oeste de Europa.

Resolución

Realice la siguiente acción si es factible según sus necesidades y entorno para reducir la posibilidad de error:

- Si usa una hora en punto para la creación del trabajo (las 12:00, 1:00, 2:00, etc.), normalmente en las horas en punto o medias horas, se recomienda mover la hora de inicio del trabajo a cinco minutos antes o después de las horas en punto o medias horas. Esto se debe a que la mayoría de los clientes usan el principio de la hora para la ejecución del trabajo, lo que aumenta drásticamente la carga en el servicio, mientras que la carga es relativamente baja en las otras franjas horarias.

Escenario: se produce un error en el runbook con el mensaje de error "this.Client.SubscriptionId no puede ser null"

Incidencia

El runbook con una identidad administrada Connect-AzAccount -Identity que intenta administrar objetos de Azure no funciona correctamente y registra el siguiente error: `this.Client.SubscriptionId cannot be null.`

error

```
get-azvm : 'this.Client.SubscriptionId' cannot be null. At line:5 char:1 + get-azvm +
~~~~~~~~ + CategoryInfo : CloseError: (:) [Get-AzVM], ValidationException +
FullyQualifiedErrorId : Microsoft.Azure.Commands.Compute.GetAzureVMCommand
```

Causa

Esto puede ocurrir cuando a la identidad administrada (u otra cuenta usada en el runbook) no se le ha concedido ningún permiso para acceder a la suscripción.

Resolución

Conceda a la identidad administrada (u otra cuenta usada en el runbook) una pertenencia a roles adecuada en la suscripción. Aprende más

automihrwtest | Identity
Automation Account

Search (Ctrl+/)

Hybrid worker groups
Watcher tasks

Shared Resources
Schedules
Modules
Python packages
Credentials
Connections
Certificates
Variables

Related Resources
Linked workspace
Event grid
Start/Stop VM

Account Settings
Properties
Networking
Keys
Pricing
Source control
Run as accounts
Identity

System assigned | User assigned

A system assigned managed identity is restricted to one per resource and is
you don't have to store any credentials in code. Learn more about Managed

Save | Discard | Refresh | Got feedback?

Status
Off | On

Object (principal) ID
123a10ab-de9x-9988-7abc-d1234x98y76a

Permissions
Azure role assignments

This resource is registered with Azure Active Directory. The managed ident

Home > automihrwtest >
Azure role assignments

Add role assignment (Preview) | Refresh

If this identity has role assignments that you don't have permission to read, they won't be shown in the list. Learn more

Subscription *
Contoso

Role	Resource Name	Resource Type
No role assignments found for the selected subscription.		

Add role assignment (Preview)

Scope
Subscription

Subscription
Contoso

Role
Select a role

Learn more about RBAC

Select a role
Owner
Contributor
Reader
AcrDelete
AcrImageSigner
AcrPull
AcrPush
AcrQuarantineReader
AcrQuarantineWriter
AgFood Platform Sensor Partner Contributor
AgFood Platform Service Admin
AgFood Platform Service Contributor
AgFood Platform Service Reader
AnyBuild Builder
API Management Developer Portal Content Editor
API Management Service Contributor

Escenario: acceso bloqueado a Azure Storage o Azure Key Vault o Azure SQL

Este escenario usa Azure Storage como ejemplo; sin embargo, la información es igualmente aplicable a Azure Key Vault y Azure SQL.

Incidencia

Si se intenta acceder a Azure Storage desde un Runbook, se produce un error similar al siguiente mensaje: `The remote server returned an error: (403) Forbidden. HTTP Status Code: 403 - HTTP Error Message: This request is not authorized to perform this operation.`

Causa

El Azure Firewall en Azure Storage está habilitado.

Resolución

Habilitar el Azure Firewall en Azure Storage, Azure Key Vault o Azure SQL bloquea el acceso de los runbooks de Azure Automation para esos servicios. El acceso se bloqueará incluso cuando la excepción del firewall para permitir los servicios de Microsoft de confianza esté habilitada, ya que Automatización no forma parte de la lista de servicios de confianza. Con un firewall habilitado, el acceso solo se puede realizar mediante hybrid Runbook Worker y un punto de conexión de servicio de red virtual.

Escenario: Se produce un error en el runbook del tipo Sin permisos o Prohibido (403).

Incidencia

Se produce un error en el runbook del tipo Sin permisos, Prohibido (403) o equivalente.

Causa

Es posible que las cuentas de ejecución no tengan los mismos permisos en los recursos de Azure que su cuenta de Automation actual.

Resolución

Asegúrese de que la cuenta de ejecución tiene permisos para acceder a los recursos usados en el script.

Escenario: error al iniciar sesión en Azure cuenta

Incidencia

Se produce uno de los siguientes errores al trabajar con el cmdlet `Connect-AzAccount`:

error

```
Unknown_user_type: Unknown User Type
```

error

```
No certificate was found in the certificate store with thumbprint
```

Causa

Estos errores se producen si el nombre de recurso de credencial no es válido. También pueden producirse si el nombre de usuario y la contraseña que usó para configurar el recurso de credencial de Automation no son válidos.

Resolución

Para determinar cuál es el problema, siga estos pasos:

1. Asegúrese de no haber incluido ningún carácter especial. Estos caracteres incluyen el carácter `\@` en el nombre del recurso de credenciales de Automation que usa para conectarse a Azure.

2. Compruebe si puede usar el nombre de usuario y la contraseña que se almacenan en la credencial de Azure Automation en el editor de PowerShell ISE local. Ejecute los siguientes cmdlets en el ISE de PowerShell.

   PowerShell

   ```
   $Cred = Get-Credential
   #Using Azure Service Management
   Add-AzureAccount -Credential $Cred
   #Using Azure Resource Manager
   Connect-AzAccount -Credential $Cred
   ```

3. Si se produce un error en la autenticación localmente, no ha configurado correctamente las credenciales de Microsoft Entra. Para configurar correctamente la cuenta de Microsoft Entra, consulte el artículo Authenticate para Azure con Microsoft Entra ID.

4. Si el error parece ser transitorio, intente agregar lógica de reintento a su rutina de autenticación para hacer de la autenticación un proceso más sólido.

PowerShell

```
$logonAttempt = 0
$logonResult = $False

while(!($connectionResult) -And ($logonAttempt -le 10))
{
    $LogonAttempt++
    #Logging in to Azure...
    $connectionResult = Connect-AzAccount `
    Start-Sleep -Seconds 30
    if($connectionResult)
    {
       $logonResult = $True
    }
}
```

Escenario: Ejecución de Login-AzureRMAccount para iniciar sesión

Incidencia

Recibe el error siguiente al ejecutar un runbook:

error

```
Run Login-AzureRMAccount to login.
```

Causa

Este error puede producirse cuando no usa una cuenta de ejecución o esta ha expirado.

Este error tiene dos causas principales:

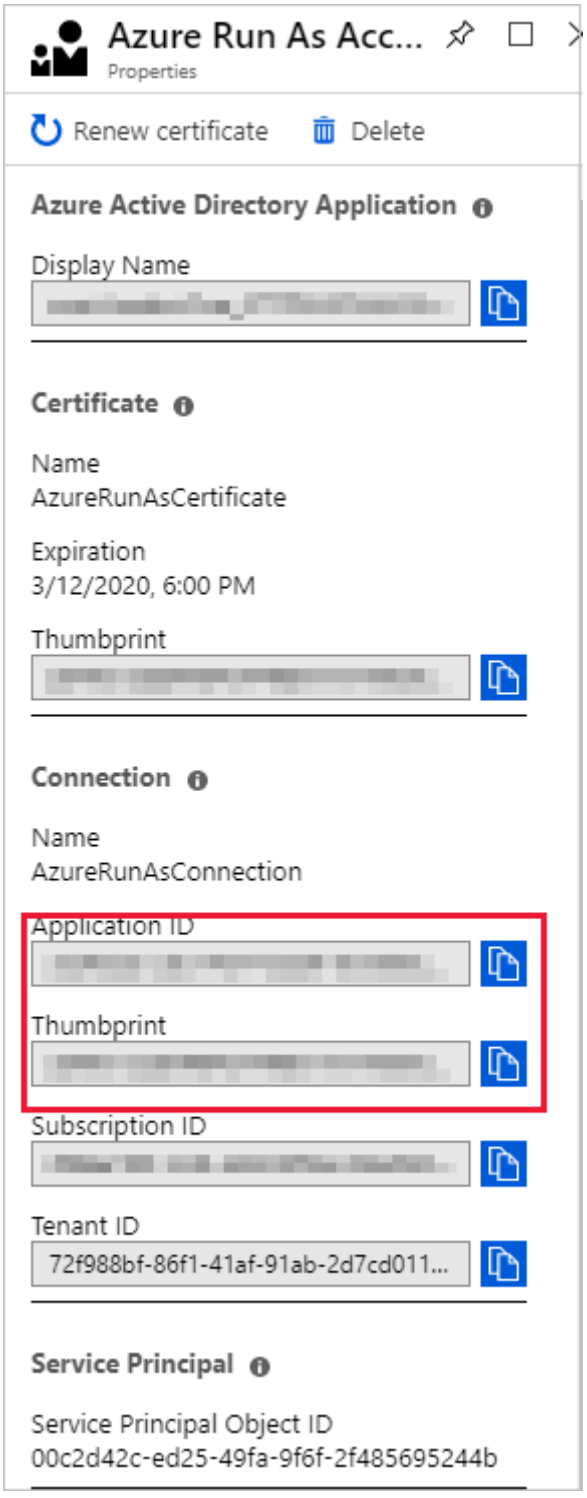
- Hay versiones diferentes del módulo de AzureRM o Az.
- Intenta acceder a los recursos de otra suscripción.

Resolución

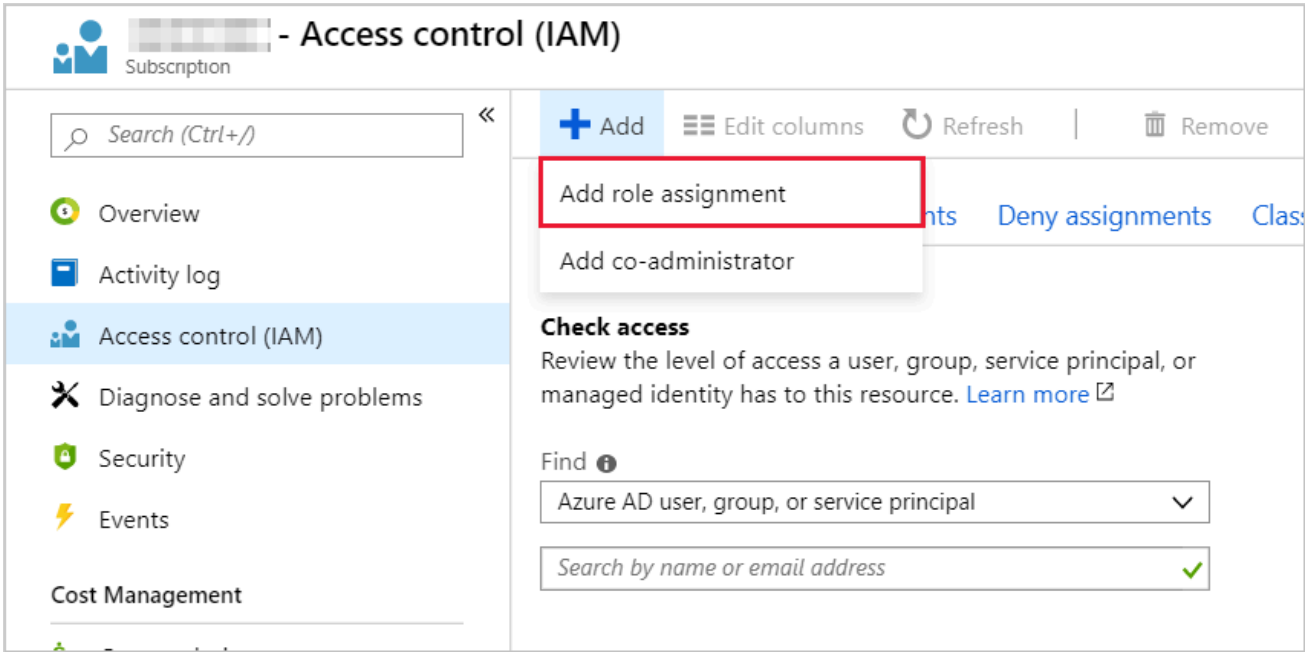
Si recibe este error después de actualizar un módulo de AzureRM o Az, actualice todos los módulos a la misma versión.

Si intenta acceder a los recursos de otra suscripción, siga estos pasos para configurar los permisos:

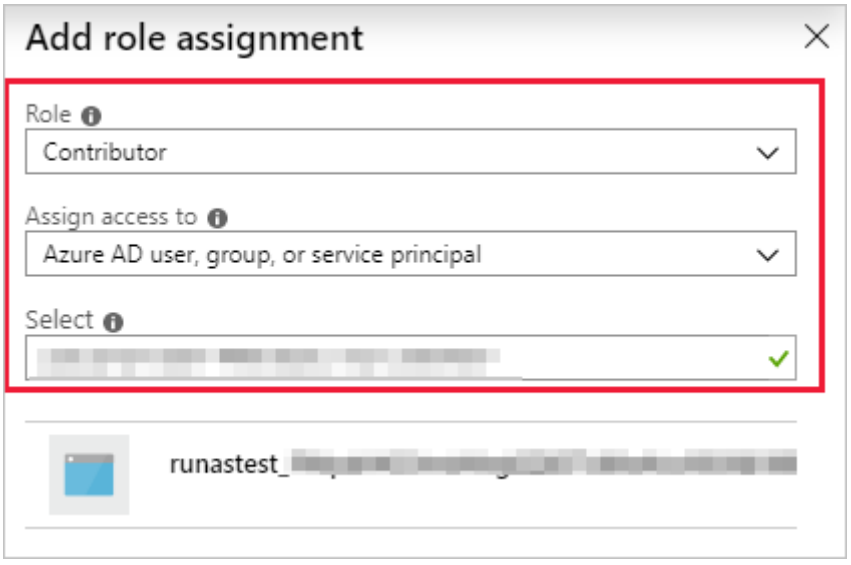
1. Vaya a la cuenta de ejecución de Automation y copie los valores de **Application ID** (Id. de aplicación) y **Thumbprint** (Huella digital).



2. Vaya a la opción **Access control** (Control de acceso) de la suscripción donde la cuenta de Automation *no* está hospedada y agregue una asignación de roles nueva.



3. Agregue el **identificador de aplicación** recopilado anteriormente. Seleccione **Permisos de colaborador** .



4. Copie el nombre de la suscripción.

5. Ahora puede usar el código del runbook siguiente para probar los permisos de la cuenta de Automation en la otra suscripción. Reemplace `<CertificateThumbprint>` por el valor que copió en el paso 1. Reemplace `"<SubscriptionName>"` por el valor que copió en el paso 4.

PowerShell

```
$Conn = Get-AutomationConnection -Name AzureRunAsConnection
Connect-AzAccount -ServicePrincipal -Tenant $Conn.TenantID -ApplicationId
$Conn.ApplicationID -CertificateThumbprint "<CertificateThumbprint>"
```

