Contents

Managed identities for Azure resources
Overview
  About managed identities for Azure resources
Quickstarts
  Use a VM managed identity to access Azure Resource Manager
    Windows VM
    Linux VM
Tutorials
  Windows VM/VMSS
    Access Azure Data Lake Store
    Access Azure Storage
    Access Azure Storage using an access key
    Access Azure Storage using SAS
    Access Azure SQL
    Access Azure Key Vault
    Access Azure Cosmos DB
    Access Azure AD Graph API
    Access Azure Resource Manager with a user-assigned managed identity
  Linux VM/VMSS
    Access Azure Data Lake Store
    Access Azure Storage
    Access Azure Storage using an access key
    Access Azure Storage using SAS
    Access Azure Key Vault
    Access Azure Cosmos DB
    Access Azure Container Registry
    Access Azure AD Graph API
    Access Azure Resource Manager with a user-assigned managed identity
  Azure Service Bus

- Azure App Service and Functions
- Azure Event Hubs
- Azure Container Instances

How-to guides

- Configure managed identities on VMs
  - Portal
  - CLI
  - PowerShell
  - Azure Resource Manager Template
  - REST
  - Azure SDKs
- Configure managed identities on virtual machine scale sets
  - Portal
  - CLI
  - PowerShell
  - Azure Resource Manager Template
  - REST
- Use managed identities on VMs
  - Acquire an access token
  - Sign to Powershell and CLI using Managed Identity
  - Use with Azure SDKs
  - Stop using VM extension and start using Azure IMDS
- Assign a managed identity access to another Azure resource
  - Portal
  - CLI
  - PowerShell
- Manage user-assigned managed identities
  - Portal
  - CLI
  - PowerShell
  - Azure Resource Manager Template
  - REST

    - View managed identity service principals
        - Portal
        - CLI
        - PowerShell
- Reference
    - CLI
    - PowerShell
    - .NET
    - Java
    - Python
    - Ruby
    - Node.js
    - Go
- Resources
    - FAQs and known issues
    - Azure services that support managed identities for Azure resources
    - Stack Overflow
    - AAD Developers forum

What is managed identities for Azure resources?

4/5/2019 • 8 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

A common challenge when building cloud applications is how to manage the credentials in your code for authenticating to cloud services. Keeping the credentials secure is an important task. Ideally, the credentials never appear on developer workstations and aren't checked into source control. Azure Key Vault provides a way to securely store credentials, secrets, and other keys, but your code has to authenticate to Key Vault to retrieve them.

The managed identities for Azure resources feature in Azure Active Directory (Azure AD) solves this problem. The feature provides Azure services with an automatically managed identity in Azure AD. You can use the identity to authenticate to any service that supports Azure AD authentication, including Key Vault, without any credentials in your code.

The managed identities for Azure resources feature is free with Azure AD for Azure subscriptions. There's no additional cost.

NOTE

Managed identities for Azure resources is the new name for the service formerly known as Managed Service Identity (MSI).

Terminology

The following terms are used throughout the managed identities for Azure resources documentation set:

- **Client ID** - a unique identifier generated by Azure AD that is tied to an application and service principal during its initial provisioning.
- **Principal ID** - the object ID of the service principal object for your managed identity that is used to grant role-based access to an Azure resource.
- **Azure Instance Metadata Service (IMDS)** - a REST endpoint accessible to all IaaS VMs created via the Azure Resource Manager. The endpoint is available at a well-known non-routable IP address (169.254.169.254) that can be accessed only from within the VM.

How does the managed identities for Azure resources work?

There are two types of managed identities:

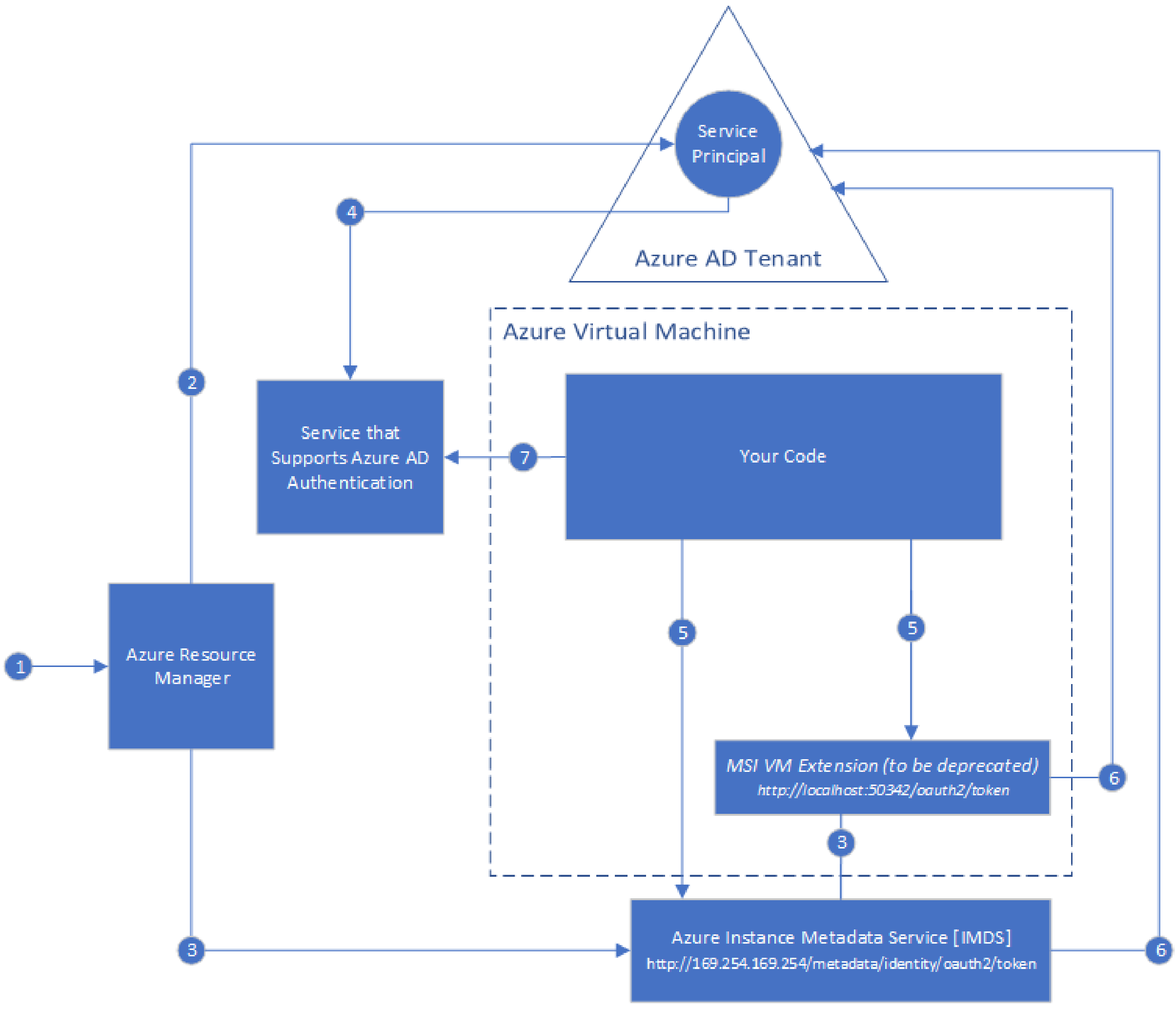
- A **system-assigned managed identity** is enabled directly on an Azure service instance. When the identity is enabled, Azure creates an identity for the instance in the Azure AD tenant that's trusted by the subscription of the instance. After the identity is created, the credentials are provisioned onto the instance. The lifecycle of a system-assigned identity is directly tied to the Azure service instance that it's enabled on. If the instance is deleted, Azure automatically cleans up the credentials and the identity in Azure AD.

- A **user-assigned managed identity** is created as a standalone Azure resource. Through a create process, Azure creates an identity in the Azure AD tenant that's trusted by the subscription in use. After the identity is created, the identity can be assigned to one or more Azure service instances. The lifecycle of a user-assigned identity is managed separately from the lifecycle of the Azure service instances to which it's assigned.

Internally, managed identities are service principals of a special type, which are locked to only be used with Azure resources. When the managed identity is deleted, the corresponding service principal is automatically removed.

Your code can use a managed identity to request access tokens for services that support Azure AD authentication. Azure takes care of rolling the credentials that are used by the service instance.

The following diagram shows how managed service identities work with Azure virtual machines (VMs):


Service Principal
Azure AD Tenant
Azure Virtual Machine
Service that Supports Azure AD Authentication
Your Code
Azure Resource Manager
MSI VM Extension (to be deprecated)
http://localhost:50342/oauth2/token
Azure Instance Metadata Service [IMDS]
http://169.254.169.254/metadata/identity/oauth2/token
1
2
3
3
4
5
5
6
6
7

PROPERTY	SYSTEM-ASSIGNED MANAGED IDENTITY	USER-ASSIGNED MANAGED IDENTITY
Creation	Created as part of an Azure resource (for example, an Azure virtual machine or Azure App Service)	Created as a stand-alone Azure resource
Lifecycle	Shared lifecycle with the Azure resource that the managed identity is created with. When the parent resource is deleted, the managed identity is deleted as well.	Independent life-cycle. Must be explicitly deleted.

PROPERTY	SYSTEM-ASSIGNED MANAGED IDENTITY	USER-ASSIGNED MANAGED IDENTITY
Sharing across Azure resources	Cannot be shared. It can only be associated with a single Azure resource.	Can be shared The same user-assigned managed identity can be associated with more than one Azure resource.
Common use cases	Workloads that are contained within a single Azure resource Workloads for which you need independent identities. For example, an application that runs on a single virtual machine	Workloads that run on multiple resources and which can share a single identity. Workloads that need pre-authorization to a secure resource as part of a provisioning flow. Workloads where resources are recycled frequently, but permissions should stay consistent. For example, a workload where multiple virtual machines need to access the same resource

How a system-assigned managed identity works with an Azure VM

1. Azure Resource Manager receives a request to enable the system-assigned managed identity on a VM.
2. Azure Resource Manager creates a service principal in Azure AD for the identity of the VM. The service principal is created in the Azure AD tenant that's trusted by the subscription.
3. Azure Resource Manager configures the identity on the VM:
   a. Updates the Azure Instance Metadata Service identity endpoint with the service principal client ID and certificate.
   b. Provisions the VM extension (planned for deprecation in January 2019), and adds the service principal client ID and certificate. (This step is planned for deprecation.)
4. After the VM has an identity, use the service principal information to grant the VM access to Azure resources. To call Azure Resource Manager, use role-based access control (RBAC) in Azure AD to assign the appropriate role to the VM service principal. To call Key Vault, grant your code access to the specific secret or key in Key Vault.
5. Your code that's running on the VM can request a token from the Azure Instance Metadata service endpoint, accessible only from within the VM: `http://169.254.169.254/metadata/identity/oauth2/token`
   - The resource parameter specifies the service to which the token is sent. To authenticate to Azure Resource Manager, use `resource=https://management.azure.com/`.
   - API version parameter specifies the IMDS version, use api-version=2018-02-01 or greater.

> **NOTE**
>
> Your code can also request a token from VM extension endpoint, but this is planned for deprecation soon. For more information about the VM extension, see Migrate from the VM extension to Azure IMDS for authentication.

6. A call is made to Azure AD to request an access token (as specified in step 5) by using the client ID and certificate configured in step 3. Azure AD returns a JSON Web Token (JWT) access token.
7. Your code sends the access token on a call to a service that supports Azure AD authentication.

How a user-assigned managed identity works with an Azure VM

1. Azure Resource Manager receives a request to create a user-assigned managed identity.

2. Azure Resource Manager creates a service principal in Azure AD for the user-assigned managed identity. The service principal is created in the Azure AD tenant that's trusted by the subscription.

3. Azure Resource Manager receives a request to configure the user-assigned managed identity on a VM:
   a. Updates the Azure Instance Metadata Service identity endpoint with the user-assigned managed identity service principal client ID and certificate.
   b. Provisions the VM extension, and adds the user-assigned managed identity service principal client ID and certificate. (This step is planned for deprecation.)
4. After the user-assigned managed identity is created, use the service principal information to grant the identity access to Azure resources. To call Azure Resource Manager, use RBAC in Azure AD to assign the appropriate role to the service principal of the user-assigned identity. To call Key Vault, grant your code access to the specific secret or key in Key Vault.

   > **NOTE**
   > You can also do this step before step 3.

5. Your code that's running on the VM can request a token from the Azure Instance Metadata Service identity endpoint, accessible only from within the VM: `http://169.254.169.254/metadata/identity/oauth2/token`
   - The resource parameter specifies the service to which the token is sent. To authenticate to Azure Resource Manager, use `resource=https://management.azure.com/`.
   - The client ID parameter specifies the identity for which the token is requested. This value is required for disambiguation when more than one user-assigned identity is on a single VM.
   - The API version parameter specifies the Azure Instance Metadata Service version. Use `api-version=2018-02-01` or higher.

> **NOTE**
> Your code can also request a token from VM extension endpoint, but this is planned for deprecation soon. For more information about the VM extension, see Migrate from the VM extension to Azure IMDS for authentication.

6. A call is made to Azure AD to request an access token (as specified in step 5) by using the client ID and certificate configured in step 3. Azure AD returns a JSON Web Token (JWT) access token.
7. Your code sends the access token on a call to a service that supports Azure AD authentication.

How can I use managed identities for Azure resources?

To learn how to use managed identities to access different Azure resources, try these tutorials.

> **NOTE**
> Check out the Implementing Managed Identities for Microsoft Azure Resources course for more information about managed identities, including detailed video walkthroughs of several supported scenarios.

Learn how to use a managed identity with a Windows VM:

- Access Azure Data Lake Store
- Access Azure Resource Manager
- Access Azure SQL
- Access Azure Storage by using an access key
- Access Azure Storage by using shared access signatures
- Access a non-Azure AD resource with Azure Key Vault

Learn how to use a managed identity with a Linux VM:

- Access Azure Data Lake Store
- Access Azure Resource Manager
- Access Azure Storage by using an access key
- Access Azure Storage by using shared access signatures
- Access a non-Azure AD resource with Azure Key Vault

Learn how to use a managed identity with other Azure services:

- Azure App Service
- Azure Functions
- Azure Logic Apps
- Azure Service Bus
- Azure Event Hubs
- Azure API Management
- Azure Container Instances

What Azure services support the feature?

Managed identities for Azure resources can be used to authenticate to services that support Azure AD authentication. For a list of Azure services that support the managed identities for Azure resources feature, see Services that support managed identities for Azure resources.