```
#Select the subscription you want to work with
Select-AzSubscription -SubscriptionName '<YourSubscriptionNameGoesHere>'

#Test and get outputs of the subscriptions you granted access.
$subscriptions = Get-AzSubscription
foreach($subscription in $subscriptions)
{
    Set-AzContext $subscription
    Write-Output $subscription.Name
}
```

Escenario: No se puede encontrar la suscripción de Azure

Incidencia

Recibirá el siguiente error al trabajar con el cmdlet `Select-AzureSubscription`, `Select-AzureRMSubscription` o `Select-AzSubscription`:

error

```
The subscription named <subscription name> cannot be found.
```

Error

Este error se puede generar si:

- El nombre de la suscripción no es válido.
- El Microsoft Entra usuario que intenta obtener los detalles de la suscripción no está configurado como administrador de la suscripción.
- El cmdlet no está disponible.
- Se produjo un cambio de contexto.

Resolución

Para la conmutación de contexto, consulte Context switching in Azure Automation.

Escenario: se produce un error en los runbooks cuando se trabaja con varias suscripciones

Incidencia

Al ejecutar los runbooks, el runbook no puede administrar recursos de Azure.

Causa

El runbook no está utilizando el contexto correcto cuando se ejecuta. Esto puede deberse a que el runbook intenta tener acceso accidentalmente a la suscripción incorrecta.

Es posible que vea errores como este:

error

```
Get-AzVM : The client '<client-id>' with object id '<object-id> does not have
authorization to perform action 'Microsoft.Compute/virtualMachines/read' over scope
'/subscriptions/<subscriptionIdOfSubscriptionWhichDoesntContainTheVM>/resourceGroups/R
EsourceGroupName/providers/Microsoft.Compute/virtualMachines/VMName '.
   ErrorCode: AuthorizationFailed
   StatusCode: 403
   ReasonPhrase: Forbidden Operation
   ID : <AGuidRepresentingTheOperation> At line:51 char:7 + $vm = Get-AzVM -
ResourceGroupName $ResourceGroupName -Name $UNBV... +
```

O como este:

error

```
Get-AzureRmResource : Resource group "SomeResourceGroupName" could not be found.
... resources = Get-AzResource -ResourceGroupName $group.ResourceGro ...
                  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : CloseError: (:) [Get-AzResource], CloudException
    + FullyQualifiedErrorId :
Microsoft.Azure.Commands.ResourceManager.Cmdlets.Implementation.GetAzureResourceCmdlet
```

Resolución

Para evitar intentar acceder accidentalmente a la suscripción incorrecta, consulte cambio de contexto en Azure Automation.

Escenario: se produce un error en la autenticación para Azure porque la autenticación multifactor está habilitada

Incidencia

Recibe el siguiente error al autenticarse en Azure con el nombre de usuario y la contraseña de Azure:

```
error
Add-AzureAccount: AADSTS50079: Strong authentication enrollment (proof-up) is required
```

Causa

Si tiene autenticación multifactor en la cuenta de Azure, no puede usar un usuario de Microsoft Entra para autenticarse en Azure. En su lugar, tiene que usar un certificado o una entidad de servicio para autenticarse.

Resolución

Para usar una entidad de servicio con cmdlets de Azure Resource Manager, consulte Crear entidad de servicio mediante Azure portal y Authenticar una entidad de servicio con Azure Resource Manager.

Escenario: Se produce un error en un runbook con el mensaje "Se canceló una tarea"

Incidencia

El runbook genera un error similar al ejemplo siguiente:

```
error
Exception: A task was cancelled.
```

Causa

Este error puede deberse al uso de módulos de Azure obsoletos.

Resolución

Para resolver este error, actualice los módulos de Azure a la versión más reciente:

1. En la cuenta de Automation, seleccione **Modules** y seleccione **Actualizar Azure módulos**.

2. La actualización tarda aproximadamente 15 minutos. Una vez finalizada, vuelva a ejecutar el runbook en el que se produjo el error.

Para obtener más información sobre cómo actualizar los módulos, consulte Actualizar módulos de Azure en Azure Automation.

Escenario: Término no reconocido como nombre de un cmdlet, función o script

Incidencia

El runbook genera un error similar al ejemplo siguiente:

error

```
The term 'Connect-AzAccount' is not recognized as the name of a cmdlet, function,
script file, or operable program. Check the spelling of the name, or if the path was
included verify that the path is correct and try again.
```

Causa

Este error puede deberse a los siguientes motivos:

- El módulo que contiene el cmdlet no está importado en la cuenta de Automation.
- El módulo que contiene el cmdlet está importado, pero no está actualizado.

Resolución

Realice una de las siguientes tareas para solucionar este error:

- Para obtener un módulo de Azure, consulte Cómo actualizar módulos de Azure PowerShell en Azure Automation para obtener información sobre cómo actualizar los módulos en su cuenta de Automation.
- Para un módulo que no sea de Azure, asegúrese de que el módulo se importa en la cuenta de Automation.

Escenario: el cmdlet produce un error en el Runbook de PnP PowerShell en Azure Automation

Incidencia

Cuando un runbook escribe un objeto PnP generado por PowerShell directamente en la salida de Azure Automation, la salida del cmdlet no puede volver a transmitirse a Automation.

Causa

Este problema normalmente se produce cuando Azure Automation procesa runbooks que invocan cmdlets de PowerShell de PnP; por ejemplo, `add-pnplistitem`, sin detectar los objetos devueltos.

Resolución

Edite los scripts para asignar los valores devueltos a variables de forma que los cmdlets no intenten escribir objetos completos en la salida estándar. Un script puede redirigir el flujo de salida a un cmdlet, como se muestra a continuación.

Azure CLI

```
$null = add-pnplistitem
```

Si el script analiza la salida del cmdlet, el script debe almacenar la salida en una variable y manipular la variable en lugar de simplemente transmitir la salida.

Azure CLI

```
$SomeVariable = add-pnplistitem ....
if ($SomeVariable.someproperty -eq ....
```

Escenario: No se reconoce el Cmdlet cuando se ejecuta un runbook

Incidencia

Se produce un error en el trabajo de runbook con el error:

error

```
<cmdlet name>: The term <cmdlet name> is not recognized as the name of a cmdlet,
function, script file, or operable program.
```

Causa

Este error se produce cuando el motor de PowerShell no puede encontrar el cmdlet que está usando en su runbook. Es posible que el módulo que contiene el cmdlet no esté presente en la cuenta, que haya un conflicto de nombres con un nombre de runbook o que el cmdlet también exista en otro módulo y Automation no pueda resolver el nombre.

Resolución

Use cualquiera de las siguientes soluciones para solucionar el problema:

- Asegúrese de que ha especificado correctamente el nombre del cmdlet.
- Asegúrese de que el cmdlet exista en su cuenta de Automation y de que no haya ningún conflicto. Para comprobar si está presente el cmdlet, abra un runbook en modo de edición y busque el cmdlet que quiere encontrar en la biblioteca o ejecute `Get-Command <CommandName>`. Una vez que haya validado que el cmdlet está disponible para la cuenta y que no hay conflictos de nombres con otros cmdlets o runbooks, agregue el cmdlet al lienzo. Asegúrese de que usa un conjunto válido de parámetros en el runbook.
- Si tiene un conflicto de nombres y el cmdlet está disponible en dos módulos diferentes, use el nombre completo del cmdlet para resolver este problema. Por ejemplo, puede usar `ModuleName\CmdletName`.
- Si está ejecutando el runbook en el entorno local de un grupo de Hybrid Worker, asegúrese de que el cmdlet o el módulo están instalados en la máquina que hospeda la instancia de Hybrid Worker.

Escenario: referencia de objeto incorrecta en la llamada a Add-AzAccount

Incidencia

Este error aparece cuando se trabaja con `Add-AzAccount`, que es un alias del cmdlet `Connect-AzAccount`:

error

```
Add-AzAccount : Object reference not set to an instance of an object
```

Causa

Este error puede producirse si el runbook no realiza los pasos adecuados antes de llamar a `Add-AzAccount` para agregar la cuenta de Automation. Un ejemplo de uno de los pasos necesarios es

iniciar sesión con una cuenta de ejecución. Para ver las operaciones correctas que se deben usar en el runbook, consulte Ejecución de un runbook en Azure Automation.

Escenario: Referencia a objeto no establecida como instancia de un objeto

Incidencia

Recibe el siguiente error cuando invoca un runbook secundario con el parámetro `Wait` y el flujo de salida contiene un objeto:

error

```
Object reference not set to an instance of an object
```

Causa

Si el flujo contiene objetos, significa que `Start-AzAutomationRunbook` no controla el flujo de salida correctamente.

Resolución

Implemente una lógica de sondeo y use el cmdlet Get-AzAutomationJobOutput para recuperar la salida. Aquí se define un ejemplo de esta lógica:

PowerShell

```
$AutomationAccountName = "ContosoAutomationAccount"
$RunbookName = "ChildRunbookExample"
$ResourceGroupName = "ContosoRG"

function IsJobTerminalState([string]$Status) {
  $TerminalStates = @("Completed", "Failed", "Stopped", "Suspended")
  return $Status -in $TerminalStates
}

$StartAzAutomationRunbookParameters = @{
  Name = $RunbookName
  AutomationAccountName = $AutomationAccountName
  ResourceGroupName = $ResourceGroupName
}
$Job = Start-AzAutomationRunbook @StartAzAutomationRunBookParameters
$PollingSeconds = 5
$MaxTimeout = New-TimeSpan -Hours 3 | Select-Object -ExpandProperty TotalSeconds
$WaitTime = 0
```

```
while(-NOT (IsJobTerminalState $Job.Status) -and $WaitTime -lt $MaxTimeout) {
   Start-Sleep -Seconds $PollingSeconds
   $WaitTime += $PollingSeconds
   $Job = $Job | Get-AzAutomationJob
}

$Job | Get-AzAutomationJobOutput | Get-AzAutomationJobOutputRecord | Select-Object -
ExpandProperty Value
```

Escenario: error en runbook debido a un objeto deserializado

Incidencia

Se produce un error en el runbook con el error:

error

```
Cannot bind parameter <ParameterName>.

Cannot convert the <ParameterType> value of type Deserialized <ParameterType> to type
<ParameterType>.
```

Causa

Si el runbook es un flujo de trabajo de PowerShell, almacena objetos complejos en un formato deserializado para conservar el estado del Runbook si se suspende el flujo de trabajo.

Resolución

Use cualquiera de las siguientes soluciones para corregir este problema:

- Si canaliza objetos complejos de un cmdlet a otro, encapsule dichos cmdlets en una actividad `InlineScript`.
- En lugar de pasar el objeto complejo entero, pase solamente el nombre o valor del mismo que necesite.
- Use un runbook de PowerShell en lugar de un runbook de flujo de trabajo de PowerShell.

Escenario: Estado de solicitud incorrecta 400 al llamar a un webhook

Incidencia

Al intentar invocar un webhook para un runbook de Azure Automation, recibirá el siguiente error:

```
error
400 Bad Request : This webhook has expired or is disabled
```

Causa

El webhook al que intenta llamar está deshabilitado o ha expirado.

Resolución

Si el webhook está deshabilitado, puede volver a habilitarlo a través del portal de Azure. Si el webhook expiró, debe eliminarlo y volver a crearlo. Solo puede renovar un webhook si aún no ha expirado.

Escenario: 429: Actualmente la tasa de solicitud es demasiado grande

Incidencia

Recibe el siguiente mensaje de error al ejecutar el cmdlet `Get-AzAutomationJobOutput`:

```
error
429: The request rate is currently too large. Please try again
```

Causa

Este error puede producirse al recuperar la salida de trabajo de un runbook que tiene muchos flujos detallados.

Resolución

Realice una de las siguientes acciones para solucionar este error:

- Edite el runbook y reduzca el número de flujos de trabajo que emite.
- Reduzca el número de flujos para recuperar cuando se ejecuta el cmdlet. Para ello, puede establecer el valor del `Stream` parámetro para el cmdlet Get-AzAutomationJobOutput para recuperar solo secuencias de salida.

Escenario: error en el trabajo del runbook porque se superó la cuota asignada

Incidencia

Se produce un error en el trabajo de runbook con el error:

error

```
The quota for the monthly total job run time has been reached for this subscription
```

Causa

Este error se produce cuando la ejecución del trabajo supera la cuota gratuita de 500 minutos para su cuenta. Esta cuota se aplica a todos los tipos de tareas de ejecución de trabajos. Algunas de estas tareas son probar un trabajo, iniciar un trabajo desde el portal, ejecutar un trabajo mediante webhooks o programar un trabajo para que se ejecute mediante el portal de Azure o el centro de datos. Para más información sobre los precios de Automation, consulte Precios de Automation.

Resolución

Si quiere usar más de 500 minutos de procesamiento por mes cambie la suscripción del nivel Gratis al nivel Básico:

1. Inicie sesión en la suscripción de Azure.
2. Seleccione la cuenta de Automation que quiera actualizar.
3. Seleccione **Configuración** y, a continuación, seleccione **Precios**.
4. Seleccione **Habilitar** en la parte inferior de la página para actualizar la cuenta al nivel Básico.

Escenario: flujo de salida de runbook superior a 1 MB

Incidencia

Se produce el siguiente error en el runbook que se ejecuta en el espacio aislado de Azure:

error

```
The runbook job failed due to a job stream being larger than 1MB, this is the limit
supported by an Azure Automation sandbox.
```

Causa

Este error se produce porque el runbook intentó escribir demasiados datos de excepción en el flujo de salida.

Resolución

Hay un límite de 1 MB en el flujo de salida del trabajo. Asegúrese de que el runbook incluye las llamadas a un archivo ejecutable o a un subproceso mediante bloques `try` y `catch`. Si las operaciones producen una excepción, haga que el código escriba el mensaje de la excepción en una variable de Automation. Esta técnica impedirá que el mensaje se escriba en el flujo de salida del trabajo. En el caso de trabajos ejecutados de Hybrid Runbook Worker, el flujo de salida truncado en 1 MB se muestra sin ningún mensaje de error.

Escenario: Se intentó iniciar el trabajo del runbook tres veces, pero en las tres se produjo un error

Incidencia

Se produce el siguiente error en el runbook:

error

```
The job was tried three times but it failed
```

Causa

Este error se produce debido a uno de los siguientes problemas:

- **Límite de memoria.** Un trabajo puede producir un error si emplea más de 400 MB de memoria. Los límites documentados sobre la memoria que se asigna a un espacio aislado se encuentran en Límites del servicio Automation.
- **Sockets de red.** Los espacios aislados de Azure están limitados a 1000 sockets de red simultáneos. Para más información, consulte Límites de servicio de Automatización.
- **Módulo incompatible.** Es posible que las dependencias del módulo no sean correctas. En este caso, el runbook suele devolver el mensaje `Command not found` o `Cannot bind parameter`.

- **No hay autenticación con Active Directory para el sandbox.** El runbook intentó llamar a un archivo ejecutable o a un subproceso que se ejecuta en un espacio aislado de Azure. No se admite la configuración de runbooks para autenticarse con Microsoft Entra ID mediante la biblioteca de autenticación de Azure Active Directory (ADAL).

Resolución

- **Límite de memoria, sockets de red.** Algunas formas sugeridas para trabajar dentro del límite de memoria son dividir la carga de trabajo entre varios runbooks, procesar menos datos en la memoria, evitar escribir resultados innecesarios de los runbooks y considerar cuántos puntos de control se escriben en los runbooks de flujo de trabajo de PowerShell. Use el método clear, como `$myVar.clear`, para borrar las variables y utilice `[GC]::Collect` para ejecutar inmediatamente la recolección de elementos no utilizados. Estas acciones reducen la superficie de memoria de su runbook en tiempo de ejecución.

- **Módulo incompatible.** Actualice los módulos de Azure siguiendo los pasos descritos en Cómo actualizar los módulos de Azure PowerShell en Azure Automation.

- **No hay autenticación con Active Directory para el sandbox.** Cuando realice la autenticación en Microsoft Entra ID con un runbook, asegúrese de que el módulo Azure AD esté disponible en su cuenta de automatización. Asegúrese de conceder a la cuenta de ejecución los permisos necesarios para realizar las tareas que el runbook automatiza.

  Si el runbook no puede llamar a un archivo ejecutable o a un subproceso que se ejecuta en un espacio aislado de Azure, use el runbook en una instancia de Hybrid Runbook Worker. Los trabajadores híbridos no están limitados por las restricciones de memoria y red que tienen los espacios aislados de Azure.

Escenario: Error en el trabajo de PowerShell con el mensaje de error "No se puede invocar el método"

Incidencia

Recibirá el siguiente mensaje de error al iniciar un trabajo de PowerShell en un runbook que se ejecuta en Azure:

error

```
Exception was thrown - Cannot invoke method. Method invocation is supported only on
core types in this language mode.
```

Causa

Este error podría indicar que los runbooks que se ejecutan en un espacio aislado de Azure no se pueden ejecutar en el modo de lenguaje completo Full Language mode.

Resolución

Hay dos maneras de resolver este error:

- En lugar de usar Start-Job, emplee Start-AzAutomationRunbook para iniciar el runbook.
- Intente ejecutar el runbook en una instancia de Hybrid Runbook Worker.

Para obtener más información sobre este comportamiento y otros comportamientos de runbooks de Azure Automation, consulte ejecución de Runbook en Azure Automation.

Escenario: un runbook de larga ejecución no se puede completar

Incidencia

El runbook muestra un estado Detenido después de ejecutarse durante tres horas. También se muestra este error:

error

```
The job was evicted and subsequently reached a Stopped state. The job cannot continue running.
```

Este comportamiento es así por naturaleza en los espacios aislados de Azure debido a la supervisión de distribución equilibrada de los procesos en Azure Automation. Si un proceso se ejecuta durante más de tres horas, la distribución equilibrada detiene automáticamente el runbook. El estado de un runbook que va más allá del límite de tiempo de distribución equilibrada varía según el tipo de runbook. Runbooks de PowerShell y Python se establecen en un estado Detenido. Los runbooks del flujo de trabajo de PowerShell se establecen en Error.

Causa

El runbook se ejecutó por encima del límite de tres horas permitido por la distribución equilibrada en un espacio aislado de Azure.

Resolución

La solución recomendada consiste en ejecutar el runbook en un Hybrid Runbook Worker. Los roles Hybrid Worker no tienen el límite de tres horas de los runbooks de distribución equilibrada que tienen los espacios aislados de Azure. Los runbooks que se ejecutan en instancias de Hybrid Runbook Worker se deben desarrollar para admitir comportamientos de reinicio si existen problemas inesperados con la infraestructura local.

Otra solución consiste en optimizar el runbook mediante la creación de runbooks secundarios. Si el runbook recorre en bucle la misma función en varios recursos (por ejemplo, una operación de base de datos en varias bases de datos), la función se puede mover a un runbook secundario. Cada runbook secundario se ejecuta en paralelo en un proceso independiente. Este comportamiento reduce la cantidad total de tiempo que tarda en completarse el runbook primario.

Los cmdlets de PowerShell que habilitan el escenario de runbook secundario son:

- Start-AzAutomationRunbook. este cmdlet permite iniciar un runbook y pasar parámetros al mismo.
- Get-AzAutomationJob. Si hay operaciones que deben realizarse después de que se complete el runbook secundario, este cmdlet permite comprobar el estado del trabajo de cada elemento secundario.

Escenario: Error en los flujos de trabajos sobre el método get_SerializationSettings

Incidencia

Se muestra el siguiente error en sus flujos de trabajo para un runbook:

error

```
Connect-AzAccount : Method 'get_SerializationSettings' in type
'Microsoft.Azure.Management.Internal.Resources.ResourceManagementClient' from assembly
'Microsoft.Azure.Commands.ResourceManager.Common, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35'
does not have an implementation.
At line:16 char:1
+ Connect-AzAccount -ServicePrincipal -Tenant $Conn.TenantID -Appl ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Connect-AzAccount], TypeLoadException
    + FullyQualifiedErrorId :
System.TypeLoadException,Microsoft.Azure.Commands.Profile.ConnectAzAccountCommand
```

Causa

Este error probablemente se deba a una migración incompleta de los módulos AzureRM a los módulos Az en el runbook. Esta situación puede hacer que Azure Automation inicie un trabajo de runbook solo con los módulos de AzureRM y, luego, otro trabajo solo con los módulos Az, lo que lleva a un bloqueo del espacio aislado.

Resolución

No se recomienda el uso de los cmdlets Az y AzureRM en el mismo runbook. Para obtener más información sobre el uso correcto de los módulos, consulte Migrar a módulos Az.

Escenario: Acceso denegado al usar el espacio aislado de Azure para un runbook o una aplicación

Incidencia

Cuando el runbook o la aplicación intentan ejecutarse en un espacio aislado de Azure, el entorno deniega el acceso.

Causa

Este problema puede producirse porque los entornos aislados de Azure impiden el acceso a todos los servidores COM externos. Por ejemplo, una aplicación en sandbox o un runbook no pueden llamar a Windows Management Instrumentation (WMI) ni al servicio de instalación de Windows (msiserver.exe).

Resolución

Para obtener más información sobre el uso de sandboxes de Azure, consulte Entorno de ejecución de Runbook.

Escenario: Código de estado no válido "Prohibido" al usar Key Vault dentro de un runbook

Incidencia

Al intentar acceder a Azure Key Vault a través de un runbook de Azure Automation, obtendrá el siguiente error:

```
error
Operation returned an invalid status code 'Forbidden'
```

Causa

Las posibles causas de este problema son:

- No usar una cuenta de ejecución.
- Permisos insuficientes.

Resolución

No usar una cuenta de ejecución.

Siga Paso 5 - agregue autenticación para administrar recursos de Azure para asegurarse de que usa una cuenta Run As para acceder a Key Vault.

Permisos insuficientes

Agregue permisos a Key Vault para asegurarse de que la cuenta Run As tenga permisos suficientes para acceder a Key Vault.

Escenario: Se produce el error "Se superó la longitud del parámetro" en el runbook

Incidencia

El runbook usa parámetros y presenta el siguiente error:

```
error
Total Length of Runbook Parameter names and values exceeds the limit of 30,000
characters. To avoid this issue, use Automation Variables to pass values to runbook.
```

Causa

Hay un límite para la longitud total de caracteres de todos los parámetros que se pueden proporcionar en Python 2.7, Python 3.8 y runbooks de PowerShell 7.1. La longitud total de todos los nombres de parámetros y los valores de parámetros no deben superar los 30 000 caracteres.

Resolución

Para solucionar este problema, puede usar Azure Automation Variables para pasar valores al runbook. Como alternativa, puede reducir el número de caracteres en Nombres de parámetro y valores de parámetro para asegurarse de que la longitud total no supere los 30 000 caracteres.

Documentos recomendados

- ejecución de Runbook en Azure Automation
- Inicio de un runbook en Azure Automation

Pasos siguientes

Si su problema no aparece o es incapaz de resolverlo, pruebe uno de los siguientes canales para obtener soporte técnico adicional:

- Recibe respuestas de expertos de Azure a través de Azure Forums.
- Conéctese con @AzureSupport, la cuenta oficial de Microsoft Azure para mejorar la experiencia del cliente. Azure soporte técnico le conecta a la comunidad de Azure para obtener respuestas, soporte técnico y expertos.
- Si tiene previsto abrir un caso de soporte técnico para un problema sin resolver, recopile los datos de diagnóstico necesarios antes de abrirlo. Consulte Datos para recopilar al abrir un caso para Microsoft Azure Automation.
- Registrar un incidente de soporte de Azure. Vaya al sitio Soporte técnico de Azure y seleccione **Get Support**.

Last updated on 20/04/2026

Solución de problemas de Hybrid Runbook Worker basado en extensiones de máquina virtual en Automation

En este artículo se proporciona información sobre cómo solucionar problemas con Azure Automation Hybrid Runbook Worker basado en extensiones. Para solucionar problemas de los trabajos basados en agente, consulte Solución de problemas de Hybrid Runbook Worker basado en agentes en Automation. Para obtener información general, consulte Información general sobre Hybrid Runbook Worker.

Lista de comprobación general

Para ayudar a solucionar problemas con instancias de Hybrid Runbook Worker basadas en extensiones, siga estos pasos:

- Compruebe que se admite el sistema operativo y que se han cumplido los requisitos previos. Consulte Requisitos previos.
- Compruebe que la identidad administrada asignada por el sistema esté habilitada en la máquina virtual. Las máquinas virtuales de Azure y las máquinas de Azure habilitadas para Arc deben estar configuradas con una identidad administrada asignada por el sistema.
- Compruebe que la extensión esté habilitada con la configuración correcta. El archivo de configuración debe tener el derecho `AutomationAccountURL`. Realice una comprobación cruzada de la dirección URL con la propiedad `AutomationHybridServiceUrl` de la cuenta de Automation.
  - Para Windows, puede encontrar el archivo de configuración aquí:

> **Sugerencia**
>
> Reemplace `*` en la ruta de acceso siguiente por la versión específica que esté instalada si la conoce.

```
C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindow
```

```
s\*\RuntimeSettings
```

- Para Linux, puede encontrar el archivo de configuración aquí:

```
/var/lib/waagent/Microsoft.Azure.Automation.HybridWorker.HybridWorkerForLinux/
```

- Compruebe el mensaje de error que se muestra en el estado de extensión o estado detallado de Hybrid Worker. Contiene mensajes de error y las recomendaciones correspondientes para corregir el problema.

- Ejecute la herramienta de solución de problemas en la máquina virtual y genere un archivo de salida. Abra el archivo de salida y compruebe los errores identificados por la herramienta.
  - Para Windows, puede encontrar el solucionador de problemas aquí:

> **Sugerencia**
>
> Reemplace `*` en la ruta de acceso siguiente por la versión específica que esté instalada si la conoce.

```
C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindow
s\*\bin\troubleshooter\TroubleShootWindowsExtension.ps1
```

  - Para Linux, puede encontrar el solucionador de problemas aquí:

> **Sugerencia**
>
> Reemplace `*` en la ruta de acceso siguiente por la versión específica que esté instalada si la conoce.

```
/var/lib/waagent/Microsoft.Azure.Automation.HybridWorker.HybridWorkerForLinux-
*/Troubleshooter/LinuxTroubleshooter.py
```

- En el caso de las máquinas Linux, la extensión de Hybrid Worker crea un usuario `hweautomation` e inicia Hybrid Worker en el usuario. Compruebe si el usuario `hweautomation` está configurado con los permisos correctos. Si el runbook está intentando acceder a los recursos locales, asegúrese de que `hweautomation` tiene los permisos correctos para los recursos locales.

- Compruebe si el proceso de Hybrid Worker se está ejecutando.
  - Para Windows, compruebe el servicio `Hybrid Worker Service` (***HybridWorkerService***).
  - Para Linux, compruebe el servicio `hwd`.

- Recopilación de registros:
  - Para Windows, ejecute la herramienta del recopilador de registros que se encuentra aquí:

    > **Sugerencia**
    >
    > Reemplace `*` en la ruta de acceso siguiente por la versión específica que esté instalada si la conoce.

    ```
    C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows\*\bin\troubleshooter\PullLogs.ps1
    ```

    Los registros se encontrarán aquí:

    ```
    C:\HybridWorkerExtensionLogs
    ```

  - Para Linux: los registros se encuentran en las carpetas siguientes:

    ```
    /var/log/azure/Microsoft.Azure.Automation.HybridWorker.HybridWorkerForLinux
    ```

    y

    ```
    /home/hweautomation
    ```

Escenario: los trabajos en Hybrid Worker de Linux pueden quedar bloqueados en estado en ejecución

Incidencia

Los trabajos en Hybrid Worker de Linux pueden quedar bloqueados en estado en ejecución mientras que el uso del núcleo de CPU es inferior al 25 %.

Causa

Cada núcleo de CPU tiene un límite de cuota predeterminado de 25% para Hybrid Worker de Linux.

Resolución

Puede quitar este límite y hacerlo sin restricciones con los pasos siguientes:

Cambiar a permisos de sudo -

1. unidad de búsqueda de sudo
2. systemctl status hwd.service // verifique y asegúrese de que el servicio hwd está funcionando correctamente.

Actualización de la configuración en el archivo siguiente en Hybrid Worker

1. vi /lib/systemd/system/hwd.service
2. Actualice la configuración de CPUQuota=25% a CPUQuota= como se muestra a continuación para que el uso no esté restringido.