Next steps

Get started with the managed identities for Azure resources feature with the following quickstarts:

- Use a Windows VM system-assigned managed identity to access Resource Manager
- Use a Linux VM system-assigned managed identity to access Resource Manager

Use a Windows VM system-assigned managed identity to access Resource Manager

3/26/2019 • 3 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This quickstart shows you how to access the Azure Resource Manager API using a Windows virtual machine with system-assigned managed identity enabled. Managed identities for Azure resources are automatically managed by Azure and enable you to authenticate to services that support Azure AD authentication without needing to insert credentials into your code. You learn how to:

- Grant your VM access to a Resource Group in Azure Resource Manager
- Get an access token using the VM identity and use it to call Azure Resource Manager

Prerequisites

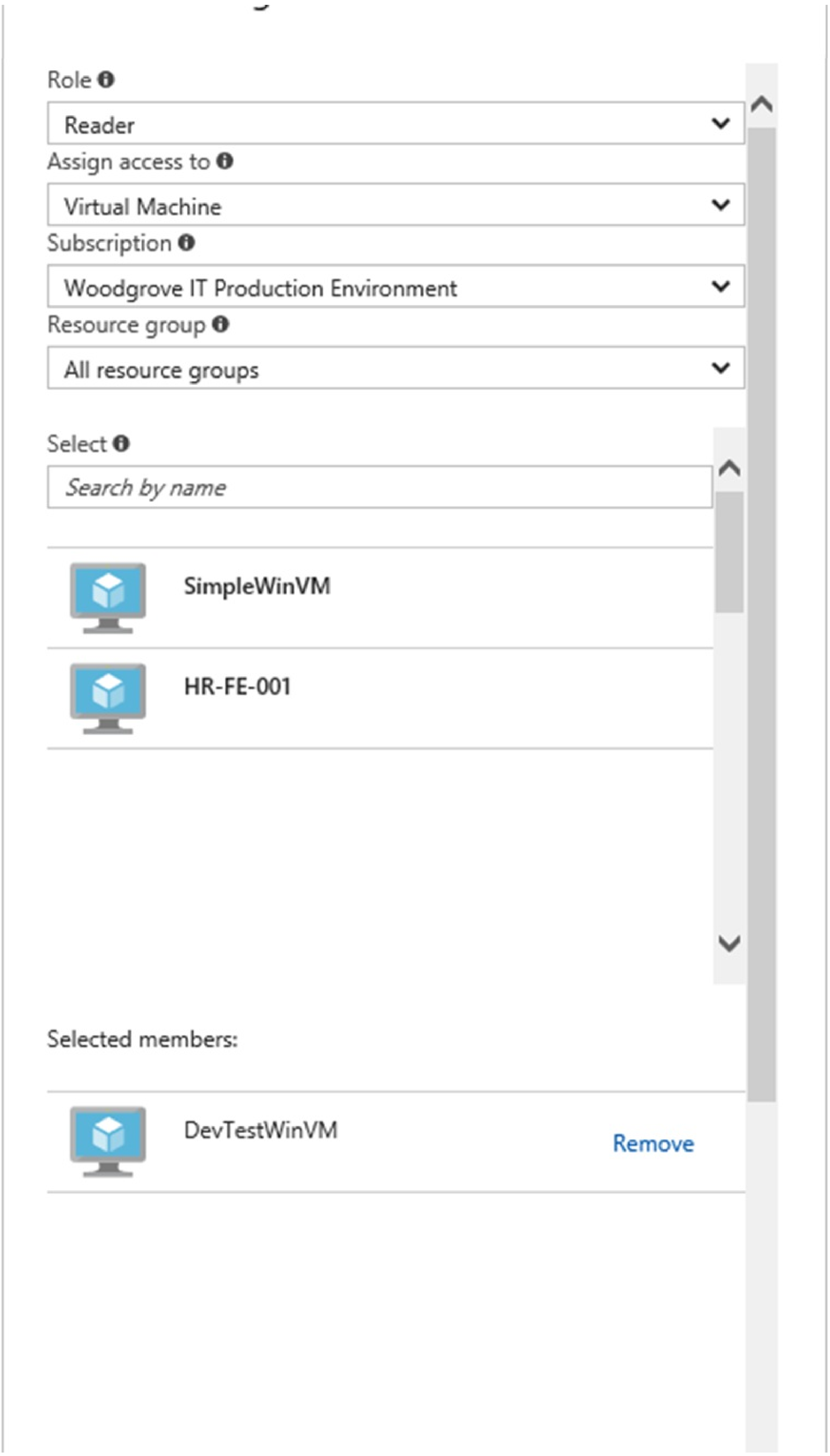
- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Grant your VM access to a resource group in Resource Manager

Using managed identities for Azure resources, your code can get access tokens to authenticate to resources that support Azure AD authentication. The Azure Resource Manager supports Azure AD authentication. First, we need to grant this VM’s system-assigned managed identity access to a resource in Resource Manager, in this case the Resource Group in which the VM is contained.

1. Navigate to the tab for **Resource Groups**.
2. Select the specific **Resource Group** you created for your **Windows VM**.
3. Go to **Access control (IAM)** in the left panel.
4. Then **Add role assignment** a new role assignment for your **Windows VM**. Choose **Role** as **Reader**.
5. In the next drop-down, **Assign access to** the resource **Virtual Machine**.
6. Next, ensure the proper subscription is listed in the **Subscription** dropdown. And for **Resource Group**, select **All resource groups**.
7. Finally, in **Select** choose your Windows VM in the dropdown and click **Save**.

Add role assignment


Role

Reader

Assign access to

Virtual Machine

Subscription

Woodgrove IT Production Environment

Resource group

All resource groups

Select

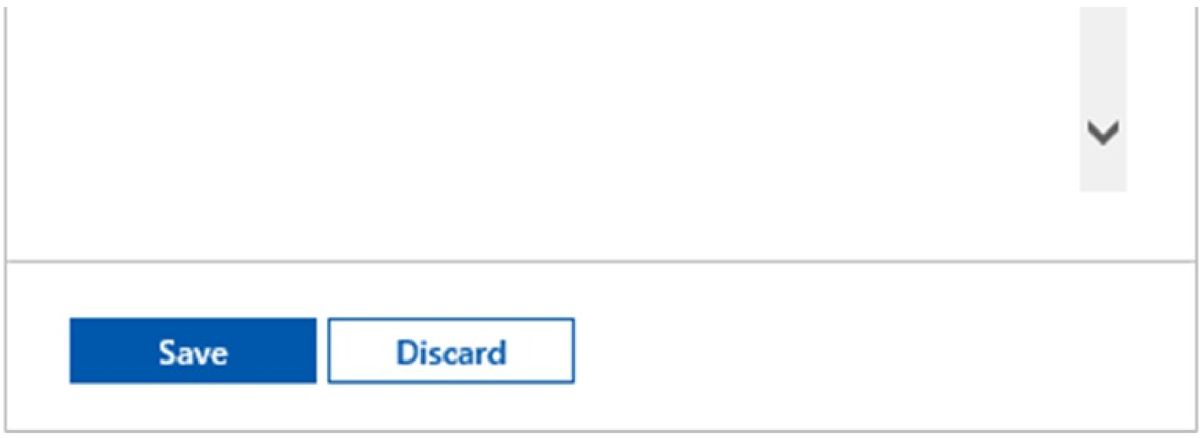
Search by name

SimpleWinVM

HR-FE-001

Selected members:

DevTestWinVM



Get an access token using the VM's system-assigned managed identity and use it to call Azure Resource Manager

You will need to use **PowerShell** in this portion. If you don't have **PowerShell** installed, download it here.

1. In the portal, navigate to **Virtual Machines** and go to your Windows virtual machine and in the **Overview**, click **Connect**.
2. Enter in your **Username** and **Password** for which you added when you created the Windows VM.
3. Now that you have created a **Remote Desktop Connection** with the virtual machine, open **PowerShell** in the remote session.
4. Using the Invoke-WebRequest cmdlet, make a request to the local managed identity for Azure resources endpoint to get an access token for Azure Resource Manager.

   ```
   $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https://management.azure.com/' -Method GET -Headers @{Metadata="true"}
   ```

   NOTE
   The value of the "resource" parameter must be an exact match for what is expected by Azure AD. When using the Azure Resource Manager resource ID, you must include the trailing slash on the URI.

   Next, extract the full response, which is stored as a JavaScript Object Notation (JSON) formatted string in the $response object.

   ```
   $content = $response.Content | ConvertFrom-Json
   ```

   Next, extract the access token from the response.

   ```
   $ArmToken = $content.access_token
   ```

   Finally, call Azure Resource Manager using the access token. In this example, we're also using the Invoke-WebRequest cmdlet to make the call to Azure Resource Manager, and include the access token in the Authorization header.

   ```
   (Invoke-WebRequest -Uri https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE GROUP>?api-version=2016-06-01 -Method GET -ContentType "application/json" -Headers @{ Authorization ="Bearer $ArmToken"}).content
   ```

NOTE

The URL is case-sensitive, so ensure if you are using the exact same case as you used earlier when you named the Resource Group, and the uppercase "G" in "resourceGroups."

The following command returns the details of the Resource Group:

```
{"id":"/subscriptions/98f51385-2edc-4b79-bed9-
7718de4cb861/resourceGroups/DevTest","name":"DevTest","location":"westus","properties":
{"provisioningState":"Succeeded"}}
```

Next steps

In this quickstart, you learned how to use a system-assigned managed identity to access the Azure Resource Manager API. To learn more about Azure Resource Manager see:

Azure Resource Manager

Use a Linux VM system-assigned managed identity to access Azure Resource Manager

3/26/2019 • 3 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This quickstart shows you how to use a system-assigned identity for a Linux virtual machine (VM) to access the Azure Resource Manager API. Managed identities for Azure resources are automatically managed by Azure and enable you to authenticate to services that support Azure AD authentication without needing to insert credentials into your code. You learn how to:

- Grant your VM access to a Resource Group in Azure Resource Manager
- Get an access token using the VM identity and use it to call Azure Resource Manager

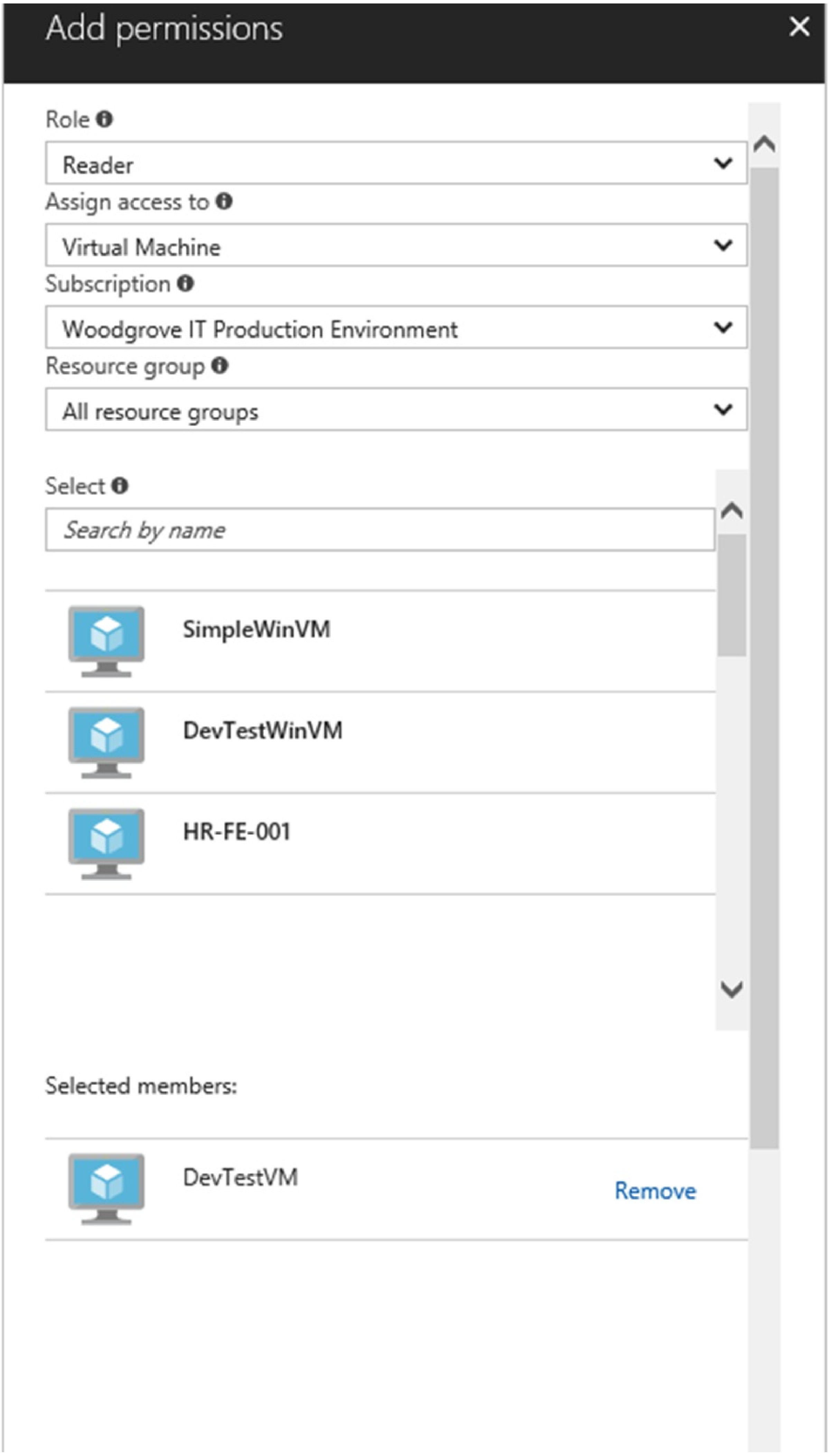
Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Grant your VM access to a Resource Group in Azure Resource Manager

Using managed identities for Azure resources, your code can get access tokens to authenticate to resources that support Azure AD authentication. The Azure Resource Manager API supports Azure AD authentication. First, we need to grant this VM's identity access to a resource in Azure Resource Manager, in this case the Resource Group in which the VM is contained.

1. Navigate to the tab for **Resource Groups**.
2. Select the specific **Resource Group** you created earlier.
3. Go to **Access control(IAM)** in the left panel.
4. Click to **Add** a new role assignment for your VM. Choose **Role** as **Reader**.
5. In the next dropdown, **Assign access to** the resource **Virtual Machine**.
6. Next, ensure the proper subscription is listed in the **Subscription** dropdown. And for **Resource Group**, select **All resource groups**.
7. Finally, in **Select** choose your Linux Virtual Machine in the dropdown and click **Save**.


Add permissions

Role
Reader

Assign access to
Virtual Machine

Subscription
Woodgrove IT Production Environment

Resource group
All resource groups

Select
Search by name

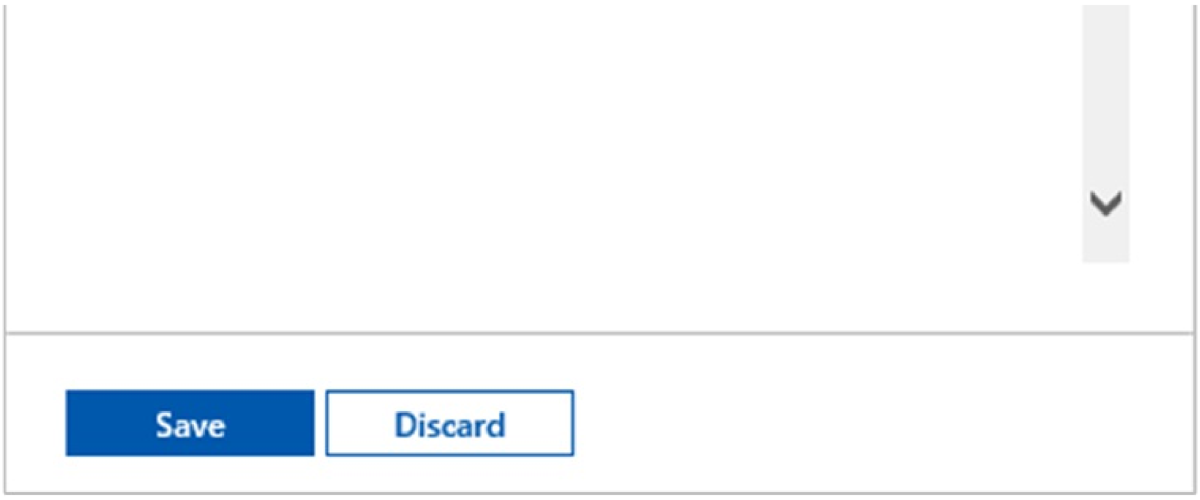
SimpleWinVM

DevTestWinVM

HR-FE-001

Selected members:

DevTestVM Remove



Get an access token using the VM's system-assigned managed identity and use it to call Resource Manager

To complete these steps, you will need an SSH client. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to Use SSH keys with Windows on Azure, or How to create and use an SSH public and private key pair for Linux VMs in Azure.

1. In the portal, navigate to your Linux VM and in the **Overview**, click **Connect**.
2. **Connect** to the VM with the SSH client of your choice.
3. In the terminal window, using `curl`, make a request to the local managed identities for Azure resources endpoint to get an access token for Azure Resource Manager. The `curl` request for the access token is below.