```
  root@ubuntu2204:~# cat /lib/systemd/system/hwd.service
  [Unit]
  Description=HW Service
  After=network.target

  [Service]
  Type=simple
  ExecStart=/usr/bin/python3
/var/lib/waagent/Microsoft.Azure.Automation.HybridWorker.HybridWorkerForLinux-
1.1.16/HybridWorkerAgent/DaemonScripts/Scripts/3.x/automationWorkerStarterScript.py
  TimeoutStartSec=5
  Restart=always
  RestartSec=10s
  TimeoutStopSec=600
  CPUQuota=
```

```
KillMode=process

[Install]
WantedBy=multi-user.target
```

Reinicio del servicio hwd

1. systemctl daemon-reload
2. systemctl restart hwd.service

Escenario: los runbooks entran en un estado suspendido en una instancia de Hybrid Runbook Worker cuando se usa una cuenta personalizada en un servidor con control de cuentas de usuario (UAC) habilitado

Incidencia

Los trabajos producen un error y entran en un estado suspendido en Hybrid Runbook Worker. Los registros de eventos Microsoft-SMA indican `Win32 Process Exited with code [2148734720]` y un error correspondiente en el registro de aplicación cuando el runbook intenta ejecutarse es `.NET Runtime version : 4.0.30319.0` que indica que no se pudo iniciar la aplicación.

Causa

Cuando un sistema tiene UAC/LUA en su lugar, se deben conceder permisos directamente y no a través de ninguna pertenencia a grupos y cuando el usuario tiene que elevar permisos, los trabajos comienzan a producir errores.

Resolución

Para usuario personalizado en Hybrid Runbook Worker, actualice los permisos en las siguientes carpetas y registro:

Expandir tabla

Carpeta	Permisos
`C:\ProgramData\AzureConnectedMachineAgent\Tokens`	Lectura
`C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows`	Lectura y ejecución

Expandir tabla

Registro	Permisos
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\EventLog`	Lectura
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WinSock2\Parameters`	Acceso total
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Wbem\CIMOM`	Acceso total
`HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\SystemCertificates\Root`	Acceso total
`HKEY_LOCAL_MACHINE\Software\Microsoft\SystemCertificates`	Acceso total
`HKEY_LOCAL_MACHINE\Software\Microsoft\EnterpriseCertificates`	Acceso total
`HKEY_LOCAL_MACHINE\software\Microsoft\HybridRunbookWorker`	Acceso total
`HKEY_LOCAL_MACHINE\software\Microsoft\HybridRunbookWorkerV2`	Acceso total
`HKEY_CURRENT_USER\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed`	Acceso total
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Setup\PnpLockdownFiles`	Acceso total

Escenario: no se pudo iniciar el trabajo porque Hybrid Worker no estaba disponible cuando se inició el trabajo programado

Incidencia

El trabajo no se puede iniciar en una instancia de Hybrid Worker y verá el siguiente error:

No se pudo iniciar, ya que el trabajo híbrido no estaba disponible cuando se inició el trabajo programado, trabajo híbrido estaba activo por última vez en mm/dd/aaaa.

Causa

Este error puede producirse por alguno de los siguientes motivos:

- Las máquinas ya no existen.
- La máquina está desactivada y no se puede acceder a ella.
- La máquina tiene una incidencia de conectividad de red.
- La extensión de Hybrid Runbook Worker se desinstaló de la máquina.

Resolución

- Asegúrese de que la máquina exista y de que la extensión de Hybrid Runbook Worker esté instalada en ella. El estado de Hybrid Worker debe ser correcto y debería emitir un latido. Solucione los problemas de red revisando los registros de eventos Microsoft-SMA en los Workers del grupo Hybrid Runbook Worker que intentaron ejecutar este trabajo.
- También puede supervisar la métricaHybridWorkerPing que proporciona el número de pings de Hybrid Worker y puede ayudar a comprobar las incidencias relacionados con ping.

Escenario: se suspendió el trabajo, ya que superó el límite de trabajo de Hybrid Worker

Incidencia

El trabajo se suspende con el siguiente mensaje de error:

El trabajo se suspendió, ya que se superó el límite de trabajos de Hybrid Worker. Agregue más instancias de Hybrid Worker al grupo de Hybrid Worker para solucionar esta incidencia.

Causa

Los trabajos podrían suspenderse por alguna de las siguientes razones:

- Cada Hybrid Worker activo del grupo sondeará los trabajos cada 30 segundos para ver si hay algún trabajo disponible. El rol de trabajo elige los trabajos siguiendo la premisa de que el primero que llega se lo queda. En función de cuándo se insertó un trabajo, la instancia de Hybrid Worker que esté dentro del grupo de Hybrid Worker y que haga ping al servicio Automation tomará el trabajo en primer lugar. Normalmente, una sola instancia de Hybrid Worker puede seleccionar cuatro trabajos por ping (es decir, cada 30 segundos). Si la tasa de inserción de trabajos fuera superior a cuatro por cada 30 segundos y ningún otro Rol de trabajo recogiera el trabajo, es posible que el trabajo se suspenda.
- Hybrid Worker podría no sondear como se esperaba cada 30 segundos. Esto podría ocurrir si el Rol de trabajo no fuera correcto o hubiera incidencias de red.

Resolución

- Si el límite de trabajos de una instancia de Hybrid Worker superase los cuatro trabajos por cada 30 segundos, podría agregar más instancias de Hybrid Worker al grupo de Hybrid Worker para lograr alta disponibilidad y equilibrio de carga. También puede programar trabajos para que no superen el límite de cuatro trabajos por 30 segundos. El tiempo de procesamiento de la cola de trabajos depende de la carga y del perfil de hardware de

Hybrid Worker. Asegúrese de que el estado de Hybrid Worker sea correcto y de que emita un latido.

- Solucione los problemas de red revisando los registros de eventos Microsoft-SMA en los Workers del grupo Hybrid Runbook Worker que intentaron ejecutar este trabajo.
- También podría supervisar la métrica HybridWorkerPing, que proporciona el número de pings de una instancia de Hybrid Worker y podría ayudar a comprobar las incidencias relacionadas con ping.

Escenario: Se produce un error en la implementación de Hybrid Worker con Private Link error

Incidencia

Va a implementar una instancia de Hybrid Runbook Worker basada en extensiones en una máquina virtual y se produce un error: *Error de autenticación para vínculos privados*.

Causa

La red virtual de la máquina virtual es diferente del punto de conexión privado de la cuenta de Azure Automation, **o** no están conectados entre sí.

Resolución

Asegúrese de que el punto de conexión privado de Azure Automation cuenta esté conectado al mismo Virtual Network, al que está conectada la máquina virtual. Siga los pasos mencionados en Planeamiento basado en la red para conectarse a un punto de conexión privado. Establezca también marcas de acceso a la red pública para configurar una cuenta de Automation para denegar toda la configuración pública y permitir solo las conexiones a través de puntos de conexión privados. Para más información sobre cómo configurar las opciones de DNS para puntos de conexión privados, consulte Configuración de DNS

Escenario: se produce un error en la implementación de Hybrid Worker cuando el grupo de Hybrid Worker proporcionado no existe

Incidencia

Va a implementar una instancia de Hybrid Runbook Worker basada en extensiones en una máquina virtual y se produce un error: *La cuenta o el grupo especificados no existe*.

Causa

El grupo de Hybrid Runbook Worker en el que se va a implementar Hybrid Worker ya se ha eliminado.

Resolución

Asegúrese de crear el grupo de Hybrid Runbook Worker y agregar la máquina virtual como Hybrid Worker en ese grupo. Siga los pasos mencionados en crear un grupo de Hybrid Runbook Worker mediante el portal de Azure.

Escenario: se produce un error en la implementación de Hybrid Worker cuando la identidad administrada asignada por el sistema no está habilitada en la máquina virtual

Incidencia

Está implementando un Hybrid Runbook Worker basado en extensiones en una máquina virtual y falla con un error:
No se puede recuperar el punto de conexión de identidad de IMDS para máquinas virtuales que no son de Azure. Asegúrese de que el agente de máquina conectada Azure está instalado y que la identidad asignada por el sistema está habilitada.

Causa

Va a implementar el Hybrid Worker basado en la extensión en una máquina virtual fuera de Azure sin el agente de máquina Arc Connect instalado.

Resolución

Las máquinas que no son de Azure deben tener instalado el agente de máquina conectada a Arc antes de implementarlo como hybrid Runbook Worker basado en extensiones. Para instalar el `AzureConnectedMachineAgent`, consulte conectar máquinas híbridas a Azure desde el portal de Azure para servidores habilitados para Arc o Gestionar máquinas virtuales VMware con Azure Arc para habilitar la gestión de invitados para máquinas virtuales VMware habilitadas para Arc.

Escenario: se produce un error en la implementación de Hybrid Worker debido a que la identidad asignada por el sistema no está habilitada

Incidencia

Va a implementar un Hybrid Runbook Worker basado en una extensión en una máquina virtual, y se produce el error: *El token de autorización no es válido*.

Causa

La identidad administrada asignada por el usuario de la máquina virtual está habilitada, pero la identidad administrada asignada por el sistema no está habilitada.

Resolución

Siga los pasos que se enumeran a continuación:

1. Habilite una identidad administrada asignada por el sistema en la máquina virtual.
2. Elimine la extensión de Hybrid Worker instalada en la máquina virtual.
3. Vuelva a instalar la extensión de Hybrid Worker en la máquina virtual.

Escenario: el proceso de instalación de la extensión de Hybrid Worker en Windows máquina virtual se bloquea

Incidencia

Ha instalado la extensión de Hybrid Worker en una máquina virtual de Windows desde el portal, pero no recibe una notificación de que el proceso se ha completado correctamente.

Causa

A veces, es posible que el proceso de instalación se bloquee.

Resolución

Siga los pasos mencionados a continuación para volver a instalar la extensión de Hybrid Worker:

1. Abra la consola de PowerShell.

2. **Remove la clave del Registro**, si está presente:
   `HKLM:\Software\Microsoft\Azure\HybridWorker`

   a. Código de PowerShell para eliminar la clave del registro junto con cualquier subclave y valores bajo la misma:

```
PowerShell
Get-Item HKLM:\Software\Microsoft\Azure\HybridWorker | Remove-Item -Recurse
```

3. **Remove la clave del Registro**, si está presente:

   `HKLM:\Software\Microsoft\HybridRunbookWorkerV2`

   a. Código de PowerShell para eliminar la clave del registro junto con cualquier subclave y valores bajo la misma:

```
PowerShell
Get-Item HKLM:\Software\Microsoft\HybridRunbookWorkerV2 | Remove-Item -Recurse
```

4. Navegue hasta la carpeta de instalación de la extensión de Hybrid Worker:

> **Sugerencia**
>
> Reemplace `*` en el comando de abajo por la versión específica que está instalada si la conoce.

```
PowerShell
cd "C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows\*"
```

5. **Instale** la extensión de Hybrid Worker:

```
PowerShell
.\bin\install.ps1
```

6. **Habilite** la extensión de Hybrid Worker:

```
PowerShell
.\bin\enable.ps1
```

Escenario: el proceso de desinstalación de la extensión de Hybrid Worker en Windows máquina virtual se bloquea

Incidencia

Ha instalado una extensión de Hybrid Worker en una máquina virtual de Windows desde el portal, pero no recibe una notificación de que el proceso se ha completado correctamente.

Causa

A veces, es posible que el proceso de desinstalación se bloquee.

Resolución

1. Abra la consola de PowerShell.

2. Navegue hasta la carpeta de instalación de la extensión de Hybrid Worker:

   > **Sugerencia**
   >
   > Reemplace `*` en el comando de abajo por la versión específica que está instalada si la conoce.

   PowerShell
   ```
   cd "C:\Packages\Plugins\Microsoft.Azure.Automation.HybridWorker.HybridWorkerForWindows\*"
   ```

3. **Deshabilite** la extensión de Hybrid Worker:

   PowerShell
   ```
   .\bin\disable.cmd
   ```

4. **Desinstale** la extensión de Hybrid Worker:

   PowerShell
   ```
   .\bin\uninstall.ps1
   ```

5. **Eliminar** la clave del registro, si está presente:
   `HKLM:\Software\Microsoft\Azure\HybridWorker`

a. Código de PowerShell para quitar la clave del Registro junto con las subclaves y los valores que contiene.

```
PowerShell
Get-Item HKLM:\Software\Microsoft\Azure\HybridWorker | Remove-Item -Recurse
```

6. **Remove la clave del Registro**, si está presente:
`HKLM:\Software\Microsoft\HybridRunbookWorkerV2`

a. Código de PowerShell para quitar la clave del Registro junto con las subclaves y los valores que contiene.

```
PowerShell
Get-Item HKLM:\Software\Microsoft\HybridRunbookWorkerV2 | Remove-Item -Recurse
```

Escenario: el proceso de instalación de la extensión de Hybrid Worker en una máquina virtual Linux se bloquea

Incidencia

Ha instalado una extensión de Hybrid Worker en una máquina virtual Linux desde el portal, pero no recibe una notificación de que el proceso se ha completado correctamente.

Causa

A veces, es posible que el proceso de desinstalación se bloquee.

Resolución

1. **Elimine** la carpeta `state`:

```
Bash
rm -r /home/hweautomation/state
```

2. Navegue hasta la carpeta de instalación de la extensión de Hybrid Worker:

> **Sugerencia**

Reemplace `*` en el comando de abajo por la versión específica que está instalada si la conoce.

```bash
cd /var/lib/waagent/Microsoft.Azure.Automation.HybridWorker.HybridWorkerForLinux-*/
```

3. **Elimine** el archivo mrseq:

```bash
rm mrseq
```

4. **Instale** la extensión de Hybrid Worker:

```bash
./extension_shim.sh -c ./HWExtensionHandlers.py -i
```

5. **Habilite** la extensión de Hybrid Worker:

```bash
./extension_shim.sh -c ./HWExtensionHandlers.py -e
```

Escenario: el proceso de desinstalación de la extensión de Hybrid Worker en una máquina virtual Linux se bloquea

Incidencia

Ha desinstalado la extensión de Hybrid Worker en una máquina virtual Linux desde el portal, pero no recibe una notificación de que el proceso se ha completado correctamente.

Causa

A veces, es posible que el proceso de desinstalación se bloquee.

Resolución

Siga los pasos mencionados a continuación para desinstalar completamente la extensión de Hybrid Worker:

1. Navegue hasta la carpeta de instalación de la extensión de Hybrid Worker:

> **Sugerencia**
>
> Reemplace `*` en el comando de abajo por la versión específica que está instalada si la conoce.

```bash
cd /var/lib/waagent/Microsoft.Azure.Automation.HybridWorker.HybridWorkerForLinux-*/
```

2. **Deshabilite** la extensión de Hybrid Worker:

```bash
./extension_shim.sh -c ./HWExtensionHandlers.py -d
```

3. **Desinstale** la extensión de Hybrid Worker:

```bash
./extension_shim.sh -c ./HWExtensionHandlers.py -u
```

Escenario: Error en la ejecución de un runbook

Incidencia

Se produce un error en la ejecución del runbook y recibe el mensaje de error siguiente:

`The job action 'Activate' cannot be run, because the process stopped unexpectedly. The job action was attempted three times.`

El runbook se suspende poco después de intentar ejecutarse tres veces. Hay condiciones que pueden interrumpir la finalización del runbook. El mensaje de error relacionado podría no incluir ninguna información adicional.

Causa

Las posibles causas son las siguientes:

- Los runbooks no se pueden autenticar con recursos locales.

- Hybrid Worker está detrás de un firewall o proxy.
- El equipo configurado para ejecutar Hybrid Runbook Worker no cumple los requisitos mínimos de hardware.

Resolución

Compruebe que el equipo tenga acceso de salida a ***.azure-automation.net** en el puerto 443.

Los equipos que ejecutan Hybrid Runbook Worker deben cumplir los requisitos mínimos de hardware antes de que el trabajo esté configurado para hospedar esta característica. Los runbooks y el proceso en segundo plano que usan pueden provocar un uso excesivo del sistema, así como tiempos de espera y retrasos del trabajo de los runbooks.

Confirme que el equipo que ejecutará la característica Hybrid Runbook Worker cumple los requisitos mínimos de hardware. Si lo hace, supervise el uso de CPU y memoria para determinar cualquier correlación entre el rendimiento de los procesos de Hybrid Runbook Worker y los Windows. Cualquier presión de CPU o memoria puede indicar la necesidad de actualizar los recursos. También puede seleccionar otro recurso de proceso que pueda admitir los requisitos mínimos y escalarse cuando las demandas de carga de trabajo indiquen que es necesario un aumento.

Compruebe el registro de eventos **Microsoft-SMA** para obtener un evento correspondiente con la descripción `Win32 Process Exited with code [4294967295]`. La causa de este error es que no ha configurado la autenticación en sus runbooks o no ha especificado las credenciales de ejecución para el grupo de Hybrid Runbook Worker. Para confirmar que ha configurado correctamente la autenticación para sus runbooks, revise los permisos de runbook en Ejecución de runbooks en Hybrid Runbook Worker.

Escenario: no se encontró ningún certificado en el almacén de certificados de la instancia de Hybrid Runbook Worker

Incidencia

Se produce un error en un runbook que se ejecutan en Hybrid Runbook Worker con el mensaje de error siguiente:

```
Connect-AzAccount : No certificate was found in the certificate store with thumbprint
0000000000000000000000000000000000000000
At line:3 char:1
```

```
+ Connect-AzAccount -ServicePrincipal -Tenant $Conn.TenantID -Appl ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : CloseError: (:) [Connect-AzAccount],ArgumentException
    + FullyQualifiedErrorId : Microsoft.Azure.Commands.Profile.ConnectAzAccountCommand
```

Causa

Este error se produce cuando se intenta usar una cuenta de ejecución en un runbook que se realiza en una instancia de Hybrid Runbook Worker en la que no existe el certificado de la cuenta de ejecución. Las instancias de Hybrid Runbook Worker no tienen el recurso de certificado localmente de forma predeterminada. La cuenta de ejecución requiere que este recurso funcione correctamente.

Resolución

Si Hybrid Runbook Worker es una máquina virtual de Azure, puede usar la autenticación runbook con identidades administradas en su lugar. Este escenario simplifica la autenticación al permitirle autenticarse en recursos de Azure mediante la identidad administrada de la máquina virtual de Azure en lugar de la cuenta Run As. Cuando la instancia de Hybrid Runbook Worker es una máquina local, deberá instalar el certificado de cuenta de ejecución en la máquina. Para obtener información sobre cómo instalar el certificado, consulte los pasos para ejecutar el runbook de PowerShell **Export-RunAsCertificateToHybridWorker** en Ejecución de runbooks en Hybrid Runbook Worker.

Escenario: Microsoft Azure Máquinas Virtuales eliminadas automáticamente de un grupo de Trabajadores Híbridos.

Incidencia

No puede ver Hybrid Runbook Worker ni las máquinas virtuales cuando el equipo de trabajo ha estado apagado durante mucho tiempo.

Causa

La máquina Hybrid Runbook Worker no se ha comunicado con Azure Automation durante más de 30 días. Como resultado, Automation ha purgado el grupo de Hybrid Runbook Worker o el grupo de trabajo del sistema.

Resolución

Inicie la máquina de trabajo y vuelva a registrarla con Azure Automation. Para obtener instrucciones sobre cómo instalar el entorno de runbook y conectarse a Azure Automation, consulte Deploy a Windows Hybrid Runbook Worker.

Escenario: La ejecución de trabajos de Hybrid Runbook Worker en un servidor Windows habilitado con Azure Arc que utiliza una credencial personalizada se suspende inesperadamente.

Incidencia

Los trabajos de runbook ejecutados desde un servidor habilitado para Azure Arc que usan una credencial personalizada comienzan repentinamente a entrar en un estado suspendido.

Causa

Esto se debe a un problema conocido por el que se quitan los permisos de carpeta cuando se actualiza el agente de Azure Connected Machine. Los permisos de carpeta de `C:\ProgramData\AzureConnectedMachineAgent\Tokens` se quitan cuando se actualiza el agente de Azure Connected Machine.

Resolución

La resolución actual consiste en volver a aplicar los permisos de carpeta para `C:\ProgramData\AzureConnectedMachineAgent\Tokens` cuando se actualiza el agente de Azure Connected Machine. Consulte Permisos para las credenciales de Hybrid Worker.

Pasos siguientes

Si su problema no aparece aquí o no puede resolverlo, intente obtener ayuda adicional mediante uno de los siguientes canales:

- Recibe respuestas de expertos de Azure a través de Azure Forums.
- Conéctese con @AzureSupport, la cuenta oficial de Microsoft Azure para mejorar la experiencia del cliente. Azure Soporte técnico conecta la comunidad de Azure a respuestas, soporte técnico y expertos.
- Si tiene previsto abrir un caso de soporte técnico para un problema sin resolver, recopile los datos de diagnóstico necesarios antes de abrirlo. Consulte Datos para recopilar al abrir un caso para Microsoft Azure Automation.
- Registrar un incidente de soporte de Azure. Vaya al sitio Soporte técnico de Azure y seleccione **Get Support**.

Last updated on 15/04/2026

Solución de problemas de Azure Automation State Configuration

> **Nota:**
>
> Azure Automation State Configuration se retirará el 30 de septiembre de 2027, realice la transición a Azure Machine Configuration por esa fecha. Para obtener más información, consulte el anuncio de la entrada de blog. El servicio Azure Machine Configuration combina características de la extensión de DSC, Azure Automation State Configuration y las características más solicitadas de los comentarios de los clientes. Azure Machine Configuration también incluye compatibilidad con máquinas híbridas a través de servidores habilitados para Arc.

> **Importante**
>
> Los vínculos de navegación **Agregar**, **Configuración de Compose**, y **Galería** se quitarán del portal el 31 de marzo de 2025.

> **Precaución**
>
> Azure Automation DSC para Linux se ha retirado el 30 de septiembre de 2023. Para obtener más información, consulte el anuncio.

En este artículo se proporciona información sobre la solución de problemas y la resolución de problemas que se producen al compilar o implementar configuraciones en Azure Automation State Configuration. Para obtener información general sobre la función de Configuración de Estado, consulte introducción de Azure Automation State Configuration.

Diagnóstico de problemas

Si recibe un error de compilación o implementación de la configuración, los pasos siguientes le ayudarán a diagnosticar el problema.

1. Asegúrese de que la configuración se compila correctamente en la máquina local

Azure Automation State Configuration se basa en powerShell Desired State Configuration (DSC). Puede encontrar la documentación del lenguaje y la sintaxis de DSC en los documentos de DSC de PowerShell.

Al compilar una configuración de DSC en la máquina local, puede detectar y resolver errores comunes, como los siguientes:

- Módulos que faltan
- Errores de sintaxis
- Errores lógicos

2. Visualización de los registros de DSC en el nodo

Si la configuración se compila correctamente, pero se produce un error cuando se aplica a un nodo, puede encontrar información detallada en los registros de DSC. Para obtener información sobre dónde encontrar estos registros, consulte ¿Dónde se encuentran los registros de eventos de DSC?

El módulo xDscDiagnostics puede ayudarle a analizar la información detallada de los registros de DSC. Si se pone en contacto con el equipo de soporte técnico, necesitarán estos registros para diagnosticar el problema.

Puede instalar el módulo `xDscDiagnostics` en la máquina local con las instrucciones para la instalación de la versión estable del módulo.

Para instalar el módulo en su máquina de Azure, use Invoke-AzVMRunCommand. También puede usar la opción **Ejecutar** en el portal de Azure siguiendo los pasos descritos en Ejecutar scripts de PowerShell en su máquina virtual de Windows con Ejecutar Comando.

Para obtener información sobre el uso de **xDscDiagnostics**, consulte Uso de xDscDiagnostics para analizar los registros de DSC. Consulte también Cmdlets de xDscDiagnostics.

3. Asegúrese de que los nodos y el área de trabajo de Automation tengan los módulos necesarios

DSC depende de módulos que hay instalados en el nodo. Al usar Azure Automation State Configuration, importe los módulos necesarios en la cuenta de Automation siguiendo los pasos

descritos en Import Modules. Las configuraciones también pueden tener una dependencia en versiones específicas de los módulos. Para más información, consulte Solución de problemas de módulos.

Escenario: no se puede eliminar una configuración con caracteres especiales del portal

Problema

Al intentar eliminar una configuración de DSC del portal, verá el siguiente error:

error

```
An error occurred while deleting the DSC configuration '<name>'. Error-details: The
argument
configurationName with the value <name> is not valid. Valid configuration names can
contain only
letters, numbers, and underscores. The name must start with a letter. The length of
the name must be
between 1 and 64 characters.
```

Causa

Este error es un problema temporal. Vuelva a intentarlo más tarde.

Solución

Use el cmdlet Remove-AzAutomationDscConfiguration para eliminar la configuración.

Escenario: no se pudo registrar el agente de DSC

Problema

Recibirá un error al usar Set-DscLocalConfigurationManager u otro cmdlet de DSC.

error

```
Registration of the Dsc Agent with the server
https://<location>-agentservice-prod-1.azure-automation.net/accounts/00000000-0000-
0000-0000-000000000000
failed. The underlying error is: Failed to register Dsc Agent with AgentId
00000000-0000-0000-0000-000000000000 with the server
https://<location>-agentservice-prod-1.azure-automation.net/accounts/00000000-0000-
0000-0000-000000000000/Nodes(AgentId='00000000-0000-0000-0000-000000000000').
```

```
    + CategoryInfo          : InvalidResult:
(root/Microsoft/...gurationManager:String) [], CimException
    + FullyQualifiedErrorId :
RegisterDscAgentCommandFailed,Microsoft.PowerShell.DesiredStateConfiguration.Commands.
RegisterDscAgentCommand
    + PSComputerName        : <computerName>
```

Causa

Este error puede provocarlo un problema de red. Compruebe la configuración del firewall o si la máquina está detrás de un servidor proxy.

Solución

Compruebe que la máquina tenga acceso a los puntos de conexión adecuados para DSC y vuelva a intentarlo. Para ver una lista de puertos y direcciones necesarios, consulte el artículo sobre planeamiento de red.

Escenario: los informes de estado devuelven el código de respuesta No autorizado

Problema

Al registrar un nodo con Azure Automation State Configuration, recibirá uno de los siguientes mensajes de error:

error

```
The attempt to send status report to the server https://{your Automation account
URL}/accounts/xxxxxxxxxxxxxxxxxxxxxx/Nodes(AgentId='xxxxxxxxxxxxxxxxxxxxxxxxx')/SendRe
port returned
unexpected response code Unauthorized.
```

error

```
VM has reported a failure when processing extension 'Microsoft.Powershell.DSC /
Registration of the
Dsc Agent with the server failed.
```

Causa

Las posibles causas son las siguientes:

- Un certificado incorrecto o expirado. Consulte Nuevo registro de un nodo.
- Configuración de proxy que no permite el acceso a `*.azure-automation.net`. Para obtener más información, vea Configuración de redes privadas.
- Al deshabilitar la autenticación local en Azure Automation. Consulte Deshabilitación de la autenticación local. Para corregirlo, consulte Volver a habilitar la autenticación local.
- El tiempo del equipo cliente difiere en varios minutos del tiempo real. Use el comando siguiente para comprobar la hora: `w32tm /stripchart /computer:time.windows.com /samples:6`.

Solución

Siga estos pasos para volver a registrar el nodo de DSC con errores.

Paso 1: anulación del registro del nodo

1. En el portal de Azure, vaya a **Inicio**>**Cuentas de Automatización**> (su cuenta de Automatización) >**Configuración de Estado (DSC)**.
2. Seleccione **Nodos** y, luego, el nodo con problemas.
3. Seleccione **Cancelar registro** para cancelar el registro del nodo.

Paso 2: desinstalación de la extensión DSC del nodo

1. En el portal de Azure, vaya a **Home**>**Virtual Machine**> (nodo con errores) >**Extensions**.
2. Seleccione **Microsoft. Powershell.DSC**, la extensión DSC de PowerShell.
3. Haga clic en **Desinstalar** para desinstalar la extensión.

Paso 3: eliminación de todos los certificados incorrectos o expirados del nodo

En un nodo defectuoso, desde un símbolo del sistema de PowerShell con permisos elevados, ejecute estos comandos:

PowerShell

```
$certs = @()
$certs += dir cert:\localmachine\my | ?{$_.FriendlyName -like "DSC"}
$certs += dir cert:\localmachine\my | ?{$_.FriendlyName -like "DSC-OaaS Client
Authentication"}
$certs += dir cert:\localmachine\CA | ?{$_.subject -like "CN=AzureDSCExtension*"}
"";"== DSC Certificates found: " + $certs.Count
$certs | FL ThumbPrint,FriendlyName,Subject
```

```
If (($certs.Count) -gt 0)
{
    ForEach ($Cert in $certs)
    {
        RD -LiteralPath ($Cert.Pspath)
    }
}
```

Paso 4: repetición del registro del nodo con errores

1. En el portal de Azure, vaya a **Inicio**>**Cuentas de Automatización**> (su cuenta de Automatización) >**Configuración de Estado (DSC)**.
2. Seleccione **Nodos**.
3. Seleccione **Agregar**.
4. Seleccione el nodo con errores.
5. Seleccione **Conectar** y, luego, las opciones deseadas.

Escenario: el nodo se encuentra en estado de error con el error "No encontrado"

Problema

El nodo tiene un informe con el estado de error y contiene el error:

error

```
The attempt to get the action from server
https://<url>//accounts/<account-id>/Nodes(AgentId=<agent-id>)/GetDscAction failed
because a valid
configuration <guid> cannot be found.
```

Causa

Este error suele ocurrir cuando se asigna al nodo un nombre de configuración, como **ABC**, en lugar de un nombre de configuración de nodo (archivo MOF), como **ABC.WebServer**.

Solución

- Asegúrese de que está asignando al nodo el nombre de configuración del nodo y no el nombre de configuración.
- Puede asignar una configuración de nodo a un nodo mediante el portal de Azure o con un cmdlet de PowerShell.

- En el portal de Azure, vaya a **Inicio**>**Cuentas de Automatización**> (su cuenta de Automatización) >**Configuración de Estado (DSC)**. Seleccione un nodo y, luego, **Asignar configuración de nodo**.
- Utiliza el cmdlet Set-AzAutomationDscNode.

Escenario: no se produjeron configuraciones de nodo (archivos MOF) al compilarse una configuración

Problema

Su trabajo de compilación de DSC se suspende con el error:

```
error
Compilation completed successfully, but no node configuration **.mof** files were generated.
```

Causa

Cuando la expresión que aparece junto a la palabra clave `Node` en la configuración de DSC se evalúa en `$null`, no se produce ninguna configuración de nodo.

Solución

Use cualquiera de las siguientes soluciones para corregir el problema.

- Asegúrese de que la expresión junto a la palabra clave `Node` en la definición de configuración no evalúe a nulo.
- Si pasa ConfigurationData al compilar la configuración, asegúrese de que pasa los valores que la configuración espera de los datos de configuración.

Escenario: el informe de nodo de DSC se queda bloqueado en el estado En curso

Problema

El agente DSC genera la salida:

```
error
```

```
No instance found with given property values
```

Causa

Este problema puede producirse si Instrumental de administración de Windows (WMI) está dañado en el nodo.

Solución

Siga las instrucciones de Problemas y limitaciones conocidos en DSC.

Escenario: no se puede usar una credencial en una configuración de DSC

Problema

El trabajo de compilación de DSC ha sido suspendido debido al error:

error