   ```
   curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
   01&resource=https://management.azure.com/' -H Metadata:true
   ```

   NOTE

   The value of the “resource” parameter must be an exact match for what is expected by Azure AD. In the case of the Resource Manager resource ID, you must include the trailing slash on the URI.

   The response includes the access token you need to access Azure Resource Manager.

   Response:

   ```
   {"access_token":"eyJ0eXAiOi...",
   "refresh_token":"",
   "expires_in":"3599",
   "expires_on":"1504130527",
   "not_before":"1504126627",
   "resource":"https://management.azure.com",
   "token_type":"Bearer"}
   ```

   You can use this access token to access Azure Resource Manager, for example to read the details of the Resource Group to which you previously granted this VM access. Replace the values of <SUBSCRIPTION ID>, <RESOURCE GROUP>, and <ACCESS TOKEN> with the ones you created earlier.

NOTE

The URL is case-sensitive, so ensure if you are using the exact same case as you used earlier when you named the Resource Group, and the uppercase “G” in “resourceGroup”.

```
curl https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE GROUP>?api-version=2016-09-01 -H "Authorization: Bearer <ACCESS TOKEN>"
```

The response back with the specific Resource Group information:

```
{"id":"/subscriptions/98f51385-2edc-4b79-bed9-7718de4cb861/resourceGroups/DevTest","name":"DevTest","location":"westus","properties":{"provisioningState":"Succeeded"}}
```

Next steps

In this quickstart, you learned how to use a system-assigned managed identity to access the Azure Resource Manager API. To learn more about Azure Resource Manager see:

Azure Resource Manager

Tutorial: Use a Windows VM system-assigned managed identity to access Azure Data Lake Store

3/26/2019 • 6 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Windows virtual machine (VM) to access an Azure Data Lake Store. Managed Service Identities are automatically managed by Azure and enable you to authenticate to services that support Azure AD authentication, without needing to insert credentials into your code. You learn how to:

- Grant your VM access to an Azure Data Lake Store
- Get an access token using the VM identity and use it to access an Azure Data Lake Store

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Grant your VM access to Azure Data Lake Store

Now you can grant your VM access to files and folders in an Azure Data Lake Store. For this step, you can use an existing Data Lake Store or create a new one. To create a new Data Lake Store using the Azure portal, follow this Azure Data Lake Store quickstart. There are also quickstarts that use the Azure CLI and Azure PowerShell in the Azure Data Lake Store documentation.

In your Data Lake Store, create a new folder and grant your VM's system-assigned identity permission to read, write, and execute files in that folder:

1. In the Azure portal, click **Data Lake Store** in the left-hand navigation.
2. Click the Data Lake Store you want to use for this tutorial.
3. Click **Data Explorer** in the command bar.
4. The root folder of the Data Lake Store is selected. Click **Access** in the command bar.
5. Click **Add**. In the **Select** field, enter the name of your VM, for example **DevTestVM**. Click to select your VM from the search results, then click **Select**.
6. Click **Select Permissions**. Select **Read** and **Execute**, add to **This folder**, and add as **An access permission only**. Click **Ok**. The permission should be added successfully.
7. Close the **Access** blade.
8. For this tutorial, create a new folder. Click **New Folder** in the command bar, and give the new folder a name, for

example **TestFolder**. Click **Ok**.

9. Click on the folder you created, then click **Access** in the command bar.
10. Similar to step 5, click **Add**, in the **Select** field enter the name of your VM, select it and click **Select**.
11. Similar to step 6, click **Select Permissions**, select **Read**, **Write**, and **Execute**, add to **This folder**, and add as **An access permission entry and a default permission entry**. Click **Ok**. The permission should be added successfully.

Your VM's system-assigned managed identity can now perform all operations on files in the folder you created. For more information on managing access to Data Lake Store, read this article on Access Control in Data Lake Store.

Get an access token using the VM's system-assigned managed identity and use it to call the Azure Data Lake Store filesystem

Azure Data Lake Store natively supports Azure AD authentication, so it can directly accept access tokens obtained using managed identities for Azure resources. To authenticate to the Data Lake Store filesystem you send an access token issued by Azure AD to your Data Lake Store filesystem endpoint, in an Authorization header in the format "Bearer <ACCESS_TOKEN_VALUE>". To learn more about Data Lake Store support for Azure AD authentication, read Authentication with Data Lake Store using Azure Active Directory

NOTE

The Data Lake Store filesystem client SDKs do not yet support managed identities for Azure resources. This tutorial will be updated when support is added to the SDK.

In this tutorial, you authenticate to the Data Lake Store filesystem REST API using PowerShell to make REST requests. To use the VM's system-assigned managed identity for authentication, you need to make the requests from the VM.

1. In the portal, navigate to **Virtual Machines**, go to your Windows VM, and in the **Overview** click **Connect**.
2. Enter in your **Username** and **Password** for which you added when you created the Windows VM.
3. Now that you have created a **Remote Desktop Connection** with the virtual machine, open **PowerShell** in the remote session.
4. Using PowerShell's `Invoke-WebRequest`, make a request to the local managed identities for Azure resources endpoint to get an access token for Azure Data Lake Store. The resource identifier for Data Lake Store is `https://datalake.azure.net/`. Data Lake does an exact match on the resource identifier and the trailing slash is important.

   ```
   $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https%3A%2F%2Fdatalake.azure.net%2F' -Method GET -Headers @{Metadata="true"}
   ```

   Convert the response from a JSON object to a PowerShell object.

   ```
   $content = $response.Content | ConvertFrom-Json
   ```

   Extract the access token from the response.

   ```
   $AccessToken = $content.access_token
   ```

5. Using PowerShell's `Invoke-WebRequest', make a request to your Data Lake Store's REST endpoint to list

the folders in the root folder. This is a simple way to check everything is configured correctly. It is important the string "Bearer" in the Authorization header has a capital "B". You can find the name of your Data Lake Store in the **Overview** section of the Data Lake Store blade in the Azure portal.

```
Invoke-WebRequest -Uri https://<YOUR_ADLS_NAME>.azuredatalakestore.net/webhdfs/v1/?op=LISTSTATUS -
Headers @{Authorization="Bearer $AccessToken"}
```

A successful response looks like:

```
StatusCode        : 200
StatusDescription : OK
Content           : {"FileStatuses":{"FileStatus":
[{"length":0,"pathSuffix":"TestFolder","type":"DIRECTORY", "blockSize":0,"accessTime":1507934941392,
"modificationTime":1507944835699,"replication":0, "permission":"770","ow..."
RawContent        : HTTP/1.1 200 OK
                    Pragma: no-cache
                    x-ms-request-id: b4b31e16-e968-46a1-879a-3474aa7d4528
                    x-ms-webhdfs-version: 17.04.22.00
                    Status: 0x0
                    X-Content-Type-Options: nosniff
                    Strict-Transport-Security: ma...
Forms             : {}
Headers           : {[Pragma, no-cache], [x-ms-request-id, b4b31e16-e968-46a1-879a-3474aa7d4528],
                    [x-ms-webhdfs-version, 17.04.22.00], [Status, 0x0]...}
Images            : {}
InputFields       : {}
Links             : {}
ParsedHtml        : System.__ComObject
RawContentLength  : 556
```

6. Now you can try uploading a file to your Data Lake Store. First, create a file to upload.

```
echo "Test file." > Test1.txt
```

7. Using PowerShell's `Invoke-WebRequest`, make a request to your Data Lake Store's REST endpoint to upload the file to the folder you created earlier. This request takes two steps. In the first step, you make a request and get a redirection to where the file should be uploaded. In the second step, you actually upload the file. Remember to set the name of the folder and file appropriately if you used different values than in this tutorial.

```
$HdfsRedirectResponse = Invoke-WebRequest -Uri
https://<YOUR_ADLS_NAME>.azuredatalakestore.net/webhdfs/v1/TestFolder/Test1.txt?op=CREATE -Method PUT -
Headers @{Authorization="Bearer $AccessToken"} -Infile Test1.txt -MaximumRedirection 0
```

If you inspect the value of `$HdfsRedirectResponse` it should look like the following response:

```
PS C:\> $HdfsRedirectResponse

StatusCode        : 307
StatusDescription : Temporary Redirect
Content           : {}
RawContent        : HTTP/1.1 307 Temporary Redirect
                    Pragma: no-cache
                    x-ms-request-id: b7ab492f-b514-4483-aada-4aa0611d12b3
                    ContentLength: 0
                    x-ms-webhdfs-version: 17.04.22.00
                    Status: 0x0
                    X-Content-Type-Options: nosn...
Headers           : {[Pragma, no-cache], [x-ms-request-id, b7ab492f-b514-4483-aada-4aa0611d12b3],
                    [ContentLength, 0], [x-ms-webhdfs-version, 17.04.22.00]...}
RawContentLength  : 0
```

Complete the upload by sending a request to the redirect endpoint:

```
Invoke-WebRequest -Uri $HdfsRedirectResponse.Headers.Location -Method PUT -Headers
@{Authorization="Bearer $AccessToken"} -Infile Test1.txt -MaximumRedirection 0
```

A successful response look like:

```
StatusCode        : 201
StatusDescription : Created
Content           : {}
RawContent        : HTTP/1.1 201 Created
                    Pragma: no-cache
                    x-ms-request-id: 1e70f36f-ead1-4566-acfa-d0c3ec1e2307
                    ContentLength: 0
                    x-ms-webhdfs-version: 17.04.22.00
                    Status: 0x0
                    X-Content-Type-Options: nosniff
                    Strict...
Headers           : {[Pragma, no-cache], [x-ms-request-id, 1e70f36f-ead1-4566-acfa-d0c3ec1e2307],
                    [ContentLength, 0], [x-ms-webhdfs-version, 17.04.22.00]...}
RawContentLength  : 0
```

Using other Data Lake Store filesystem APIs you can append to files, download files, and more.

Next steps

In this tutorial, you learned how to use a system-assigned managed identity for a Windows virtual machine to access an Azure Data Lake Store. To learn more about Azure Data Lake Store see:

Azure Data Lake Store

Tutorial: Use a Windows VM system-assigned managed identity to access Azure Storage

4/1/2019 • 4 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Windows virtual machine (VM) to access Azure Storage. You learn how to:

- Create a blob container in a storage account
- Grant your Windows VM's system-assigned managed identity access to a storage account
- Get an access and use it to call Azure Storage

NOTE

Azure Active Directory authentication for Azure Storage is in public preview.

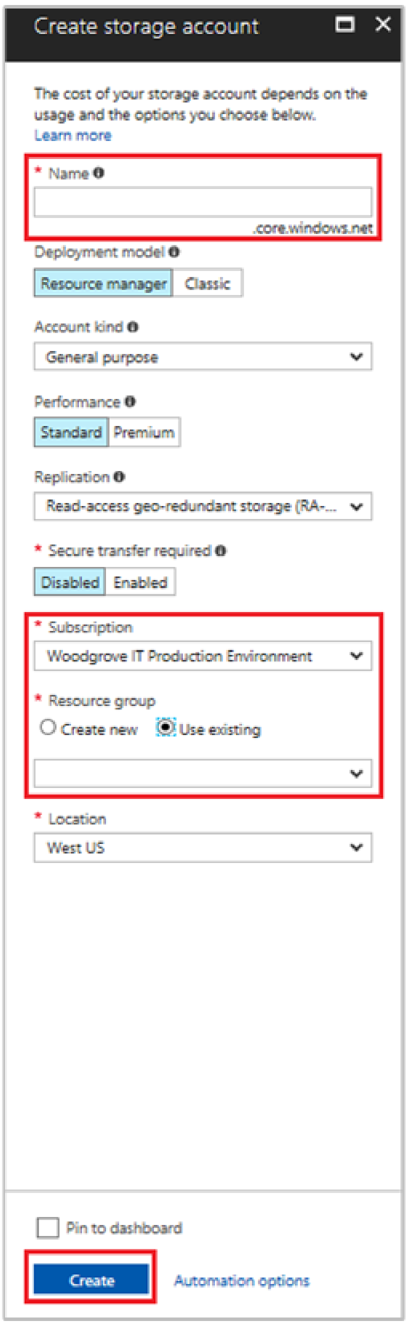
Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Create a storage account

In this section, you create a storage account.

1. Click the **+ Create a resource** button found on the upper left-hand corner of the Azure portal.
2. Click **Storage**, then **Storage account - blob, file, table, queue**.
3. Under **Name**, enter a name for the storage account.
4. **Deployment model** and **Account kind** should be set to **Resource manager** and **Storage (general purpose v1)**.
5. Ensure the **Subscription** and **Resource Group** match the ones you specified when you created your VM in the previous step.
6. Click **Create**.


Create storage account

The cost of your storage account depends on the usage and the options you choose below.
Learn more

* Name
.core.windows.net

Deployment model: [✓ Resource manager] Classic

Account kind: General purpose

Performance: [✓ Standard] Premium

Replication: Read-access geo-redundant storage (RA-...

* Secure transfer required: [✓ Disabled] Enabled

* Subscription: Woodgrove IT Production Environment

* Resource group: Create new [✓ Use existing]

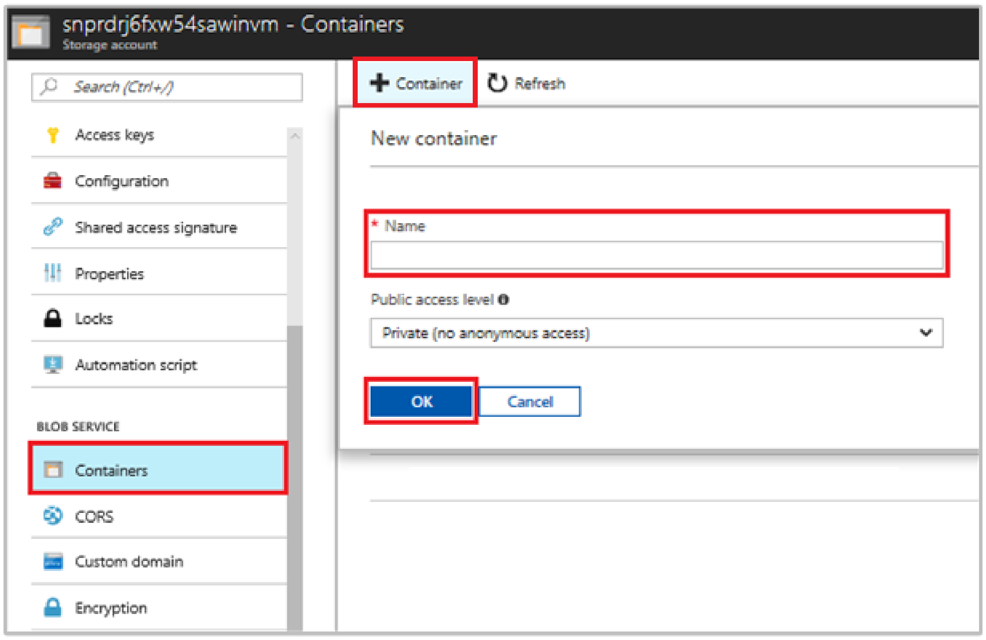
* Location: West US

Pin to dashboard

Create a blob container and upload a file to the storage account

Files require blob storage so you need to create a blob container in which to store the file. You then upload a file to the blob container in the new storage account.

1. Navigate back to your newly created storage account.
2. Under **Blob Service**, click **Containers**.
3. Click **+ Container** on the top of the page.
4. Under **New container**, enter a name for the container and under **Public access level** keep the default value .



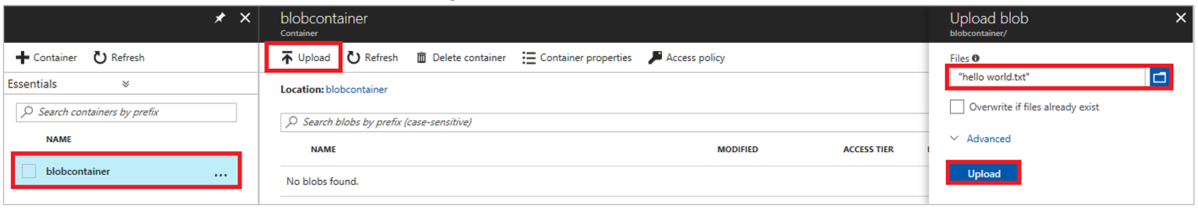
* Name

Public access level: Private (no anonymous access)

5. Using an editor of your choice, create a file titled *hello world.txt* on your local machine. Open the file and add the text (without the quotes) "Hello world! :)" and then save it.

6. Upload the file to the newly created container by clicking on the container name, then **Upload**

7. In the **Upload blob** pane, under **Files**, click the folder icon and browse to the file **hello_world.txt** on your local machine, select the file, then click **Upload**.



Location: blobcontainer

No blobs found.

Files: "hello world.txt"

Grant your VM access to an Azure Storage container

You can use the VM's system-assigned managed identity to retrieve the data in the Azure storage blob.

1. Navigate back to your newly created storage account.

2. Click the **Access control (IAM)** link in the left panel.

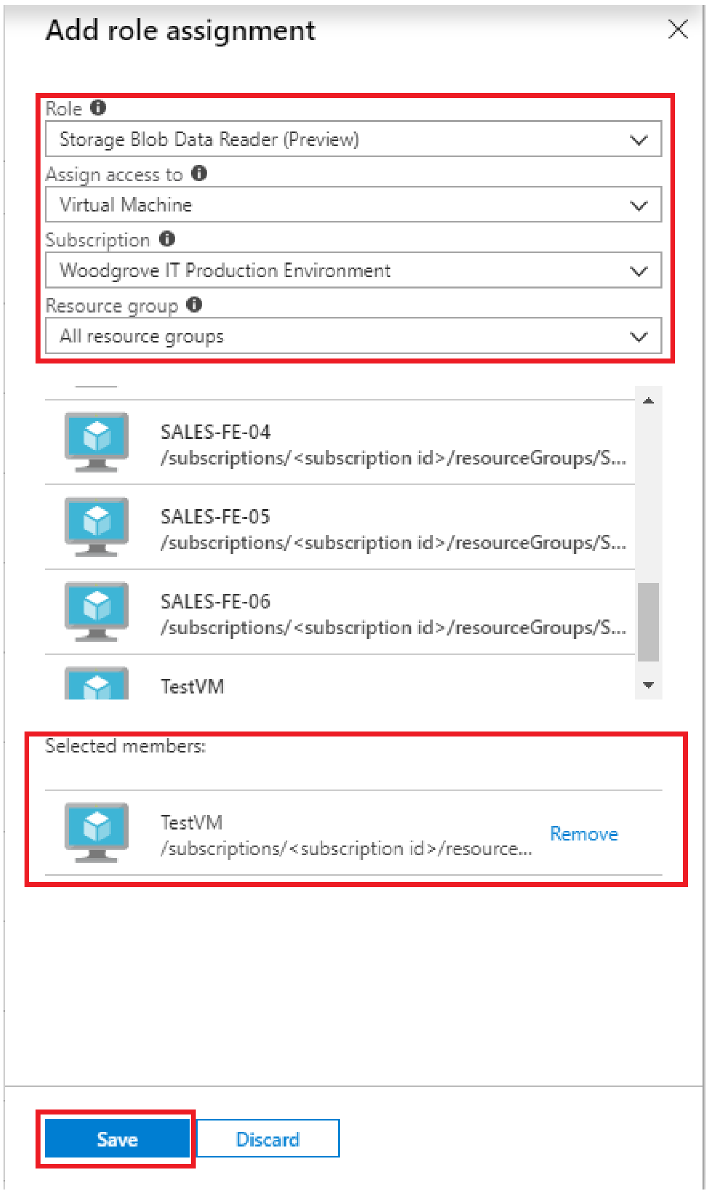
3. Click **+ Add role assignment** on top of the page to add a new role assignment for your VM.

4. Under **Role**, from the dropdown, select **Storage Blob Data Reader**.

5. In the next dropdown, under **Assign access to**, choose **Virtual Machine**.

6. Next, ensure the proper subscription is listed in **Subscription** dropdown and then set **Resource Group** to **All resource groups**.

7. Under **Select**, choose your VM and then click **Save**.


Add role assignment

Role: Storage Blob Data Reader (Preview)
Assign access to: Virtual Machine
Subscription: Woodgrove IT Production Environment
Resource group: All resource groups

SALES-FE-04
/subscriptions/<subscription id>/resourceGroups/S...
SALES-FE-05
/subscriptions/<subscription id>/resourceGroups/S...
SALES-FE-06
/subscriptions/<subscription id>/resourceGroups/S...
TestVM

Selected members:
TestVM
/subscriptions/<subscription id>/resource... Remove

Get an access token and use it to call Azure Storage

Azure Storage natively supports Azure AD authentication, so it can directly accept access tokens obtained using a managed identity. This is part of Azure Storage's integration with Azure AD, and is different from supplying credentials on the connection string.

Here's a .NET code example of opening a connection to Azure Storage using an access token and then reading the contents of the file you created earlier. This code must run on the VM to be able to access the VM's managed identity endpoint. .NET Framework 4.6 or higher is required to use the access token method. Replace the value of `<URI to blob file>` accordingly. You can obtain this value by navigating to file you created and uploaded to blob storage and copying the **URL** under **Properties** the **Overview** page.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
using System.Net;
using System.Web.Script.Serialization;
using Microsoft.WindowsAzure.Storage.Auth;
using Microsoft.WindowsAzure.Storage.Blob;

namespace StorageOAuthToken
{
    class Program
    {
        static void Main(string[] args)
        {
            //get token
            string accessToken = GetMSIToken("https://storage.azure.com/");

            //create token credential
            TokenCredential tokenCredential = new TokenCredential(accessToken);

            //create storage credentials
            StorageCredentials storageCredentials = new StorageCredentials(tokenCredential);

            Uri blobAddress = new Uri("<URI to blob file>");

            //create block blob using storage credentials
            CloudBlockBlob blob = new CloudBlockBlob(blobAddress, storageCredentials);

            //retrieve blob contents
            Console.WriteLine(blob.DownloadText());
            Console.ReadLine();
        }

        static string GetMSIToken(string resourceID)
        {
            string accessToken = string.Empty;
            // Build request to acquire MSI token
            HttpWebRequest request = 
(HttpWebRequest)WebRequest.Create("http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
01&resource=" + resourceID);
            request.Headers["Metadata"] = "true";
            request.Method = "GET";

            try
            {
                // Call /token endpoint
                HttpWebResponse response = (HttpWebResponse)request.GetResponse();

                // Pipe response Stream to a StreamReader, and extract access token
                StreamReader streamResponse = new StreamReader(response.GetResponseStream());
                string stringResponse = streamResponse.ReadToEnd();
                JavaScriptSerializer j = new JavaScriptSerializer();
                Dictionary<string, string> list = (Dictionary<string, string>)j.Deserialize(stringResponse, 
typeof(Dictionary<string, string>));
                accessToken = list["access_token"];
                return accessToken;
            }
            catch (Exception e)
            {
                string errorText = String.Format("{0} \n\n{1}", e.Message, e.InnerException != null ? 
e.InnerException.Message : "Acquire token failed");
                return accessToken;
            }
        }
    }
}
```

The response contains the contents of the file:

```
Hello world! :)
```

Next steps

In this tutorial, you learned how enable a Windows VM's system-assigned identity to access Azure Storage. To learn more about Azure Storage see:

Tutorial: Use a Windows VM system-assigned managed identity to access Azure Storage via access key

5/6/2019 • 5 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

IMPORTANT

Azure Storage now supports Azure AD authentication. As a best practice, use Azure AD authentication instead of access keys.

This tutorial shows you how to use a system-assigned managed identity for Windows virtual machine (VM) to retrieve storage account access keys. You can use storage access keys as usual when doing storage operations, for example when using the Storage SDK. For this tutorial, we upload and download blobs using Azure Storage PowerShell. You will learn how to:

- Create a storage account
- Grant your VM access to storage account access keys in Resource Manager
- Get an access token using your VM's identity, and use it to retrieve the storage access keys from Resource Manager

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

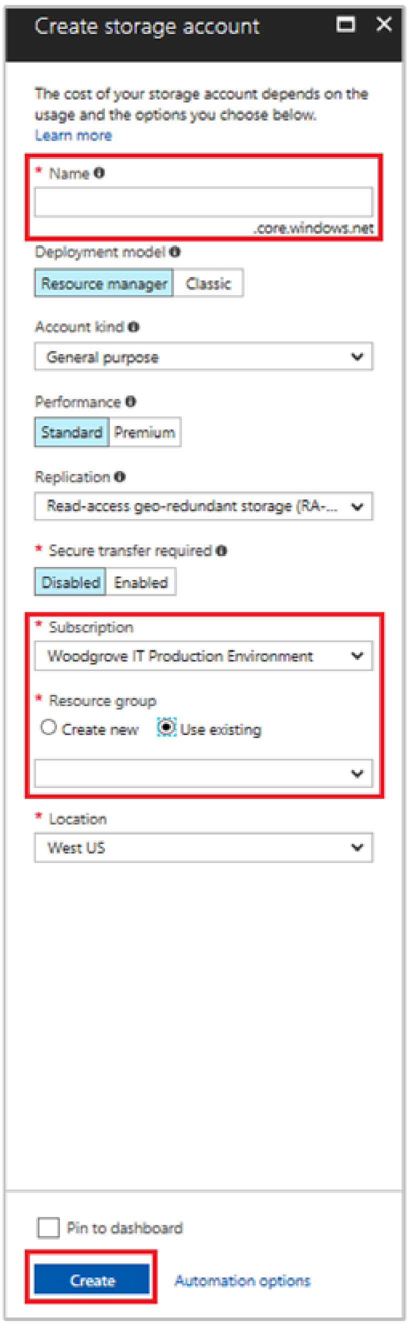
NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Create a storage account

If you don't already have one, you will now create a storage account. You can also skip this step and grant your VM's system-assigned managed identity access to the keys of an existing storage account.

1. Click the **+/Create new service** button found on the upper left-hand corner of the Azure portal.
2. Click **Storage**, then **Storage Account**, and a new "Create storage account" panel will display.
3. Enter a name for the storage account, which you will use later.
4. **Deployment model** and **Account kind** should be set to "Resource manager" and "General purpose", respectively.
5. Ensure the **Subscription** and **Resource Group** match the ones you specified when you created your VM in the previous step.
6. Click **Create**.



Create storage account

The cost of your storage account depends on the usage and the options you choose below.
Learn more

* Name: .core.windows.net

Deployment model: [✓ Resource manager] Classic

Account kind: General purpose

Performance: [✓ Standard] Premium

Replication: Read-access geo-redundant storage (RA-...

* Secure transfer required: [✓ Disabled] Enabled

* Subscription: Woodgrove IT Production Environment

* Resource group: Create new [✓ Use existing]

* Location: West US

Pin to dashboard

Create a blob container in the storage account

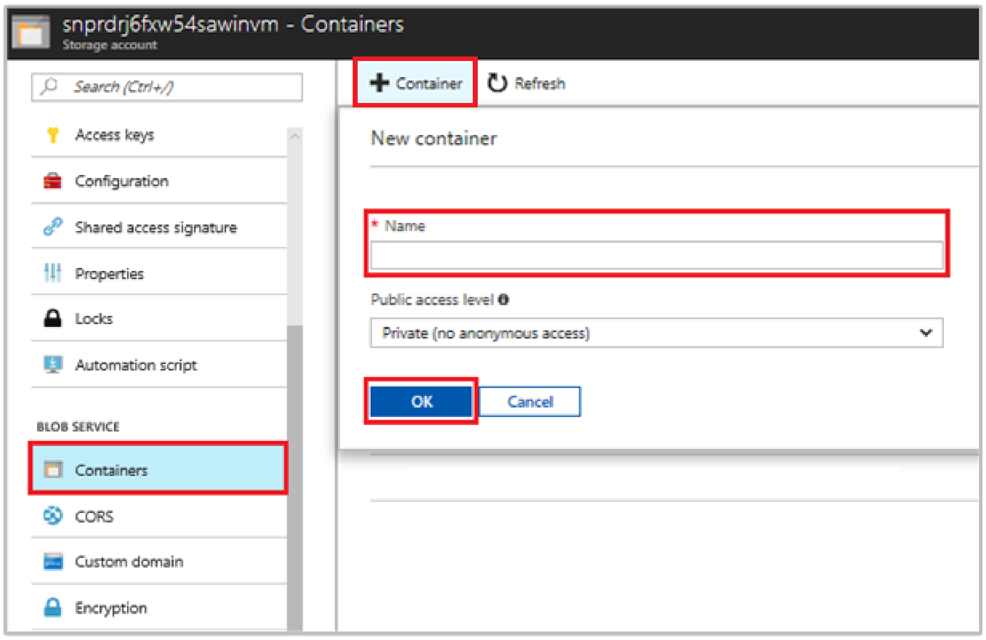
Later we will upload and download a file to the new storage account. Because files require blob storage, we need to create a blob container in which to store the file.

1. Navigate back to your newly created storage account.

2. Click the **Containers** link in the left, under "Blob service."

3. Click **+ Container** on the top of the page, and a "New container" panel slides out.

4. Give the container a name, select an access level, then click **OK**. The name you specified will be used later in the tutorial.



Grant your VM's system-assigned managed identity access to use storage account access keys

Azure Storage does not natively support Azure AD authentication. However, you can use your VM's system-assigned managed identity to retrieve storage account access keys from the Resource Manager, then use a key to access storage. In this step, you grant your VM's system-assigned managed identity access to the keys to your storage account.

1. Navigate back to your newly created storage account.

2. Click the **Access control (IAM)** link in the left panel.

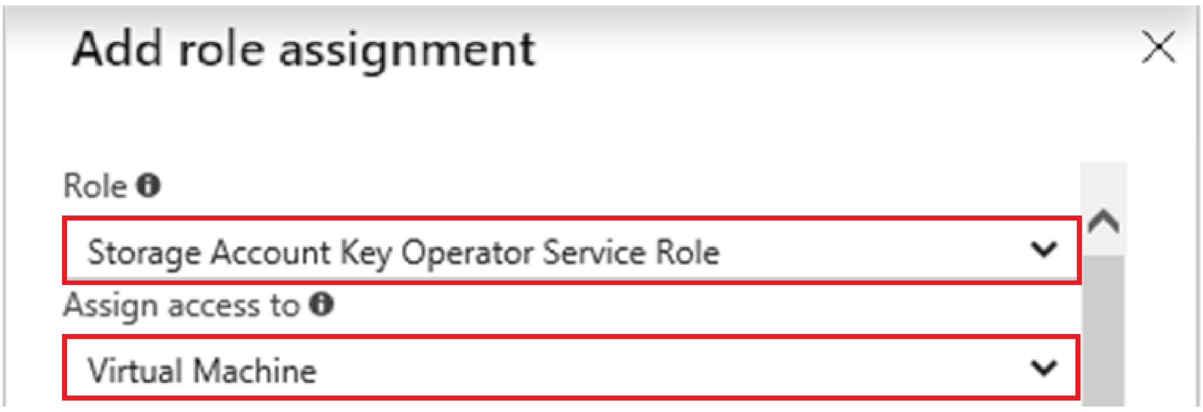
3. Click **+ Add role assignment** on top of the page to add a new role assignment for your VM

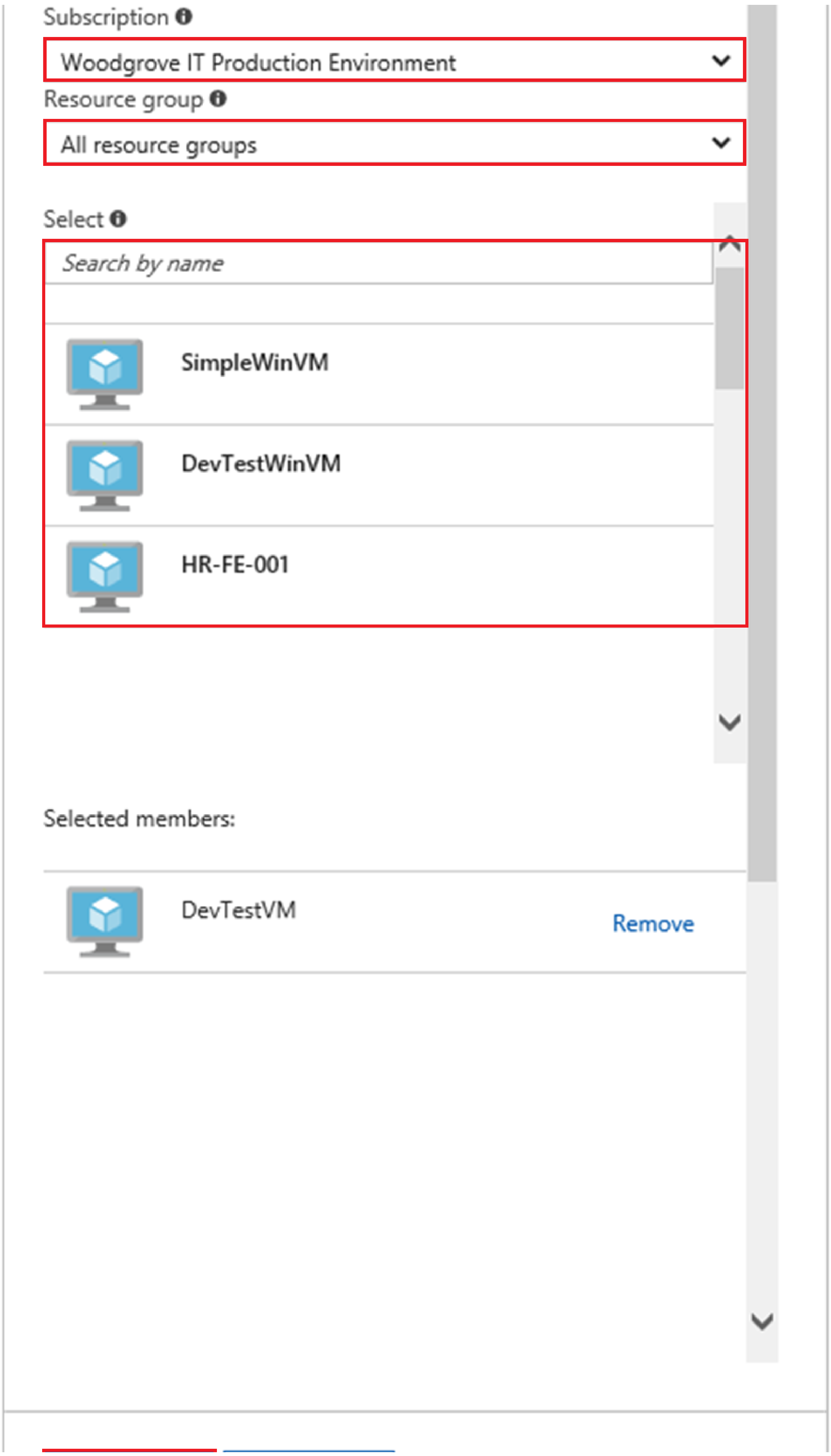
4. Set **Role** to "Storage Account Key Operator Service Role", on the right side of the page.

5. In the next dropdown, set **Assign access to** the resource "Virtual Machine".

6. Next, ensure the proper subscription is listed in **Subscription** dropdown, then set **Resource Group** to "All resource groups".

7. Finally, under **Select** choose your Windows Virtual Machine in the dropdown, then click **Save**.





Subscription

Woodgrove IT Production Environment

Resource group

All resource groups

Select

Search by name

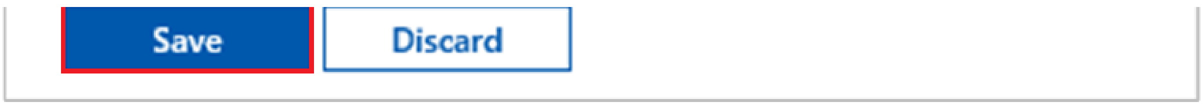
SimpleWinVM

DevTestWinVM

HR-FE-001

Selected members:

DevTestVM Remove



Get an access token using the VM's system-assigned managed identity to call Azure Resource Manager

For the remainder of the tutorial, we will work from the VM we created earlier.

You will need to use the Azure Resource Manager PowerShell cmdlets in this portion. If you don’t have it installed, download the latest version before continuing.

1. In the Azure portal, navigate to **Virtual Machines**, go to your Windows virtual machine, then from the **Overview** page click **Connect** at the top.
2. Enter in your **Username** and **Password** for which you added when you created the Windows VM.
3. Now that you have created a **Remote Desktop Connection** with the virtual machine, open PowerShell in the remote session.
4. Using Powershell’s Invoke-WebRequest, make a request to the local managed identity for Azure resources endpoint to get an access token for Azure Resource Manager.