```
System.InvalidOperationException error processing property 'Credential' of type <some resource
name>: Converting and storing an encrypted password as plaintext is allowed only if PSDscAllowPlainTextPassword is set to true.
```

Causa

Este problema puede producirse cuando se usa una credencial en una configuración, pero no se proporciona el valor adecuado de `ConfigurationData` para establecer `PSDscAllowPlainTextPassword` en true para cada configuración de nodo.

Solución

Asegúrese de pasar el `ConfigurationData` adecuado, estableciendo `PSDscAllowPlainTextPassword` en true para cada configuración de nodo. Consulte Compilar configuraciones de DSC en Azure Automation State Configuration.

Escenario: error de procesamiento de la extensión al habilitar una máquina desde una extensión DSC

Problema

Al habilitar una máquina mediante una extensión DSC, se produce un error que contiene lo siguiente:

error

```
VM has reported a failure when processing extension 'Microsoft.Powershell.DSC'. Error
message: \"DSC
COnfiguration 'RegistrationMetaConfigV2' completed with error(s). Following are the
first few:
Registration of the Dsc Agent with the server <url> failed. The underlying error is:
The attempt to
register Dsc Agent with Agent Id <ID> with the server <url> return unexpected response
code
BadRequest. .\".
```

Causa

Este error suele producirse cuando se asigna al nodo un nombre de configuración que no existe en el servicio.

Solución

Asegúrese de que el nombre del nodo coincida exactamente con el nombre en el servicio, o no incluya el nombre de configuración del nodo. Esto habilita el nodo, pero no asigna una configuración de nodo.

Escenario: error "Se produjeron uno o más errores" al registrar un nodo con PowerShell

Problema

Al registrar un nodo mediante Register-AzAutomationDSCNode, se produce el siguiente error:

error

```
One or more errors occurred.
```

Causa

Este error se produce cuando se intenta registrar un nodo en una suscripción diferente de la que usa la cuenta de Automation.

Solución

Trate el nodo entre suscripciones como un nodo definido para una nube independiente o local. Registre el nodo mediante una de estas opciones para habilitar las máquinas:

- Windows: Máquinas de Windows físicas o virtuales locales o en una nube distinta de Azure/AWS.
- Linux: Máquinas Linux físicas o virtuales locales o en una nube distinta de Azure.

Escenario: mensaje de error: "Error en el aprovisionamiento"

Problema

Al registrar un nodo, se muestra el error:

error

```
Provisioning has failed
```

Causa

Este mensaje se produce cuando hay un problema con la conectividad entre el nodo y Azure.

Solución

Determine si el nodo está en una red privada virtual (VPN) o tiene otros problemas al conectarse a Azure. Vea Solución de incidencias en la implementación de características.

Escenario: error general al aplicar una configuración en Linux

Problema

Al aplicar una configuración en Linux, se produce un error que contiene lo siguiente:

error