```
   $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -Method GET -Headers @{Metadata="true"}
```

> **NOTE**
> The value of the "resource" parameter must be an exact match for what is expected by Azure AD. When using the Azure Resource Manager resource ID, you must include the trailing slash on the URI.

Next, extract the "Content" element, which is stored as a JavaScript Object Notation (JSON) formatted string in the $response object.

```
$content = $response.Content | ConvertFrom-Json
```

Next, extract the access token from the response.

```
$ArmToken = $content.access_token
```

Get storage account access keys from Azure Resource Manager to make storage calls

Now use PowerShell to call Resource Manager using the access token we retrieved in the previous section, to retrieve the storage access key. Once we have the storage access key, we can call storage upload/download operations.

```
$keysResponse = Invoke-WebRequest -Uri https://management.azure.com/subscriptions/<SUBSCRIPTION-ID>/resourceGroups/<RESOURCE-GROUP>/providers/Microsoft.Storage/storageAccounts/<STORAGE-ACCOUNT>/listKeys/?api-version=2016-12-01 -Method POST -Headers @{Authorization="Bearer $ARMToken"}
```

NOTE

The URL is case-sensitive, so ensure you use the exact same case used earlier, when you named the Resource Group, including the uppercase "G" in "resourceGroups."

```
$keysContent = $keysResponse.Content | ConvertFrom-Json
$key = $keysContent.keys[0].value
```

Next we create a file called "test.txt". Then use the storage access key to authenticate with the `New-AzStorageContent` cmdlet, upload the file to our blob container, then download the file.

```
echo "This is a test text file." > test.txt
```

Be sure to install the Azure Storage cmdlets first, using `Install-Module Az.Storage`. Then upload the blob you just created, using the `Set-AzStorageBlobContent` PowerShell cmdlet:

```
$ctx = New-AzStorageContext -StorageAccountName <STORAGE-ACCOUNT> -StorageAccountKey $key
Set-AzStorageBlobContent -File test.txt -Container <CONTAINER-NAME> -Blob testblob -Context $ctx
```

Response:

```
ICloudBlob        : Microsoft.WindowsAzure.Storage.Blob.CloudBlockBlob
BlobType          : BlockBlob
Length            : 56
ContentType       : application/octet-stream
LastModified      : 9/13/2017 6:14:25 PM +00:00
SnapshotTime      :
ContinuationToken :
Context           : Microsoft.WindowsAzure.Commands.Storage.AzureStorageContext
Name              : testblob
```

You can also download the blob you just uploaded, using the `Get-AzStorageBlobContent` PowerShell cmdlet:

```
Get-AzStorageBlobContent -Blob testblob -Container <CONTAINER-NAME> -Destination test2.txt -Context $ctx
```

Response:

```
ICloudBlob        : Microsoft.WindowsAzure.Storage.Blob.CloudBlockBlob
BlobType          : BlockBlob
Length            : 56
ContentType       : application/octet-stream
LastModified      : 9/13/2017 6:14:25 PM +00:00
SnapshotTime      :
ContinuationToken :
Context           : Microsoft.WindowsAzure.Commands.Storage.AzureStorageContext
Name              : testblob
```

Next steps

In this tutorial, you learned how to create a system-assigned managed identity to access Azure Storage using an access key. To learn more about Azure Storage access keys see:

Manage your storage access keys

Tutorial: Use a Windows VM system-assigned managed identity to access Azure Storage via a SAS credential

3/26/2019 • 6 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned identity for a Windows virtual machine (VM) to obtain a storage Shared Access Signature (SAS) credential. Specifically, a Service SAS credential.

A Service SAS provides the ability to grant limited access to objects in a storage account, for limited time and a specific service (in our case, the blob service), without exposing an account access key. You can use a SAS credential as usual when doing storage operations, for example when using the Storage SDK. For this tutorial, we demonstrate uploading and downloading a blob using Azure Storage PowerShell. You will learn how to:

- Create a storage account
- Grant your VM access to a storage account SAS in Resource Manager
- Get an access token using your VM's identity, and use it to retrieve the SAS from Resource Manager

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Create a storage account

If you don't already have one, you will now create a storage account. You can also skip this step and grant your VM's system-assigned managed identity access to the SAS credential of an existing storage account.

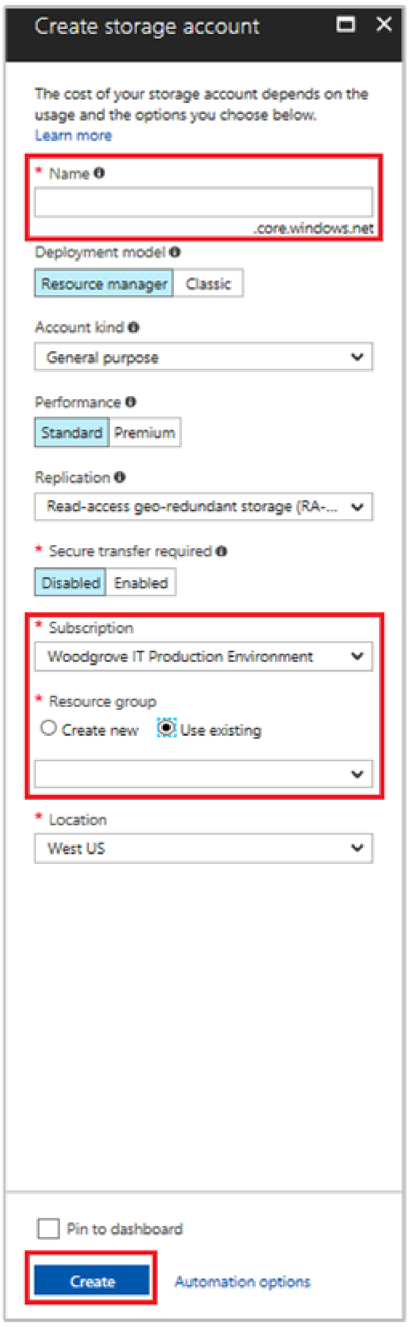
1. Click the **+/Create new service** button found on the upper left-hand corner of the Azure portal.
2. Click **Storage**, then **Storage Account**, and a new "Create storage account" panel will display.

3. Enter a name for the storage account, which you will use later.

4. **Deployment model** and **Account kind** should be set to "Resource manager" and "General purpose", respectively.

5. Ensure the **Subscription** and **Resource Group** match the ones you specified when you created your VM in the previous step.

6. Click **Create**.


Create storage account

The cost of your storage account depends on the usage and the options you choose below.
Learn more

* Name
.core.windows.net

Deployment model: [✓ Resource manager] Classic

Account kind: General purpose

Performance: [✓ Standard] Premium

Replication: Read-access geo-redundant storage (RA-...

* Secure transfer required: [✓ Disabled] Enabled

* Subscription: Woodgrove IT Production Environment

* Resource group: Create new [✓ Use existing]

* Location: West US

Pin to dashboard

Create a blob container in the storage account

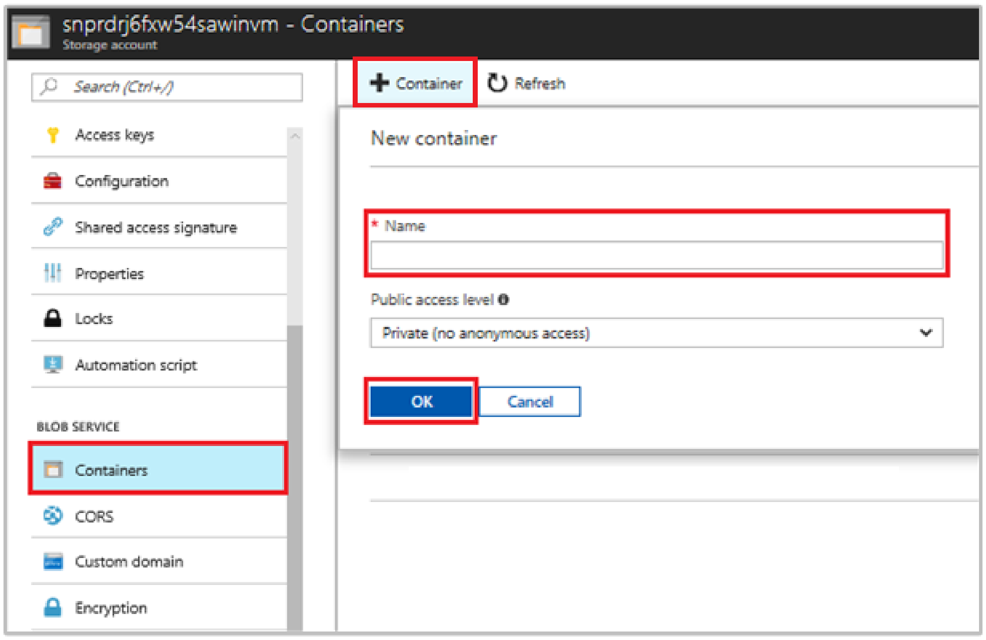
Later we will upload and download a file to the new storage account. Because files require blob storage, we need to create a blob container in which to store the file.

1. Navigate back to your newly created storage account.

2. Click the **Containers** link in the left panel, under "Blob service."

3. Click **+ Container** on the top of the page, and a "New container" panel slides out.

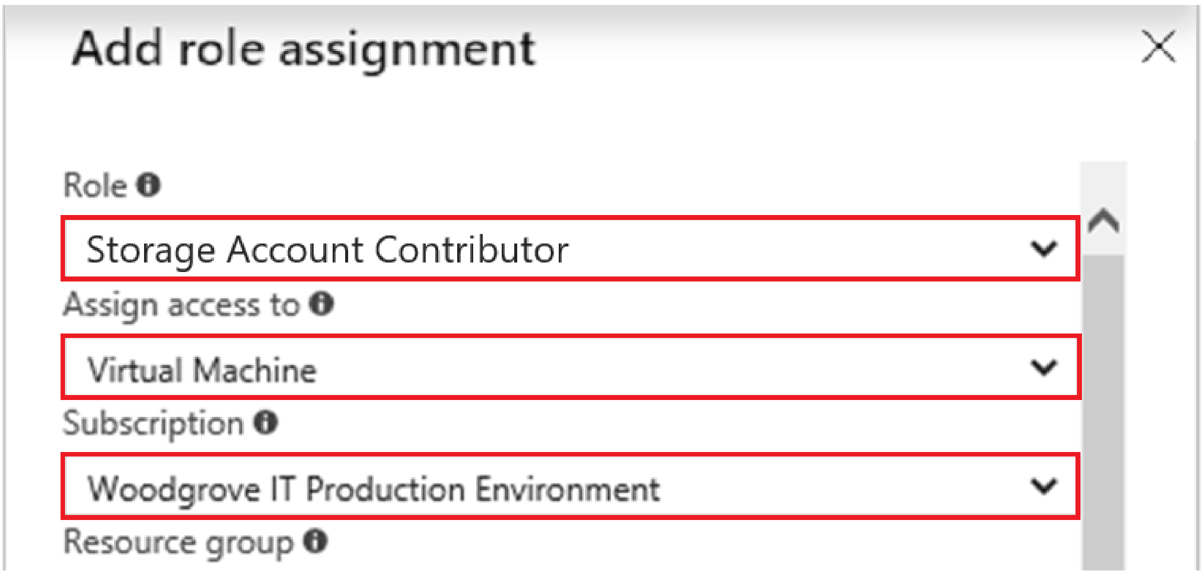
4. Give the container a name, select an access level, then click **OK**. The name you specified will be used later in the tutorial.



Grant your VM's system-assigned managed identity access to use a storage SAS

Azure Storage does not natively support Azure AD authentication. However, you can use a managed identity to retrieve a storage SAS from Resource Manager, then use the SAS to access storage. In this step, you grant your VM's system-assigned managed identity access to your storage account SAS.

1. Navigate back to your newly created storage account.
2. Click the **Access control (IAM)** link in the left panel.
3. Click **+ Add role assignment** on top of the page to add a new role assignment for your VM
4. Set **Role** to "Storage Account Contributor", on the right side of the page.
5. In the next dropdown, set **Assign access to** the resource "Virtual Machine".
6. Next, ensure the proper subscription is listed in **Subscription** dropdown, then set **Resource Group** to "All resource groups".
7. Finally, under **Select** choose your Windows Virtual Machine in the dropdown, then click **Save**.

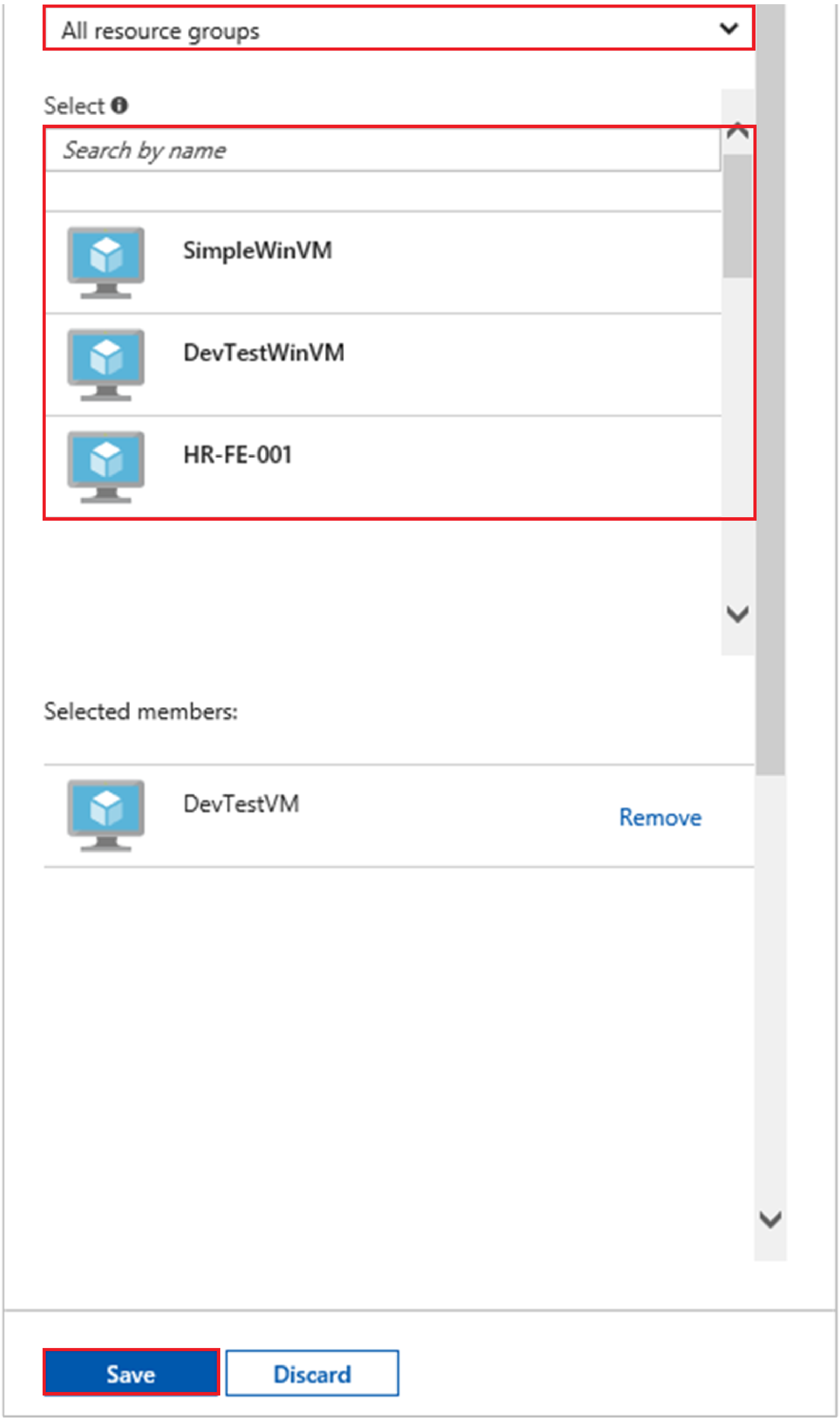

Add role assignment

Role: Storage Account Contributor

Assign access to: Virtual Machine

Subscription: Woodgrove IT Production Environment

Resource group


All resource groups

Select

Search by name

SimpleWinVM

DevTestWinVM

HR-FE-001

Selected members:

DevTestVM

Get an access token using the VM's identity and use it to call Azure Resource Manager

For the remainder of the tutorial, we will work from the VM we created earlier.

You will need to use the Azure Resource Manager PowerShell cmdlets in this portion. If you don't have it installed, download the latest version before continuing.

1. In the Azure portal, navigate to **Virtual Machines**, go to your Windows virtual machine, then from the **Overview** page click **Connect** at the top.
2. Enter in your **Username** and **Password** for which you added when you created the Windows VM.
3. Now that you have created a **Remote Desktop Connection** with the virtual machine, open PowerShell in the remote session.
4. Using Powershell's Invoke-WebRequest, make a request to the local managed identity for Azure resources endpoint to get an access token for Azure Resource Manager.

   ```
      $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -Method GET -Headers @{Metadata="true"}
   ```

   NOTE

   The value of the "resource" parameter must be an exact match for what is expected by Azure AD. When using the Azure Resource Manager resource ID, you must include the trailing slash on the URI.

   Next, extract the "Content" element, which is stored as a JavaScript Object Notation (JSON) formatted string in the $response object.

   ```
   $content = $response.Content | ConvertFrom-Json
   ```

   Next, extract the access token from the response.

   ```
   $ArmToken = $content.access_token
   ```

Get a SAS credential from Azure Resource Manager to make storage calls

Now use PowerShell to call Resource Manager using the access token we retrieved in the previous section, to create a storage SAS credential. Once we have the SAS credential, we can call storage operations.

For this request we'll use the follow HTTP request parameters to create the SAS credential:

```
{
    "canonicalizedResource":"/blob/<STORAGE ACCOUNT NAME>/<CONTAINER NAME>",
    "signedResource":"c",              // The kind of resource accessible with the SAS, in this case a
container (c).
    "signedPermission":"rcw",          // Permissions for this SAS, in this case (r)ead, (c)reate, and
(w)rite. Order is important.
    "signedProtocol":"https",          // Require the SAS be used on https protocol.
    "signedExpiry":"<EXPIRATION TIME>" // UTC expiration time for SAS in ISO 8601 format, for example 2017-09-
22T00:06:00Z.
}
```

These parameters are included in the POST body of the request for the SAS credential. For more information on the parameters for creating a SAS credential, see the List Service SAS REST reference.

First, convert the parameters to JSON, then call the storage `listServiceSas` endpoint to create the SAS credential:

```
$params = @{canonicalizedResource="/blob/<STORAGE-ACCOUNT-NAME>/<CONTAINER-
NAME>";signedResource="c";signedPermission="rcw";signedProtocol="https";signedExpiry="2017-09-23T00:00:00Z"}
$jsonParams = $params | ConvertTo-Json
```

```
$sasResponse = Invoke-WebRequest -Uri https://management.azure.com/subscriptions/<SUBSCRIPTION-
ID>/resourceGroups/<RESOURCE-GROUP>/providers/Microsoft.Storage/storageAccounts/<STORAGE-ACCOUNT-
NAME>/listServiceSas/?api-version=2017-06-01 -Method POST -Body $jsonParams -Headers @{Authorization="Bearer
$ArmToken"}
```

NOTE

The URL is case-sensitive, so ensure you use the exact same case used earlier, when you named the Resource Group, including the uppercase "G" in "resourceGroups."

Now we can extract the SAS credential from the response:

```
$sasContent = $sasResponse.Content | ConvertFrom-Json
$sasCred = $sasContent.serviceSasToken
```

If you inspect the SAS cred you'll see something like this:

```
PS C:\> $sasCred
sv=2015-04-05&sr=c&spr=https&se=2017-09-
23T00%3A00%3A00Z&sp=rcw&sig=JVhIWG48nmxqhTIuN0uiFBppdzhwHdehdYan1W%2F4O0E%3D
```

Next we create a file called "test.txt". Then use the SAS credential to authenticate with the `New-AzStorageContent` cmdlet, upload the file to our blob container, then download the file.

```
echo "This is a test text file." > test.txt
```

Be sure to install the Azure Storage cmdlets first, using `Install-Module Azure.Storage`. Then upload the blob you just created, using the `Set-AzStorageBlobContent` PowerShell cmdlet:

```
$ctx = New-AzStorageContext -StorageAccountName <STORAGE-ACCOUNT-NAME> -SasToken $sasCred
Set-AzStorageBlobContent -File test.txt -Container <CONTAINER-NAME> -Blob testblob -Context $ctx
```

Response:

```
ICloudBlob        : Microsoft.WindowsAzure.Storage.Blob.CloudBlockBlob
BlobType          : BlockBlob
Length            : 56
ContentType       : application/octet-stream
LastModified      : 9/21/2017 6:14:25 PM +00:00
SnapshotTime      :
ContinuationToken :
Context           : Microsoft.WindowsAzure.Commands.Storage.AzureStorageContext
Name              : testblob
```

You can also download the blob you just uploaded, using the `Get-AzStorageBlobContent` PowerShell cmdlet:

```
Get-AzStorageBlobContent -Blob testblob -Container <CONTAINER-NAME> -Destination test2.txt -Context $ctx
```

Response:

```
ICloudBlob        : Microsoft.WindowsAzure.Storage.Blob.CloudBlockBlob
BlobType          : BlockBlob
Length            : 56
ContentType       : application/octet-stream
LastModified      : 9/21/2017 6:14:25 PM +00:00
SnapshotTime      :
ContinuationToken :
Context           : Microsoft.WindowsAzure.Commands.Storage.AzureStorageContext
Name              : testblob
```

Next steps

In this tutorial, you learned how to use a Windows VM's system-assigned managed identity to access Azure Storage using a SAS credential. To learn more about Azure Storage SAS see:

Using shared access signatures (SAS)

Tutorial: Use a Windows VM system-assigned managed identity to access Azure SQL

3/26/2019 • 6 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned identity for a Windows virtual machine (VM) to access an Azure SQL server. Managed Service Identities are automatically managed by Azure and enable you to authenticate to services that support Azure AD authentication, without needing to insert credentials into your code. You learn how to:

- Grant your VM access to an Azure SQL server
- Enable Azure AD authentication for the SQL server
- Create a contained user in the database that represents the VM's system assigned identity
- Get an access token using the VM identity and use it to query an Azure SQL server

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Grant your VM access to a database in an Azure SQL server

To grant your VM access to a database in an Azure SQL Server, you can use an existing SQL server or create a new one. To create a new server and database using the Azure portal, follow this Azure SQL quickstart. There are also quickstarts that use the Azure CLI and Azure PowerShell in the Azure SQL documentation.

There are two steps to granting your VM access to a database:

1. Enable Azure AD authentication for the SQL server.
2. Create a **contained user** in the database that represents the VM's system-assigned identity.

Enable Azure AD authentication for the SQL server

Configure Azure AD authentication for the SQL server using the following steps:

1. In the Azure portal, select **SQL servers** from the left-hand navigation.
2. Click the SQL server to be enabled for Azure AD authentication.
3. In the **Settings** section of the blade, click **Active Directory admin**.

4. In the command bar, click **Set admin**.
5. Select an Azure AD user account to be made an administrator of the server, and click **Select.**
6. In the command bar, click **Save.**

Create a contained user in the database that represents the VM's system assigned identity

For this next step, you will need Microsoft SQL Server Management Studio (SSMS). Before beginning, it may also be helpful to review the following articles for background on Azure AD integration:

- Universal Authentication with SQL Database and SQL Data Warehouse (SSMS support for MFA)
- Configure and manage Azure Active Directory authentication with SQL Database or SQL Data Warehouse

SQL DB requires unique AAD display names. With this, the AAD accounts such as users, groups and Service Principals (applications) and VM names enabled for managed identity must be uniquely defined in AAD regarding their display names. SQL DB checks the AAD display name during T-SQL creation of such users and if it is not unique, the command fails requesting to provide a unique AAD display name for a given account.

1. Start SQL Server Management Studio.
2. In the **Connect to Server** dialog, Enter your SQL server name in the **Server name** field.
3. In the **Authentication** field, select **Active Directory - Universal with MFA support**.
4. In the **User name** field, enter the name of the Azure AD account that you set as the server administrator, for example, helen@woodgroveonline.com
5. Click **Options**.
6. In the **Connect to database** field, enter the name of the non-system database you want to configure.
7. Click **Connect**. Complete the sign-in process.
8. In the **Object Explorer**, expand the **Databases** folder.
9. Right-click on a user database and click **New query**.
10. In the query window, enter the following line, and click **Execute** in the toolbar:

    NOTE
    `VMName` in the following command is the name of the VM that you enabled system assigned identity on in the prerequsites section.

    ```
    CREATE USER [VMName] FROM EXTERNAL PROVIDER
    ```

    The command should complete successfully, creating the contained user for the VM's system-assigned identity.

11. Clear the query window, enter the following line, and click **Execute** in the toolbar:

    NOTE
    `VMName` in the following command is the name of the VM that you enabled system assigned identity on in the prerequsites section.

```
ALTER ROLE db_datareader ADD MEMBER [VMName]
```

The command should complete successfully, granting the contained user the ability to read the entire database.

Code running in the VM can now get a token using its system-assigned managed identity and use the token to authenticate to the SQL server.

Get an access token using the VM's system-assigned managed identity and use it to call Azure SQL

Azure SQL natively supports Azure AD authentication, so it can directly accept access tokens obtained using managed identities for Azure resources. You use the **access token** method of creating a connection to SQL. This is part of Azure SQL's integration with Azure AD, and is different from supplying credentials on the connection string.

Here's a .NET code example of opening a connection to SQL using an access token. This code must run on the VM to be able to access the VM's system-assigned managed identity's endpoint. **.NET Framework 4.6** or higher is required to use the access token method. Replace the values of AZURE-SQL-SERVERNAME and DATABASE accordingly. Note the resource ID for Azure SQL is `https://database.windows.net/`.