```
This event indicates that failure happens when LCM is processing the configuration.
ErrorId is 1.
ErrorDetail is The SendConfigurationApply function did not succeed.. ResourceId is
[resource]name
and SourceInfo is ::nnn::n::resource. ErrorMessage is A general error occurred, not
covered by a
more specific error code..
```

Causa

Si la ubicación **/tmp** se establece en `noexec`, la versión actual de DSC no puede aplicar las configuraciones.

Solución

Quite la opción `noexec` de la ubicación **/tmp**.

Escenario: los nombres de configuración de nodos que se superponen pueden provocar una versión incorrecta

Problema

Si usa un único script de configuración para generar varias configuraciones de nodo y algunos nombres de configuración de nodos son subconjuntos de otros nombres, el servicio de compilación puede acabar asignando la configuración incorrecta. Este problema solo se produce al utilizar un único script para generar configuraciones con datos de configuración por nodo y solo cuando la superposición de los nombres tiene lugar al principio de la cadena. Un ejemplo es un script de configuración único para generar configuraciones basadas en los datos de nodo pasados como una tabla hash mediante cmdlets, y los datos del nodo incluyen servidores denominados **server** y **1server**.

Causa

Este es un problema conocido con el servicio de compilación.

Solución

La mejor solución sería realizar la compilación localmente o en una canalización de CI/CD y cargar los archivos MOF de configuración de nodos directamente en el servicio. Si es necesario

que la compilación tenga lugar en el servicio, la siguiente mejor solución alternativa es dividir los trabajos de compilación para que no haya nombres superpuestos.

Escenario: Error de tiempo de espera del gateway en la carga de la configuración de DSC

Problema

Recibe un error `GatewayTimeout` al cargar una configuración de DSC.

Causa

Las configuraciones de DSC que tardan mucho tiempo en compilarse pueden producir este error.

Solución

Las configuraciones de DSC se pueden analizar más rápidamente si se incluye expresamente el parámetro `ModuleName` en cualquier llamada Import-DSCResource.

Escenario: Error al incorporar una máquina

Problema

Se produce un error `agent has a problem` al configurar una máquina.

Causa

Este es un problema conocido. No puede volver a asignar la misma configuración porque el nodo permanece en estado pendiente.

Solución

La solución alternativa al problema consiste en aplicar otra configuración de prueba y volver a probar la configuración original.

Pasos siguientes

Si su problema no aparece aquí o no puede resolverlo, intente uno de los siguientes canales de ayuda:

- Recibe respuestas de expertos de Azure a través de Azure Forums.

- Conéctese con @AzureSupport, la cuenta oficial de Microsoft Azure para mejorar la experiencia del cliente. Azure Soporte técnico conecta la comunidad de Azure a respuestas, soporte técnico y expertos.
- Si tiene previsto abrir un caso de soporte técnico para un problema sin resolver, recopile los datos de diagnóstico necesarios antes de abrirlo. Consulte Datos para recopilar al abrir un caso para Microsoft Azure Automation.
- Registrar un incidente de soporte de Azure. Vaya al sitio Soporte técnico de Azure y seleccione **Get Support**.

Last updated on 20/04/2026

Datos que se recopilan cuando se abre un caso para Microsoft Azure Automation

Artículo • 01/06/2023

En este artículo se describe la información que debe recopilar antes de abrir un caso para Azure Automation con el soporte técnico de Microsoft Azure. No obstante, esta información no es necesaria para abrir un caso. Sin embargo, ayuda al equipo de soporte técnico a resolver rápidamente un problema.

> **Nota**
>
> Para más información, consulte el artículo de Knowledge Base **4034605: Captura de diagnósticos con scripts de Azure Automation**.

Datos de problemas de Update Management en Linux

1. Ejecute la siguiente herramienta de recopilación de registros, además de los detalles de KB 4034605.
   - Recopilador de registros del Agente de Linux de OMS
2. Comprima el contenido de la carpeta **/var/opt/microsoft/omsagent/run/automationworker/** y, a continuación, envíe el archivo comprimido al soporte técnico de Azure.
3. Compruebe que el identificador del área de trabajo de la que depende el agente de Log Analytics para Linux es el mismo que el del área de trabajo que se está supervisando para ver si hay actualizaciones.

Datos de problemas de Update Management en Windows

1. Recopile datos de los elementos enumerados en 4034605.
2. Exporte los siguientes registros de eventos en formato EVTX:

- Sistema
- Application
- Seguridad
- Operations Manager
- Microsoft-SMA/Operational

3. Compruebe que el identificador del área de trabajo de la que depende el agente es el mismo que el del área de trabajo que Windows Update está supervisando.

Datos de problemas de trabajo

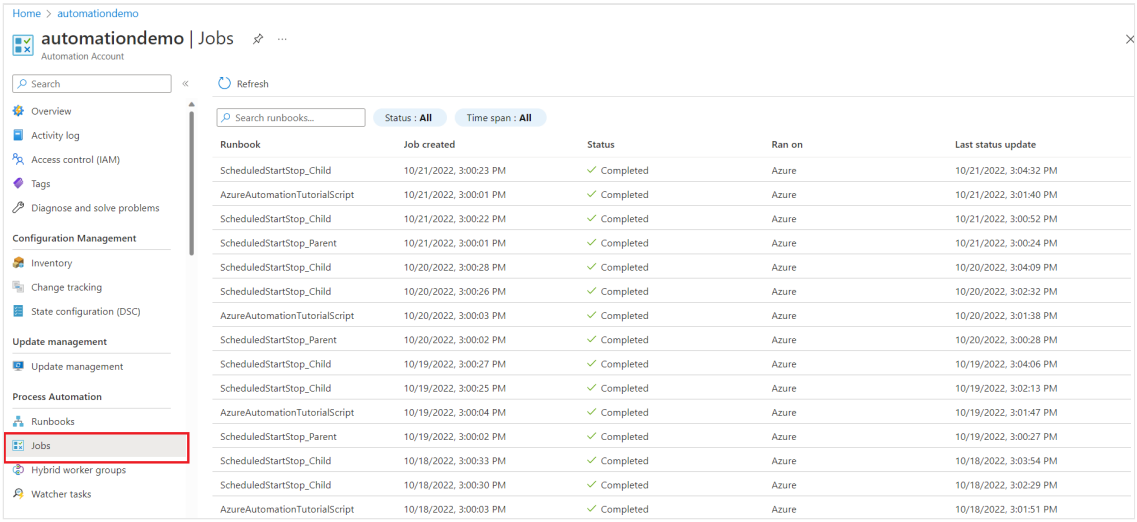
1. Recopile datos de los elementos enumerados en 4034605 ↗.

2. Recopile el identificador del trabajo que tiene un problema:

   a. En Azure Portal, vaya a **Cuentas de Automation**.

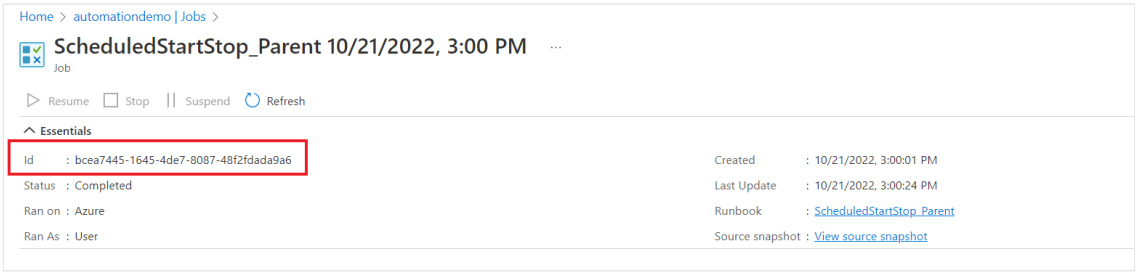
   b. Seleccione la cuenta de Automation de la que está solucionando el problema y anote el nombre.

   c. Seleccione **Trabajos**.



   d. Seleccione el trabajo del que está solucionando el problema.

   e. En el panel Resumen del trabajo, busque el valor GUID en **Id. de trabajo**.



3. Recopile un ejemplo del script que está ejecutando.

4. Recopile los archivos de registro:

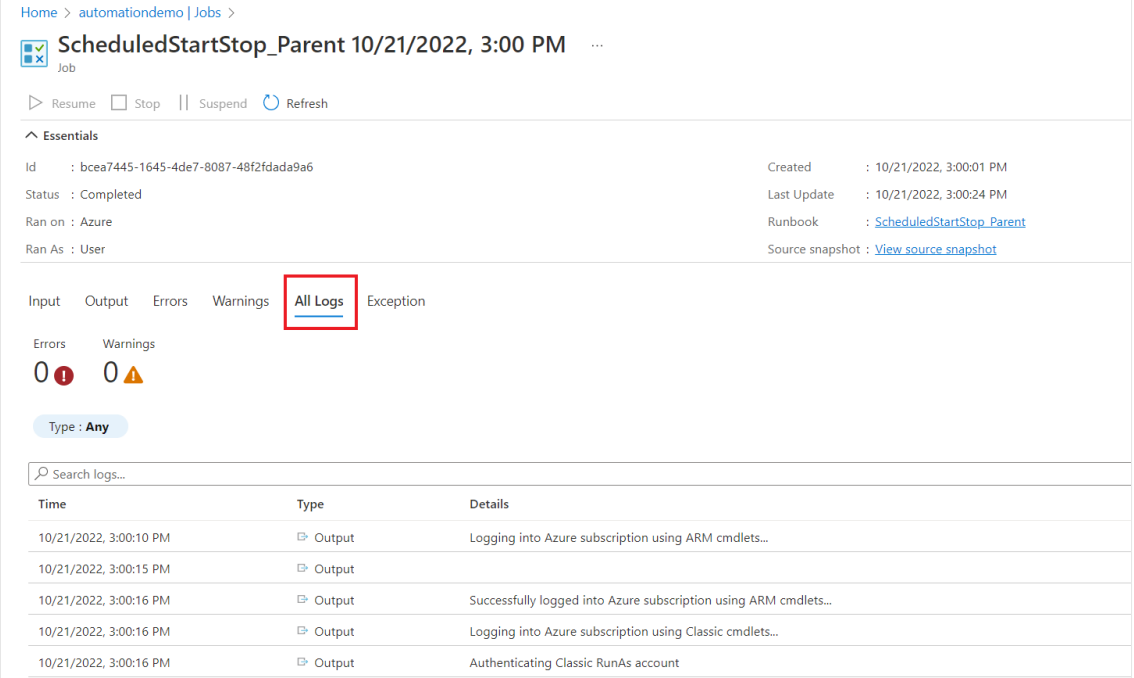
   a. En Azure Portal, vaya a **Cuentas de Automation**.

   b. Seleccione la cuenta de Automation de la que está solucionando el problema.

   c. Seleccione **Trabajos**.

   d. Seleccione el trabajo del que está solucionando el problema.

   e. Seleccione **Todos los registros**. En el panel siguiente, puede recopilar los datos.



Datos de problemas de módulos

Además del artículo de Knowledge Base 4034605: Captura de diagnósticos con scripts de Azure Automation, obtenga la siguiente información:

- Los pasos que siguió para poder reproducir el problema.
- Capturas de pantalla de los mensajes de error.
- Capturas de pantalla de los módulos actuales y sus números de versión.

Pasos siguientes

- Obtenga respuestas de expertos de Azure en los foros de Azure.
- Póngase en contacto con @AzureSupport, la cuenta de Microsoft Azure oficial para mejorar la experiencia del cliente al conectar la comunidad de Azure a los

recursos adecuados: respuestas, soporte técnico y expertos.

- Registrar un incidente de soporte técnico de Azure. Vaya al sitio de Soporte técnico de Azure y seleccione **Obtener soporte**.

Política de soporte y retirada de las pilas de lenguajes de Azure Automation

En este artículo se explica la directiva de soporte y retirada de las pilas de lenguajes de Azure Automation, como PowerShell y Python. Le ayuda a comprender cómo se determina el fin del soporte técnico, el proceso de retirada en tiempo de ejecución, sus fases y cómo afectan estos cambios a los runbooks existentes.

Información general

Los libros de trabajo de tiempo de ejecución de Azure Automation incluyen el componente de host y los trabajadores específicos del lenguaje de programación. La compatibilidad del tiempo de ejecución de los lenguajes para los libros de ejecución se ajusta a la fecha de fin de soporte de un lenguaje determinado. Para mantener los runbooks actualizados y con compatibilidad, Azure Automation aplica una reducción gradual del soporte a medida que las versiones del entorno de lenguaje alcanzan sus fechas de fin de soporte. El soporte técnico finaliza en la fecha que se produce en primer lugar a partir de lo siguiente:

- Fecha de finalización del soporte técnico de la comunidad para el idioma
- Fecha de finalización del soporte técnico para el sistema operativo base subyacente

Proceso de retirada del entorno de ejecución

Azure Automation usa un modelo de retirada en dos fases que proporciona un aviso anticipado antes de que las versiones en tiempo de ejecución retiradas se muevan a una compatibilidad reducida.

Fase de notificación

Azure Automation envía correos electrónicos de notificación sobre las versiones de lenguaje que están a punto de retirarse y que afectan a sus libros de ejecución. Estas notificaciones proporcionan un aviso anticipado de los próximos cambios, incluido el calendario de retirada y su posible impacto en los libros de procedimientos existentes. Este período de aviso le ayuda a prepararse para usar las versiones en tiempo de ejecución admitidas antes del final del soporte técnico.

Fase de retirada

> **Importante**
>
> Si usa Azure Automation runbooks con una versión de lenguaje o tiempo de ejecución no compatible, es posible que encuentre problemas y impactos en el rendimiento. El uso de versiones en tiempo de ejecución compatibles ayuda a garantizar la confiabilidad, el rendimiento y la compatibilidad continua.

Aún puede crear, implementar y ejecutar libros de ejecución y entornos que utilicen versiones de lenguaje retiradas en la plataforma. Sin embargo, funcionarán con un soporte reducido. Estos runbooks no son aptos para nuevas características, actualizaciones de seguridad o optimizaciones de rendimiento hasta que se actualicen a una versión de lenguaje compatible. En algunos casos, la plataforma también podría limitar el número de instancias asignadas a estos runbooks, incluida la restricción del escalado a una sola instancia.

Contenido relacionado

- Obtenga información sobre el entorno en tiempo de ejecución en Azure Automation.
- Obtenga información sobre cómo administrar el entorno en tiempo de ejecución y los runbooks asociados.
- Obtenga información sobre cómo administrar runbooks en Azure Automation.

Last updated on 10/07/2026

az automation

> **Nota:**
>
> Esta referencia forma parte de la extensión de **automatización** para la CLI de Azure (versión 2.75.0 o posterior). La extensión instalará automáticamente la primera vez que ejecute un comando **az automation** . Obtenga más información sobre las extensiones.

Administrar la cuenta de Automation.

Comandos

Expandir tabla

Nombre	Description	Tipo	Estado
az automation account	Cuenta de Automation.	Extension	Experimental
az automation account create	Cree una cuenta de Automation.	Extension	Experimental
az automation account delete	Elimine una cuenta de Automation.	Extension	Experimental
az automation account list	Recupere una lista de cuentas dentro de un grupo de recursos determinado. Y recuperar una lista de cuentas dentro de una suscripción determinada.	Extension	Experimental
az automation account list-keys	Recupere las claves de automatización de una cuenta.	Extension	Experimental
az automation account show	Obtenga información sobre una cuenta de Automation.	Extension	Experimental
az automation account show-linked-workspace	Obtenga el área de trabajo vinculada para el identificador de cuenta.	Extension	Experimental
az automation account update	Actualice una cuenta de Automation.	Extension	Experimental
az automation configuration	Administrar la configuración de Dsc.	Extension	GA

Nombre	Description	Tipo	Estado
az automation configuration create	Cree la configuración identificada por nombre de configuración.	Extension	GA
az automation configuration delete	Elimine la configuración de dsc identificada por nombre de configuración.	Extension	GA
az automation configuration list	Enumerar una lista de configuraciones.	Extension	GA
az automation configuration show	Obtenga la configuración identificada por nombre de configuración.	Extension	GA
az automation configuration show-content	Obtenga el script de configuración identificado por nombre de configuración.	Extension	GA
az automation configuration update	Actualice la configuración identificada por nombre de configuración.	Extension	GA
az automation hrwg	Grupo de Hybrid Runbook Worker de Automation.	Extension	GA
az automation hrwg create	Cree un grupo de Hybrid Runbook Worker.	Extension	GA
az automation hrwg delete	Elimine un grupo de Hybrid Runbook Worker.	Extension	GA
az automation hrwg hrw	Administrar Hybrid Runbook Worker de Automation.	Extension	GA
az automation hrwg hrw create	Cree una instancia de Hybrid Runbook Worker.	Extension	GA
az automation hrwg hrw delete	Elimine una instancia de Hybrid Runbook Worker.	Extension	GA
az automation hrwg hrw list	Enumerar una lista de hybrid runbook workers.	Extension	GA
az automation hrwg hrw move	Mueva una instancia de Hybrid Worker a otro grupo.	Extension	GA
az automation hrwg hrw show	Obtenga una instancia de Hybrid Runbook Worker.	Extension	GA
az automation hrwg list	Enumerar todos los grupos de Hybrid Runbook Worker.	Extension	GA
az automation hrwg show	Obtenga el grupo de Hybrid Worker.	Extension	GA

Nombre	Description	Tipo	Estado
az automation hrwg update	Actualice un grupo de Hybrid Runbook Worker.	Extension	GA
az automation job	Trabajo de automatización.	Extension	Experimental
az automation job list	Recuperar una lista de trabajos.	Extension	Experimental
az automation job resume	Reanude el trabajo identificado por jobName.	Extension	Experimental
az automation job show	Recupere el trabajo identificado por nombre de trabajo.	Extension	Experimental
az automation job stop	Detenga el trabajo identificado por jobName.	Extension	Experimental
az automation job suspend	Suspenda el trabajo identificado por nombre de trabajo.	Extension	Experimental
az automation python3-package	Python3-package.	Extension	GA
az automation python3-package create	Cree o actualice el paquete de Python 3 identificado por el nombre del paquete.	Extension	GA
az automation python3-package delete	Elimine el paquete de Python 3 por nombre.	Extension	GA
az automation python3-package list	Recupere una lista de paquetes de Python 3.	Extension	GA
az automation python3-package show	Recupere el paquete de Python 3 identificado por el nombre del paquete.	Extension	GA
az automation python3-package update	Cree o actualice el paquete de Python 3 identificado por el nombre del paquete.	Extension	GA
az automation runbook	Runbook de Automation.	Extension	Experimental
az automation runbook create	Cree el runbook identificado por el nombre del runbook.	Extension	Experimental
az automation runbook delete	Elimine el runbook por nombre.	Extension	Experimental
az automation runbook list	Recuperar una lista de runbooks.	Extension	Experimental
az automation runbook publish	Publicar borrador de runbook.	Extension	Experimental

Nombre	Description	Tipo	Estado
az automation runbook replace-content	Reemplace el contenido del runbook.	Extension	Experimental
az automation runbook revert-to-published	Revierta el contenido del runbook al último estado publicado conocido.	Extension	Experimental
az automation runbook show	Recupere el runbook identificado por el nombre del runbook.	Extension	Experimental
az automation runbook start	Inicie el runbook.	Extension	Experimental
az automation runbook update	Actualice el runbook identificado por el nombre del runbook.	Extension	Experimental
az automation runbook wait	Coloque la CLI en un estado de espera hasta que se cumpla una condición del runbook de automation.	Extension	Experimental
az automation runtime-environment	Administrar runtimeEnvironment.	Extension	GA
az automation runtime-environment create	Crear entorno en tiempo de ejecución.	Extension	GA
az automation runtime-environment delete	Elimine el entorno en tiempo de ejecución.	Extension	GA
az automation runtime-environment list	Enumerar una lista de runtimeEnvironments.	Extension	GA
az automation runtime-environment package	Administrar el paquete para el entorno en tiempo de ejecución.	Extension	GA
az automation runtime-environment package create	Cree el paquete identificado por el nombre del paquete.	Extension	GA
az automation runtime-environment package delete	Elimine el paquete por nombre.	Extension	GA
az automation runtime-environment package list	Enumerar la lista de paquetes.	Extension	GA
az automation runtime-environment package show	Obtenga el paquete identificado por nombre del paquete.	Extension	GA
az automation runtime-environment package	Actualice el paquete identificado por el nombre del paquete.	Extension	GA

Nombre	Description	Tipo	Estado
update			
az automation runtime-environment show	Obtenga información sobre el entorno en tiempo de ejecución.	Extension	GA
az automation runtime-environment update	Actualizar entorno en tiempo de ejecución.	Extension	GA
az automation schedule	Programación de automatización.	Extension	GA
az automation schedule create	Cree una programación de automatización.	Extension	GA
az automation schedule delete	Elimine una programación de automatización.	Extension	GA
az automation schedule list	Recuperar una lista de programaciones.	Extension	GA
az automation schedule show	Recupere la programación identificada por el nombre de la programación.	Extension	GA
az automation schedule update	Actualizar una programación de automatización.	Extension	GA
az automation software-update-configuration	Software-update-configuration de Automation.	Extension	GA
az automation software-update-configuration create	Cree software-update-configuration de automatización.	Extension	GA
az automation software-update-configuration delete	Elimine una configuración de software-update-configuration de automatización.	Extension	GA
az automation software-update-configuration list	Enumere todas las configuraciones de software-update-configuration para la cuenta.	Extension	GA
az automation software-update-configuration machine-runs	Se ejecuta la máquina de configuración de actualizaciones de software.	Extension	GA
az automation software-update-configuration machine-runs list	Enumerar las ejecuciones de máquina de configuración de actualizaciones de software.	Extension	GA
az automation software-update-configuration machine-runs show	Obtiene una única máquina de configuración de actualizaciones de software que ejecuta id.	Extension	GA

Nombre	Description	Tipo	Estado
az automation software-update-configuration runs	Se ejecuta la configuración de actualización de software.	Extension	GA
az automation software-update-configuration runs list	Devuelve la lista de ejecuciones de configuración de actualizaciones de software.	Extension	GA
az automation software-update-configuration runs show	Obtener una única configuración de actualización de software se ejecuta por id.	Extension	GA
az automation software-update-configuration show	Obtenga una única configuración de actualización de software por nombre.	Extension	GA
az automation source-control	Administrar el control de código fuente.	Extension	GA
az automation source-control create	Cree un control de código fuente.	Extension	GA
az automation source-control delete	Elimine el control de código fuente.	Extension	GA
az automation source-control list	Enumerar una lista de controles de origen.	Extension	GA
az automation source-control show	Obtenga el control de código fuente identificado por el nombre del control de código fuente.	Extension	GA
az automation source-control sync-job	Administrar trabajos de sincronización de control de código fuente.	Extension	GA
az automation source-control sync-job create	Cree el trabajo de sincronización para un control de código fuente.	Extension	GA
az automation source-control sync-job list	Enumerar una lista de trabajos de sincronización de control de código fuente.	Extension	GA
az automation source-control sync-job show	Obtenga el trabajo de sincronización del control de código fuente identificado por el identificador de trabajo.	Extension	GA
az automation source-control sync-job stream	Administrar el flujo de trabajo de sincronización.	Extension	GA