```
using System.Net;
using System.IO;
using System.Data.SqlClient;
using System.Web.Script.Serialization;

//
// Get an access token for SQL.
//
HttpWebRequest request =
(HttpWebRequest)WebRequest.Create("http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
01&resource=https://database.windows.net/");
request.Headers["Metadata"] = "true";
request.Method = "GET";
string accessToken = null;

try
{
    // Call managed identities for Azure resources endpoint.
    HttpWebResponse response = (HttpWebResponse)request.GetResponse();

    // Pipe response Stream to a StreamReader and extract access token.
    StreamReader streamResponse = new StreamReader(response.GetResponseStream());
    string stringResponse = streamResponse.ReadToEnd();
    JavaScriptSerializer j = new JavaScriptSerializer();
    Dictionary<string, string> list = (Dictionary<string, string>) j.Deserialize(stringResponse,
typeof(Dictionary<string, string>));
    accessToken = list["access_token"];
}
catch (Exception e)
{
    string errorText = String.Format("{0} \n\n{1}", e.Message, e.InnerException != null ?
e.InnerException.Message : "Acquire token failed");
}

//
// Open a connection to the SQL server using the access token.
//
if (accessToken != null) {
    string connectionString = "Data Source=<AZURE-SQL-SERVERNAME>; Initial Catalog=<DATABASE>;";
    SqlConnection conn = new SqlConnection(connectionString);
    conn.AccessToken = accessToken;
    conn.Open();
}
```

Alternatively, a quick way to test the end to end setup without having to write and deploy an app on the VM is using PowerShell.

1. In the portal, navigate to **Virtual Machines** and go to your Windows virtual machine and in the **Overview**, click **Connect**.

2. Enter in your **Username** and **Password** for which you added when you created the Windows VM.

3. Now that you have created a **Remote Desktop Connection** with the virtual machine, open **PowerShell** in the remote session.

4. Using PowerShell's `Invoke-WebRequest`, make a request to the local managed identity's endpoint to get an access token for Azure SQL.

   ```
       $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-
   version=2018-02-01&resource=https%3A%2F%2Fdatabase.windows.net%2F' -Method GET -Headers
   @{Metadata="true"}
   ```

   Convert the response from a JSON object to a PowerShell object.

```
$content = $response.Content | ConvertFrom-Json
```

Extract the access token from the response.

```
$AccessToken = $content.access_token
```

5. Open a connection to the SQL server. Remember to replace the values for AZURE-SQL-SERVERNAME and DATABASE.

```
$SqlConnection = New-Object System.Data.SqlClient.SqlConnection
$SqlConnection.ConnectionString = "Data Source = <AZURE-SQL-SERVERNAME>; Initial Catalog = <DATABASE>"
$SqlConnection.AccessToken = $AccessToken
$SqlConnection.Open()
```

Next, create and send a query to the server. Remember to replace the value for TABLE.

```
$SqlCmd = New-Object System.Data.SqlClient.SqlCommand
$SqlCmd.CommandText = "SELECT * from <TABLE>;"
$SqlCmd.Connection = $SqlConnection
$SqlAdapter = New-Object System.Data.SqlClient.SqlDataAdapter
$SqlAdapter.SelectCommand = $SqlCmd
$DataSet = New-Object System.Data.DataSet
$SqlAdapter.Fill($DataSet)
```

Examine the value of `$DataSet.Tables[0]` to view the results of the query.

Next steps

In this tutorial, you learned how to use a system-assigned managed identity to access Azure SQL server. To learn more about Azure SQL Server see:

Azure SQL Database service

Tutorial: Use a Windows VM system-assigned managed identity to access Azure Key Vault

3/26/2019 • 4 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Windows virtual machine (VM) to access Azure Key Vault. Serving as a bootstrap, Key Vault makes it possible for your client application to then use the secret to access resources not secured by Azure Active Directory (AD). Managed Service Identities are automatically managed by Azure and enable you to authenticate to services that support Azure AD authentication, without needing to insert credentials into your code.

You learn how to:

- Grant your VM access to a secret stored in a Key Vault
- Get an access token using the VM identity and use it to retrieve the secret from Key Vault

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Grant your VM access to a Secret stored in a Key Vault

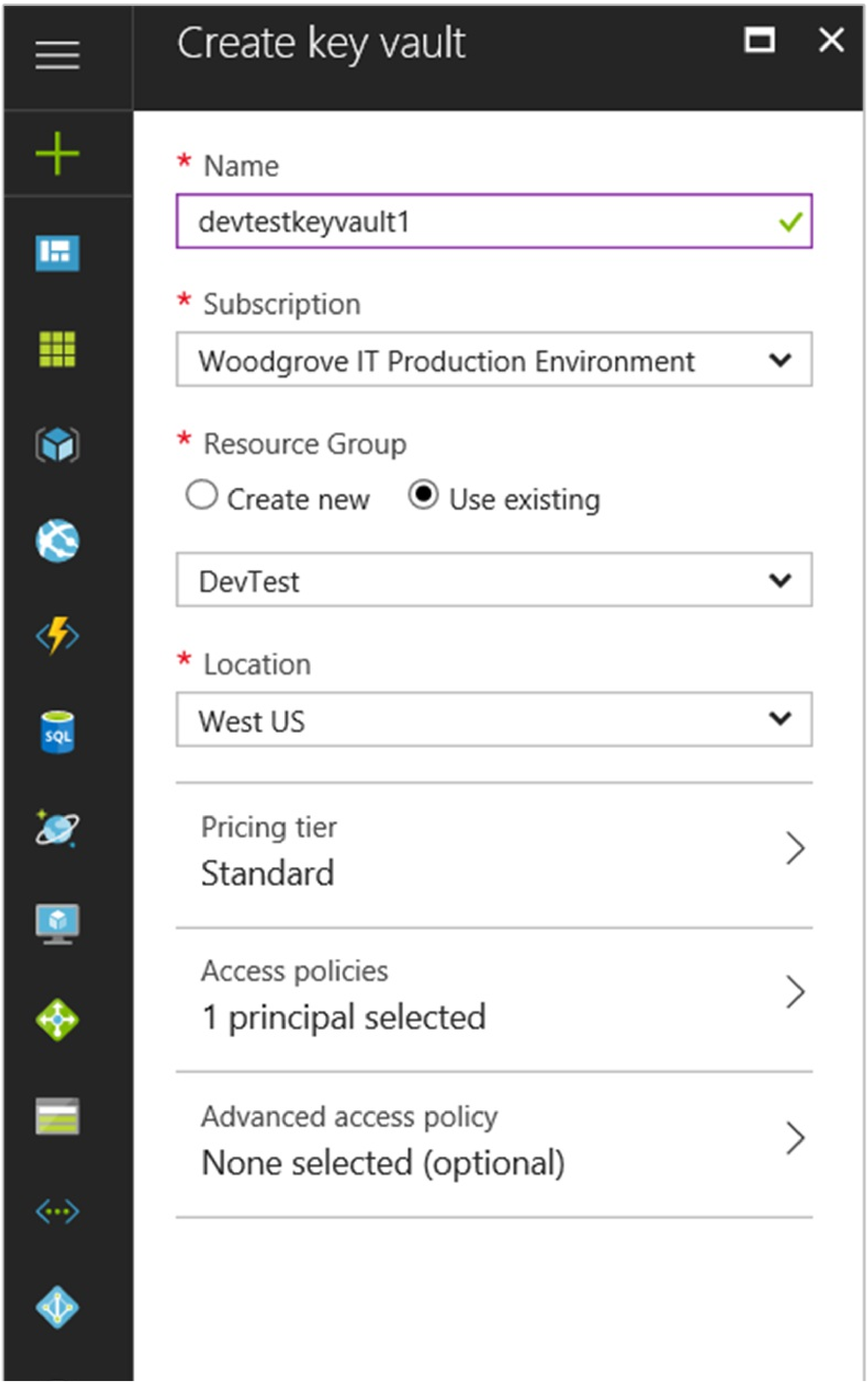
Using managed identities for Azure resources, your code can get access tokens to authenticate to resources that support Azure AD authentication. However, not all Azure services support Azure AD authentication. To use managed identities for Azure resources with those services, store the service credentials in Azure Key Vault, and use the VM's managed identity to access Key Vault to retrieve the credentials.

First, we need to create a Key Vault and grant our VM's system-assigned managed identity access to the Key Vault.

1. At the top of the left navigation bar, select **Create a resource** > **Security + Identity** > **Key Vault**.
2. Provide a **Name** for the new Key Vault.
3. Locate the Key Vault in the same subscription and resource group as the VM you created earlier.
4. Select **Access policies** and click **Add new**.
5. In Configure from template, select **Secret Management**.
6. Choose **Select Principal**, and in the search field enter the name of the VM you created earlier. Select the

VM in the result list and click **Select**.

7. Click **OK** to finishing adding the new access policy, and **OK** to finish access policy selection.
8. Click **Create** to finish creating the Key Vault.



Create key vault

* Name

devtestkeyvault1

* Subscription

Woodgrove IT Production Environment

* Resource Group

Create new [✓ Use existing]

DevTest

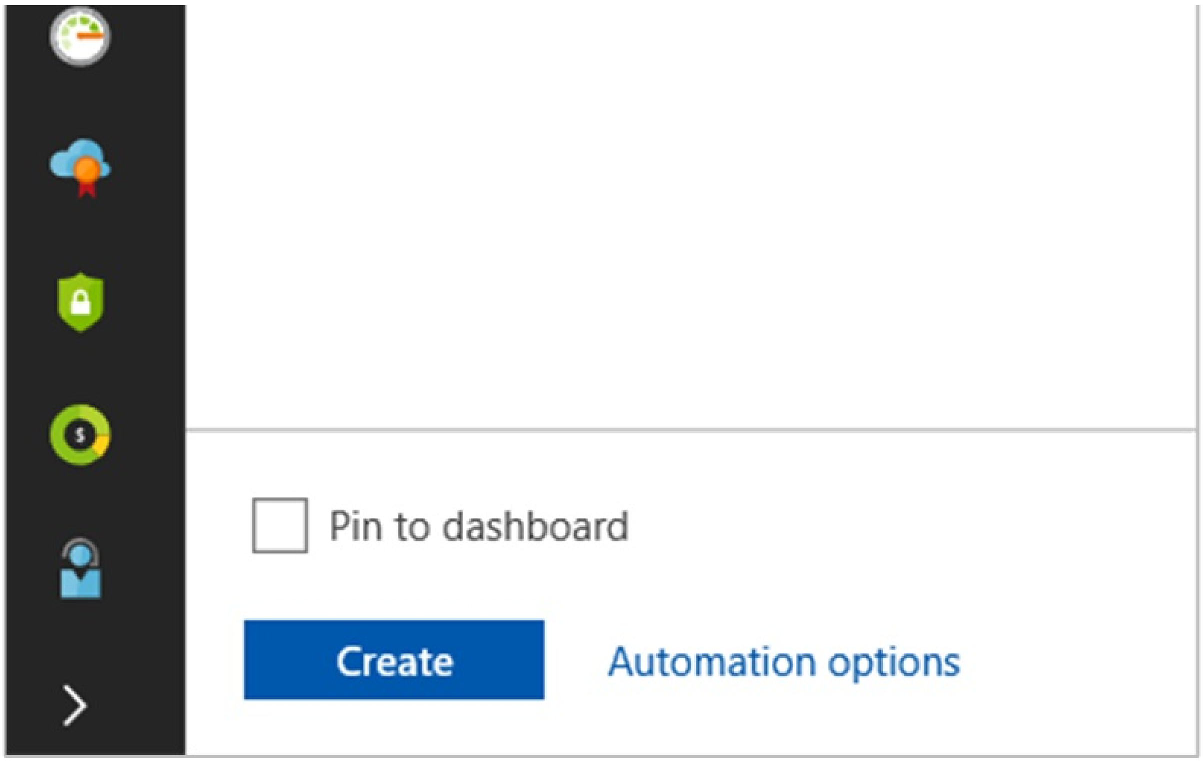
* Location

West US

Pricing tier
Standard

Access policies
1 principal selected

Advanced access policy
None selected (optional)



Next, add a secret to the Key Vault, so that later you can retrieve the secret using code running in your VM:

1. Select **All Resources**, and find and select the Key Vault you created.
2. Select **Secrets**, and click **Add**.
3. Select **Manual**, from **Upload options**.
4. Enter a name and value for the secret. The value can be anything you want.
5. Leave the activation date and expiration date clear, and leave **Enabled** as **Yes**.
6. Click **Create** to create the secret.

Get an access token using the VM identity and use it to retrieve the secret from the Key Vault

If you don’t have PowerShell 4.3.1 or greater installed, you'll need to download and install the latest version.

First, we use the VM’s system-assigned managed identity to get an access token to authenticate to Key Vault:

1. In the portal, navigate to **Virtual Machines** and go to your Windows virtual machine and in the **Overview**, click **Connect**.
2. Enter in your **Username** and **Password** for which you added when you created the **Windows VM**.
3. Now that you have created a **Remote Desktop Connection** with the virtual machine, open PowerShell in the remote session.
4. In PowerShell, invoke the web request on the tenant to get the token for the local host in the specific port for the VM.

   The PowerShell request:

   ```
   $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-
   version=2018-02-01&resource=https%3A%2F%2Fvault.azure.net' -Method GET -Headers @{Metadata="true"}
   ```

   Next, extract the full response which is stored as a JavaScript Object Notation (JSON) formatted string in the $response object.

```
$content = $response.Content | ConvertFrom-Json
```

Next, extract the access token from the response.

```
$KeyVaultToken = $content.access_token
```

Finally, use PowerShell's Invoke-WebRequest command to retrieve the secret you created earlier in the Key Vault, passing the access token in the Authorization header. You'll need the URL of your Key Vault, which is in the **Essentials** section of the **Overview** page of the Key Vault.

```
(Invoke-WebRequest -Uri https://<your-key-vault-URL>/secrets/<secret-name>?api-version=2016-10-01 -
Method GET -Headers @{Authorization="Bearer $KeyVaultToken"}).content
```

The response will look like this:

```
{"value":"p@ssw0rd!","id":"https://mytestkeyvault.vault.azure.net/secrets/MyTestSecret/7c2204c6093c4d859
bc5b9eff8f29050","attributes":
{"enabled":true,"created":1505088747,"updated":1505088747,"recoveryLevel":"Purgeable"}}
```

Once you've retrieved the secret from the Key Vault, you can use it to authenticate to a service that requires a name and password.

Next steps

In this tutorial, you learned how use a Windows VM system-assigned managed identity to access Azure Key Vault. To learn more about Azure Key Vault see:

Azure Key Vault

Tutorial: Use a Windows VM system-assigned managed identity to access Azure Cosmos DB

4/11/2019 • 6 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Windows virtual machine (VM) to access Cosmos DB. You learn how to:

- Create a Cosmos DB account
- Grant a Windows VM system-assigned managed identity access to the Cosmos DB account access keys
- Get an access token using the Windows VM system-assigned managed identity to call Azure Resource Manager
- Get access keys from Azure Resource Manager to make Cosmos DB calls

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled
- Install the latest version of Azure PowerShell

Create a Cosmos DB account

If you don't already have one, create a Cosmos DB account. You can skip this step and use an existing Cosmos DB account.

1. Click the **+/Create new service** button found on the upper left-hand corner of the Azure portal.
2. Click **Databases**, then **Azure Cosmos DB**, and a new "New account" panel displays.
3. Enter an **ID** for the Cosmos DB account, which you use later.
4. **API** should be set to "SQL." The approach described in this tutorial can be used with the other available API types, but the steps in this tutorial are for the SQL API.
5. Ensure the **Subscription** and **Resource Group** match the ones you specified when you created your VM in the previous step. Select a **Location** where Cosmos DB is available.
6. Click **Create**.

Create a collection in the Cosmos DB account

Next, add a data collection in the Cosmos DB account that you can query in later steps.

1. Navigate to your newly created Cosmos DB account.
2. On the **Overview** tab click the **+/Add Collection** button, and an "Add Collection" panel slides out.
3. Give the collection a database ID, collection ID, select a storage capacity, enter a partition key, enter a throughput value, then click **OK**. For this tutorial, it is sufficient to use "Test" as the database ID and collection ID, select a fixed storage capacity and lowest throughput (400 RU/s).

Grant Windows VM system-assigned managed identity access to the Cosmos DB account access keys

Cosmos DB does not natively support Azure AD authentication. However, you can use a system-assigned managed identity to retrieve a Cosmos DB access key from the Resource Manager, and use the key to access Cosmos DB. In this step, you grant your Windows VM system-assigned managed identity access to the keys to the Cosmos DB account.

To grant the Windows VM system-assigned managed identity access to the Cosmos DB account in Azure Resource Manager using PowerShell, update the values for `<SUBSCRIPTION ID>`, `<RESOURCE GROUP>`, and `<COSMOS DB ACCOUNT NAME>` for your environment. Cosmos DB supports two levels of granularity when using access keys: read/write access to the account, and read-only access to the account. Assign the `DocumentDB Account Contributor` role if you want to get read/write keys for the account, or assign the `Cosmos DB Account Reader Role` role if you want to get read-only keys for the account. For this tutorial, assign the `Cosmos DB Account Reader Role`:

```
$spID = (Get-AzVM -ResourceGroupName myRG -Name myVM).identity.principalid
New-AzRoleAssignment -ObjectId $spID -RoleDefinitionName "Cosmos DB Account Reader Role" -Scope
"/subscriptions/<mySubscriptionID>/resourceGroups/<myResourceGroup>/providers/Microsoft.DocumentDb/databaseAcco
unts/<COSMOS DB ACCOUNT NAME>"
```

Get an access token using the Windows VM system-assigned managed identity to call Azure Resource Manager

For the remainder of the tutorial, we will work from the VM we created earlier.

You will need to install the latest version of Azure CLI on your Windows VM.

1. In the Azure portal, navigate to **Virtual Machines**, go to your Windows virtual machine, then from the **Overview** page click **Connect** at the top.
2. Enter in your **Username** and **Password** for which you added when you created the Windows VM.
3. Now that you have created a **Remote Desktop Connection** with the virtual machine, open PowerShell in the remote session.
4. Using Powershell’s Invoke-WebRequest, make a request to the local managed identities for Azure resources endpoint to get an access token for Azure Resource Manager.

   ```
   $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-
   version=2018-02-01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -Method GET -Headers
   @{Metadata="true"}
   ```

   NOTE

   The value of the "resource" parameter must be an exact match for what is expected by Azure AD. When using the Azure Resource Manager resource ID, you must include the trailing slash on the URI.

Next, extract the "Content" element, which is stored as a JavaScript Object Notation (JSON) formatted string in the $response object.