az automation source-control sync-job stream list	Enumere una lista de flujos de trabajo de sincronización identificados por el identificador de trabajo de sincronización.	Extension	GA

Nombre	Description	Tipo	Estado
az automation source-control sync-job stream show	Obtenga un flujo de trabajo de sincronización identificado por el identificador de secuencia.	Extension	GA
az automation source-control sync-job update	Actualice el trabajo de sincronización de un control de código fuente.	Extension	GA
az automation source-control update	Actualice un control de código fuente.	Extension	GA

Az.Automation Module

En este tema se muestran temas de ayuda para los cmdlets de Azure Automation.

Automation

Expandir tabla

Cmdlet	Description
Export-AzAutomationDscConfiguration	Exports a DSC configuration from Automation to a local file.
Export-AzAutomationDscNodeReportContent	Exports the raw content of a DSC report sent from a DSC node to Automation.
Export-AzAutomationRunbook	Exports an Automation runbook.
Get-AzAutomationAccount	Gets Automation accounts in a resource group.
Get-AzAutomationCertificate	Gets Automation certificates.
Get-AzAutomationConnection	Gets an Automation connection.
Get-AzAutomationCredential	Gets Automation credentials.
Get-AzAutomationDscCompilationJob	Gets DSC compilation jobs in Automation.
Get-AzAutomationDscCompilationJobOutput	Gets the logging streams of an Automation DSC compilation job.
Get-AzAutomationDscConfiguration	Gets DSC configurations from Automation.
Get-AzAutomationDscNode	Gets DSC nodes from Automation.
Get-AzAutomationDscNodeConfiguration	Gets metadata for DSC node configurations in Automation.
Get-AzAutomationDscNodeConfigurationDeployment	Gets DSC Node configuration deployments in Automation.
Get-AzAutomationDscNodeConfigurationDeploymentSchedule	Gets a DSC Node configuration deployment job schedule in Automation.
Get-AzAutomationDscNodeReport	Gets reports sent from a DSC node to Automation.

Cmdlet	Description
Get-AzAutomationDscOnboardingMetaconfig	Creates meta-configuration .mof files.
Get-AzAutomationHybridRunbookWorker	Gets a Hybrid Runbook Worker.
Get-AzAutomationHybridRunbookWorkerGroup	Gets a Hybrid Runbook Worker Group.
Get-AzAutomationHybridWorkerGroup	Gets hybrid runbook worker groups.
Get-AzAutomationJob	Gets Automation runbook jobs.
Get-AzAutomationJobOutput	Gets the output of an Automation job.
Get-AzAutomationJobOutputRecord	Gets the full output of an Automation job output record.
Get-AzAutomationModule	Gets metadata for modules from Automation.
Get-AzAutomationPython3Package	Gets a Python3 Package.
Get-AzAutomationRegistrationInfo	Gets registration information for onboarding a DSC node or hybrid worker to Automation.
Get-AzAutomationRunbook	Gets a runbook.
Get-AzAutomationSchedule	Gets an Automation schedule.
Get-AzAutomationScheduledRunbook	Gets Automation runbooks and associated schedules.
Get-AzAutomationSoftwareUpdateConfiguration	Gets a list of azure automation software update configurations.
Get-AzAutomationSoftwareUpdateMachineRun	Gets a list of azure automation software update configuration machine runs.
Get-AzAutomationSoftwareUpdateRun	Gets a list of azure automation software update runs.
Get-AzAutomationSourceControl	Gets a list of Azure Automation source controls.
Get-AzAutomationSourceControlSyncJob	Gets Azure Automation source control sync jobs.
Get-AzAutomationSourceControlSyncJobOutput	Gets the output of an Azure Automation source control sync job.
Get-AzAutomationVariable	Gets an Automation variable.
Get-AzAutomationWebhook	Gets webhooks from Automation.

Cmdlet	Description
Import-AzAutomationDscConfiguration	Imports a DSC configuration into Automation.
Import-AzAutomationDscNodeConfiguration	Imports a MOF document as a DSC node configuration in Automation.
Import-AzAutomationRunbook	Imports an Automation runbook.
Move-AzAutomationHybridRunbookWorker	Moves a runbook worker from one runbook worker group to the other.
New-AzAutomationAccount	Creates an Automation account.
New-AzAutomationCertificate	Creates an Automation certificate.
New-AzAutomationConnection	Creates an Automation connection.
New-AzAutomationCredential	Creates an Automation credential.
New-AzAutomationHybridRunbookWorker	Create a Runbook Worker.
New-AzAutomationHybridRunbookWorkerGroup	Creates a Runbook Worker Group.
New-AzAutomationKey	Regenerates registration keys for an Automation account.
New-AzAutomationModule	Imports a module into Automation.
New-AzAutomationPython3Package	Add Python3 Package to Automation account
New-AzAutomationRunbook	Creates an Automation runbook.
New-AzAutomationSchedule	Creates an Automation schedule.
New-AzAutomationSoftwareUpdateConfiguration	Creates a scheduled azure automation software update configuration.
New-AzAutomationSourceControl	Creates an Azure Automation source control.
New-AzAutomationUpdateManagementAzureQuery	Creates an Update Management Azure Query object.
New-AzAutomationVariable	Creates an Automation variable.
New-AzAutomationWebhook	Creates a webhook for an Automation runbook.
Publish-AzAutomationRunbook	Publishes a runbook.
Register-AzAutomationDscNode	Registers an Azure virtual machine running Windows OS as a DSC node for an Automation account.

Cmdlet	Description
Register-AzAutomationScheduledRunbook	Associates a runbook to a schedule.
Remove-AzAutomationAccount	Removes an Automation account.
Remove-AzAutomationCertificate	Removes an Automation certificate.
Remove-AzAutomationConnection	Removes an Automation connection.
Remove-AzAutomationConnectionType	Removes an Automation connection type.
Remove-AzAutomationCredential	Removes an Automation credential.
Remove-AzAutomationDscConfiguration	Removes DSC configurations from Automation.
Remove-AzAutomationDscNodeConfiguration	Removes metadata from DSC node configurations in Automation.
Remove-AzAutomationHybridRunbookWorker	Removes a hybrid runbook worker.
Remove-AzAutomationHybridRunbookWorkerGroup	Removes a hybrid runbook worker group.
Remove-AzAutomationHybridWorkerGroup	Removes hybrid worker group from Automation.
Remove-AzAutomationModule	Removes a module from Automation.
Remove-AzAutomationPython3Package	Remove a Python3 Package from Automation Account
Remove-AzAutomationRunbook	Removes a runbook.
Remove-AzAutomationSchedule	Deletes an Automation schedule.
Remove-AzAutomationSoftwareUpdateConfiguration	Removes an azure automation software update configuration.
Remove-AzAutomationSourceControl	Removes an Azure Automation source control.
Remove-AzAutomationVariable	Removes an Automation variable.
Remove-AzAutomationWebhook	Removes a webhook from an Automation runbook.
Resume-AzAutomationJob	Resumes a suspended Automation job (Only applicable for PowerShell Workflow jobs).
Set-AzAutomationAccount	Modifies an Automation account.
Set-AzAutomationCertificate	Modifies the configuration of an Automation certificate.

Cmdlet	Description
Set-AzAutomationConnectionFieldValue	Modifies the value of a field in an Automation connection.
Set-AzAutomationCredential	Modifies an Automation credential.
Set-AzAutomationDscNode	Modifies the node configuration that a DSC node is mapped to.
Set-AzAutomationHybridRunbookWorkerGroup	Updates a hybrid runbook worker group.
Set-AzAutomationModule	Updates a module in Automation.
Set-AzAutomationPython3Package	Update Python3 Package in Automation account
Set-AzAutomationRunbook	Modifies a runbook.
Set-AzAutomationSchedule	Modifies an Automation schedule.
Set-AzAutomationVariable	Modifies an Automation variable.
Set-AzAutomationWebhook	Modifies a webhook for an Automation runbook.
Start-AzAutomationDscCompilationJob	Compiles a DSC configuration in Automation.
Start-AzAutomationDscNodeConfigurationDeployment	Deploys a DSC Node configuration in Automation.
Start-AzAutomationRunbook	Starts a runbook job.
Start-AzAutomationSourceControlSyncJob	Starts an Azure Automation source control sync job.
Stop-AzAutomationDscNodeConfigurationDeployment	Stops a DSC Node configuration deployment in Automation. It only stops the current deployment job but does not unassign already assigned node configurations.
Stop-AzAutomationJob	Stops an Automation job.
Suspend-AzAutomationJob	Suspends an Automation job (Only applicable for PowerShell Workflow jobs).
Unregister-AzAutomationDscNode	Removes a DSC node from management by an Automation account.
Unregister-AzAutomationScheduledRunbook	Removes an association between a runbook and a schedule.

Cmdlet	Description
Update-AzAutomationSourceControl	Updates an Azure Automation source control.

Microsoft.Azure.Management.Automation Namespace

Classes

Expandir tabla

Name	Description
ActivityOperationsExtensions	Extension methods for ActivityOperations.
AgentRegistrationOperationExtensions	
AutomationAccountOperationsExtensions	Extension methods for AutomationAccountOperations.
AutomationManagementClient	
AutomationManagementClientExtensions	
CertificateOperationsExtensions	Extension methods for CertificateOperations.
ConnectionOperationsExtensions	Extension methods for ConnectionOperations.
ConnectionTypeOperationsExtensions	Extension methods for ConnectionTypeOperations.
CredentialOperationsExtensions	Extension methods for CredentialOperations.
DscCompilationJobOperationsExtensions	Extension methods for DscCompilationJobOperations.
DscConfigurationOperationsExtensions	Extension methods for DscConfigurationOperations.
DscNodeConfigurationOperationsExtensions	Extension methods for DscNodeConfigurationOperations.
DscNodeOperationsExtensions	Extension methods for DscNodeOperations.
DscNodeReportsOperationsExtensions	
HybridRunbookWorkerGroupOperations Extensions	Extension methods for HybridRunbookWorkerGroupOperations.
JobOperationsExtensions	Extension methods for JobOperations.
JobScheduleOperationsExtensions	Extension methods for JobScheduleOperations.
JobStreamOperationsExtensions	Extension methods for JobStreamOperations.
ModuleOperationsExtensions	Extension methods for ModuleOperations.
ObjectDataTypeOperationsExtensions	

Name	Description
ResponseWithSkipToken<T>	
RunbookDraftOperationsExtensions	Extension methods for RunbookDraftOperations.
RunbookOperationsExtensions	Extension methods for RunbookOperations.
ScheduleOperationsExtensions	Extension methods for ScheduleOperations.
StatisticsOperationsExtensions	Extension methods for StatisticsOperations.
TestJobOperationsExtensions	Extension methods for TestJobOperations.
TypeFieldOperationsExtensions	
UsageOperationsExtensions	
VariableOperationsExtensions	Extension methods for VariableOperations.
WebhookOperationsExtensions	Extension methods for WebhookOperations.

Interfaces

Expandir tabla

Name	Description
IActivityOperations	ActivityOperations operations.
IAgentRegistrationOperation	
IAutomationAccountOperations	AutomationAccountOperations operations.
IAutomationManagementClient	
ICertificateOperations	CertificateOperations operations.
IConnectionOperations	ConnectionOperations operations.
IConnectionTypeOperations	ConnectionTypeOperations operations.
ICredentialOperations	CredentialOperations operations.
IDscCompilationJobOperations	DscCompilationJobOperations operations.
IDscConfigurationOperations	DscConfigurationOperations operations.
IDscNodeConfigurationOperations	DscNodeConfigurationOperations operations.
IDscNodeOperations	DscNodeOperations operations.

Name	Description
IDscNodeReportsOperations	
IHybridRunbookWorkerGroupOperations	HybridRunbookWorkerGroupOperations operations.
IJobOperations	JobOperations operations.
IJobScheduleOperations	JobScheduleOperations operations.
IJobStreamOperations	JobStreamOperations operations.
IModuleOperations	ModuleOperations operations.
IObjectDataTypeOperations	
IRunbookDraftOperations	RunbookDraftOperations operations.
IRunbookOperations	RunbookOperations operations.
IScheduleOperations	ScheduleOperations operations.
IStatisticsOperations	StatisticsOperations operations.
ITestJobOperations	TestJobOperations operations.
ITypeFieldOperations	
IUsageOperations	
IVariableOperations	VariableOperations operations.
IWebhookOperations	WebhookOperations operations.

Referencia de API de REST de Automation

Artículo • 07/02/2024

El servicio Azure Automation proporciona un motor de ejecución de flujo de trabajo escalable y altamente confiable para automatizar con frecuencia las tareas administradas que se repiten. Los procesos se automatizan mediante runbooks, que son flujos de trabajo de Windows PowerShell que se ejecutan en el motor de ejecución de Azure Automation. Puede usar esta API para crear, actualizar, leer y eliminar recursos de automatización, incluidos runbooks y trabajos de runbook. Además, puede administrar activos, como variables, programaciones, módulos de Windows PowerShell, credenciales y certificados.

Consulte también

- Documentación de Azure Automation

Definiciones integradas de Azure Policy para Azure Automation

15/08/2025

Esta página es un índice de las definiciones de directivas integradas de Azure Policy para Azure Automation. Puede encontrar elementos integrados adicionales de Azure Policy para otros servicios en Definiciones de elementos integrados de Azure Policy.

El nombre de cada definición de directiva integrada se vincula a la definición de directiva en Azure Portal. Use el vínculo de la columna **Versión** para ver el origen en el repositorio de GitHub de Azure Policy.

Azure Automation

Expandir tabla

Nombre (Azure Portal)	Descripción	Efectos	Versión (GitHub)
La cuenta de Automation debería tener una identidad administrada	Use identidades administradas como método recomendado para autenticarse con recursos de Azure desde los runbooks. La identidad administrada para la autenticación es más segura y elimina la sobrecarga de administración asociada al uso de la cuenta de ejecución en el código de runbook.	Auditar, Deshabilitado	1.0.0
Las variables de cuenta de Automation deben cifrarse	Es importante habilitar el cifrado de recursos de variables de cuentas de Automation al almacenar datos confidenciales.	Auditar, Denegar, Deshabilitado	1.1.0
Las cuentas de Automation deben deshabilitar el acceso a la red pública	Al deshabilitar el acceso a la red pública, se mejora la seguridad ya que el recurso no se expone en la red pública de Internet. En su lugar, puede limitar la exposición de los recursos de la cuenta de Automation mediante la creación de puntos de conexión privados. Más información en: https://docs.microsoft.com/azure/automation/how-to/private-link-security.	Auditar, Denegar, Deshabilitado	1.0.0
La cuenta de Azure Automation debe tener el método de autenticación local deshabilitado	Deshabilitar los métodos de autenticación local mejora la seguridad al garantizar que las cuentas de Azure Automation requiera exclusivamente identidades de Azure Active Directory para la autenticación.	Auditar, Denegar, Deshabilitado	1.0.0
Las cuentas de Azure Automation deben usar claves administradas por el cliente para cifrar los datos en reposo.	Use claves administradas por el cliente para administrar el cifrado en reposo de sus cuentas de Azure Automation. De manera predeterminada, los datos del cliente se cifran con claves administradas por el servicio, pero las claves administradas por el cliente suelen ser necesarias para cumplir estándares de cumplimiento normativo. Las claves administradas por el cliente permiten cifrar los datos con una clave de Azure Key Vault creada por el usuario y propiedad de este. Tiene control y	Auditar, Denegar, Deshabilitado	1.0.0

Nombre (Azure Portal)	Descripción	Efectos	Versión (GitHub)
	responsabilidad totales del ciclo de vida de la clave, incluidos la rotación y administración. Obtenga más información en https://aka.ms/automation-cmk.		
Configurar la cuenta de Azure Automation para deshabilitar la autenticación local	Deshabilite los métodos de autenticación local para que las cuentas de Azure Automation requieran exclusivamente identidades de Azure Active Directory para la autenticación.	Modificar, Deshabilitado	1.0.0
Configurar cuentas de Azure Automation para deshabilitar el acceso a la red pública	Deshabilite el acceso a la red pública para la cuenta de Azure Automation de modo que no sea accesible a través de la red pública de Internet. Esta configuración le ayudará a protegerlo contra los riesgos de pérdida de datos. En su lugar, puede limitar la exposición de los recursos de la cuenta de Automation mediante la creación de puntos de conexión privados. Más información en: https://aka.ms/privateendpoints.	Modificar, Deshabilitado	1.0.0
Configurar conexiones de punto de conexión privado en cuentas de Azure Automation	Las conexiones de punto de conexión privado permiten la comunicación segura al habilitar la conectividad privada con cuentas de Azure Automation sin necesidad de direcciones IP públicas en el origen o el destino. Obtenga más información sobre los puntos de conexión privados en Azure Automation en https://docs.microsoft.com/azure/automation/how-to/private-link-security.	DeployIfNotExists, deshabilitado	1.0.0
Habilitación del registro por grupo de categorías para cuentas de Automation (microsoft.automation/automationaccounts) en un centro de eventos	Los registros de recursos deben habilitarse para realizar un seguimiento de las actividades y eventos que tienen lugar en los recursos y proporcionarle visibilidad e información sobre los cambios que se produzcan. Esta directiva implementa una configuración de diagnóstico mediante un grupo de categorías para enrutar los registros a un centro de eventos para cuentas de Automation (microsoft.automation/automationaccounts).	DesplegarSiNoExiste, AuditarSiNoExiste, Deshabilitado	1.2.0
Habilitación del registro por grupo de categorías para cuentas de Automation (microsoft.automation/automationaccounts) en Log Analytics	Los registros de recursos deben habilitarse para realizar un seguimiento de las actividades y eventos que tienen lugar en los recursos y proporcionarle visibilidad e información sobre los cambios que se produzcan. Esta directiva implementa una configuración de diagnóstico mediante un grupo de categorías para enrutar los registros a un área de trabajo de Log Analytics para cuentas de Automation (microsoft.automation/automationaccounts).	DesplegarSiNoExiste, AuditarSiNoExiste, Deshabilitado	1.1.0
Habilitación del registro por grupo de categorías para cuentas de Automation (microsoft.automation/automationaccounts) en Storage	Los registros de recursos deben habilitarse para realizar un seguimiento de las actividades y eventos que tienen lugar en los recursos y proporcionarle visibilidad e información sobre los cambios que se produzcan. Esta directiva implementa una configuración de diagnóstico	DesplegarSiNoExiste, AuditarSiNoExiste, Deshabilitado	1.1.0

Nombre (Azure Portal)	Descripción	Efectos	Versión (GitHub)
	mediante un grupo de categorías para enrutar los registros a una cuenta de almacenamiento para cuentas de Automation (microsoft.automation/automationaccounts).		
Las conexiones de punto de conexión privado en cuentas de Automation deben estar habilitadas	Las conexiones de punto de conexión privado permiten la comunicación segura al habilitar la conectividad privada con cuentas de Automation sin necesidad de direcciones IP públicas en el origen o el destino. Obtenga más información sobre los puntos de conexión privados en Azure Automation en https://docs.microsoft.com/azure/automation/how-to/private-link-security.	AuditIfNotExists, Deshabilitado	1.0.0

Pasos siguientes

- Los elementos integrados se pueden encontrar en el repositorio de GitHub de Azure Policy.
- Revise la estructura de definición de Azure Policy.
- Vea la Descripción de los efectos de directivas.