```
$content = $response.Content | ConvertFrom-Json
```

Next, extract the access token from the response.

```
$ArmToken = $content.access_token
```

Get access keys from Azure Resource Manager to make Cosmos DB calls

Now use PowerShell to call Resource Manager using the access token retrieved in the previous section to retrieve the Cosmos DB account access key. Once we have the access key, we can query Cosmos DB. Be sure to replace the `<SUBSCRIPTION ID>`, `<RESOURCE GROUP>`, and `<COSMOS DB ACCOUNT NAME>` parameter values with your own values. Replace the `<ACCESS TOKEN>` value with the access token you retrieved earlier. If you want to retrieve read/write keys, use key operation type `listKeys`. If you want to retrieve read-only keys, use the key operation type `readonlykeys`:

```
Invoke-WebRequest -Uri 'https://management.azure.com/subscriptions/<SUBSCRIPTION-ID>/resourceGroups/<RESOURCE-
GROUP>/providers/Microsoft.DocumentDb/databaseAccounts/<COSMOS DB ACCOUNT NAME>/listKeys/?api-version=2016-03-
31' -Method POST -Headers @{Authorization="Bearer $ARMToken"}
```

The response give you the list of Keys. For example, if you get read-only keys:

```
{"primaryReadonlyMasterKey":"bWpDxS...dzQ==",
"secondaryReadonlyMasterKey":"38v5ns...7bA=="}
```

Now that you have the access key for the Cosmos DB account you can pass it to a Cosmos DB SDK and make calls to access the account. For a quick example, you can pass the access key to the Azure CLI. You can get the `<COSMOS DB CONNECTION URL>` from the **Overview** tab on the Cosmos DB account blade in the Azure portal. Replace the `<ACCESS KEY>` with the value you obtained above:

```
az cosmosdb collection show -c <COLLECTION ID> -d <DATABASE ID> --url-connection "<COSMOS DB CONNECTION URL>" -
-key <ACCESS KEY>
```

This CLI command returns details about the collection:

```
{
  "collection": {
    "_conflicts": "conflicts/",
    "_docs": "docs/",
    "_etag": "\"00006700-0000-0000-0000-5a8271e90000\"",
    "_rid": "Es5SAM2FDwA=",
    "_self": "dbs/Es5SAA==/colls/Es5SAM2FDwA=/",
    "_sprocs": "sprocs/",
    "_triggers": "triggers/",
    "_ts": 1518498281,
    "_udfs": "udfs/",
    "id": "Test",
    "indexingPolicy": {
      "automatic": true,
      "excludedPaths": [],
      "includedPaths": [
        {
          "indexes": [
            {
              "dataType": "Number",
              "kind": "Range",
              "precision": -1
            },
            {
              "dataType": "String",
              "kind": "Range",
              "precision": -1
            },
            {
              "dataType": "Point",
              "kind": "Spatial"
            }
          ],
          "path": "/*"
        }
      ],
      "indexingMode": "consistent"
    }
  },
  "offer": {
    "_etag": "\"00006800-0000-0000-0000-5a8271ea0000\"",
    "_rid": "f4V+",
    "_self": "offers/f4V+/",
    "_ts": 1518498282,
    "content": {
      "offerIsRUPerMinuteThroughputEnabled": false,
      "offerThroughput": 400
    },
    "id": "f4V+",
    "offerResourceId": "Es5SAM2FDwA=",
    "offerType": "Invalid",
    "offerVersion": "V2",
    "resource": "dbs/Es5SAA==/colls/Es5SAM2FDwA=/"
  }
}
```

Next steps

In this tutorial, you learned how to use a Windows VM system-assigned identity to access Cosmos DB. To learn more about Cosmos DB see:

Azure Cosmos DB overview

Tutorial: Use a Windows VM system-assigned managed identity to access Azure AD Graph API

4/19/2019 • 6 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Windows virtual machine (VM) to access the Azure AD Graph API to retrieve its group memberships. Managed identities for Azure resources are automatically managed by Azure and enable you to authenticate to services that support Azure AD authentication without needing to insert credentials into your code. For this tutorial you will query your VM identity's membership in Azure AD groups. Group information is often used in authorization decisions, for example. Under the covers, your VM's managed identity is represented by a **Service Principal** in Azure AD. Before you do the group query, add the service principal representing the VM's identity to a group in Azure AD. You can do this using Azure PowerShell, Azure AD PowerShell, or the Azure CLI.

- Connect to Azure AD
- Add your VM identity to a group in Azure AD
- Grant the VM's identity access to Azure AD Graph
- Get an access token using the VM identity and use it to call Azure AD Graph

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

- To grant a VM identity access to Azure AD Graph, your account needs to be assigned the **Global Admin** role in Azure AD.
- Install the latest Azure AD PowerShell if you haven't already.

Connect to Azure AD

You need to connect to Azure AD to assign the VM to a group as well as grant the VM permission to retrieve its group memberships. You can use Connect-AzureAD cmdlet directly or with TenantId paramter in case you have multiple tenants.

```
Connect-AzureAD
```

OR

```
Connect-AzureAD -TenantId "Object Id of the tenant"
```

Add your VM identity to a group in Azure AD

When you enabled system-assigned managed identity on the Windows VM, it created a service principal in Azure AD. Now you need to add the VM to a group. The following example creates a new group in Azure AD and adds your VM's service principal to that group:

```
New-AzureADGroup -DisplayName "myGroup" -MailEnabled $false -SecurityEnabled $true -MailNickName "NotSet"
$AzureADGroup = Get-AzureADGroup -Filter "displayName eq 'myGroup'"
$ManagedIdentitiesServicePrincipal = Get-AzureADServicePrincipal -Filter "displayName eq 'myVM'"
Add-AzureADGroupMember -ObjectId $AzureADGroup.ObjectID -RefObjectId
$ManagedIdentitiesServicePrincipal.ObjectId
```

Grant your VM access to the Azure AD Graph API

Using managed identities for Azure resources, your code can get access tokens to authenticate to resources that support Azure AD authentication. The Microsoft Azure AD Graph API supports Azure AD authentication. In this step, you will grant your VM identity's service principal access to the Azure AD Graph so that it can query group memberships. Service principals are granted access to the Microsoft or Azure AD Graph through **Application Permissions**. The type of application permission you need to grant depends on the entity you want to access in the MS or Azure AD Graph.

For this tutorial, you will grant your VM identity the ability to query group memberships using the `Directory.Read.All` application permission. To grant this permission, you will need a user account that is assigned the Global Admin role in Azure AD. Normally you would grant an application permission by visiting your application's registration in the Azure portal and adding the permission there. However, managed identities for Azure resources does not register application objects in Azure AD, it only registers service principals. To register the application permission you will use the Azure AD PowerShell command line tool.

Azure AD Graph:

- Service Principal appId (used when granting app permission): 00000002-0000-0000-c000-000000000000
- Resource ID (used when requesting access token from managed identities for Azure resources): https://graph.windows.net
- Permission scope reference: Azure AD Graph Permissions Reference

Grant application permissions using Azure AD PowerShell

You will need Azure AD PowerShell to use this option. If you don't have it installed, download the latest version before continuing.

1. Open a PowerShell window and connect to Azure AD:

   ```
   Connect-AzureAD
   ```

   To connect to a specific Azure Active Directory, use the *TenantId* parameter, as follows:

   ```
   Connect-AzureAD -TenantId "Object Id of the tenant"
   ```

2. Run the following PowerShell commands to assign the `Directory.Read.All` application permission to the service principal that represents your VM's identity.

```
$ManagedIdentitiesServicePrincipal = Get-AzureADServicePrincipal -Filter "displayName eq 'myVM'"
$GraphAppId = "00000002-0000-0000-c000-000000000000"
$GraphServicePrincipal = Get-AzureADServicePrincipal -Filter "appId eq '$GraphAppId'"
$PermissionName = "Directory.Read.All"
$AppRole = $GraphServicePrincipal.AppRoles | Where-Object {$_.Value -eq $PermissionName -and
$_.AllowedMemberTypes -contains "Application"}
New-AzureAdServiceAppRoleAssignment -ObjectId $ManagedIdentitiesServicePrincipal.ObjectId -PrincipalId
$ManagedIdentitiesServicePrincipal.ObjectId -ResourceId $GraphServicePrincipal.ObjectId -Id $AppRole.Id
```

Output from the final command should look like this, returning the ID of the assignment:

`ObjectId` : `gzR5KyLAiUOTiqFhNeWZWBtK7ZKqNJxAiWYXYVHlgMs`

`ResourceDisplayName` : `Windows Azure Active Directory`

`PrincipalDisplayName` : `myVM`

If the call to `New-AzureAdServiceAppRoleAssignment` fails with the error `bad request, one or more properties are invalid` the app permission may already be assigned to the VM identity's service principal. You can use the following PowerShell commands to check if the application permission already exists between your VM's identity and Azure AD Graph:

```
$ManagedIdentitiesServicePrincipal = Get-AzureADServicePrincipal -Filter "displayName eq '<VM-NAME>'"
$GraphAppId = "00000002-0000-0000-c000-000000000000"
$GraphServicePrincipal = Get-AzureADServicePrincipal -Filter "appId eq '$GraphAppId'"
$PermissionName = "Directory.Read.All"
$AppRole = $GraphServicePrincipal.AppRoles | Where-Object {$_.Value -eq $PermissionName -and
$_.AllowedMemberTypes -contains "Application"}
Get-AzureADServiceAppRoleAssignment -ObjectId $GraphServicePrincipal.ObjectId | Where-Object {$_.Id -eq
$AppRole.Id -and $_.PrincipalId -eq $ManagedIdentitiesServicePrincipal.ObjectId}
```

You can use the following PowerShell commands to list all app permissions that have been granted to Azure AD Graph:

```
$GraphAppId = "00000002-0000-0000-c000-000000000000"
$GraphServicePrincipal = Get-AzureADServicePrincipal -Filter "appId eq '$GraphAppId'"
Get-AzureADServiceAppRoleAssignment -ObjectId $GraphServicePrincipal.ObjectId
```

You can use the following PowerShell commands to remove app permissions that have been granted to your VM's identity for Azure AD Graph:

```
$ManagedIdentitiesServicePrincipal = Get-AzureADServicePrincipal -Filter "displayName eq '<VM-NAME>'"
$GraphAppId = "00000002-0000-0000-c000-000000000000"
$GraphServicePrincipal = Get-AzureADServicePrincipal -Filter "appId eq '$GraphAppId'"
$PermissionName = "Directory.Read.All"
$AppRole = $GraphServicePrincipal.AppRoles | Where-Object {$_.Value -eq $PermissionName -and
$_.AllowedMemberTypes -contains "Application"}
$ServiceAppRoleAssignment = Get-AzureADServiceAppRoleAssignment -ObjectId
$GraphServicePrincipal.ObjectId | Where-Object {$_.Id -eq $AppRole.Id -and $_.PrincipalId -eq
$ManagedIdentitiesServicePrincipal.ObjectId}
Remove-AzureADServiceAppRoleAssignment -AppRoleAssignmentId $ServiceAppRoleAssignment.ObjectId -ObjectId
$ManagedIdentitiesServicePrincipal.ObjectId
```

Get an access token using the VM's identity to call Azure AD Graph

To use the VM's system assigned managed identity for authentication to Azure AD Graph, you need to make requests from the VM.

1. In the portal, navigate to **Virtual Machines**, go to your Windows VM, and in the **Overview** blade, click **Connect**.
2. Enter in your **Username** and **Password** you used when you created the Windows VM.
3. Now that you have created a Remote Desktop Connection with the virtual machine, open PowerShell in the remote session.
4. Using PowerShell's Invoke-WebRequest, make a request to the local managed identities for Azure resources endpoint to get an access token for Azure AD Graph.

   ```
   $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-
   version=2018-02-01&resource=https://graph.windows.net' -Method GET -Headers @{Metadata="true"}
   ```

   Convert the response from a JSON object to a PowerShell object.

   ```
   $content = $response.Content | ConvertFrom-Json
   ```

   Extract the access token from the response.

   ```
   $AccessToken = $content.access_token
   ```

5. Using the Object ID of your VM identity's service principal (you can retrieve this value from the variable declared in previous steps: `$ManagedIdentitiesServicePrincipal.ObjectId`), you can query the Azure AD Graph API to retrieve its group memberships. Replace `<OBJECT ID>` with the Object ID from the previous step and < `ACCESS-TOKEN>` with the previously obtained access token:

   ```
   Invoke-WebRequest 'https://graph.windows.net/<Tenant ID>/servicePrincipals/<VM Object
   ID>/getMemberGroups?api-version=1.6' -Method POST -Body '{"securityEnabledOnly":"false"}' -Headers
   @{Authorization="Bearer $AccessToken"} -ContentType "application/json"
   ```

   The response contains the `Object ID` of the group that you added your VM's service principal to in earlier steps.

   Response:

   ```
   Content : {"odata.metadata":"https://graph.windows.net/<Tenant
   ID>/$metadata#Collection(Edm.String)","value":["<ObjectID of VM's group membership>"]}
   ```

Next steps

In this tutorial, you learned how to use a Windows VM system-assigned managed identity to access Azure AD Graph. To learn more about Azure AD Graph see:

Azure AD Graph

Tutorial: Use a User-assigned Managed Identity on a Windows VM, to access Azure Resource Manager

4/11/2019 • 4 minutes to read • Edit Online

User assigned managed identities are a preview feature of Azure Active Directory. Make sure you review the known issues before you begin. For more information about previews, see Supplemental Terms of Use for Microsoft Azure Previews.

This tutorial explains how to create a user-assigned identity, assign it to a Windows Virtual Machine (VM), and then use that identity to access the Azure Resource Manager API. Managed Service Identities are automatically managed by Azure. They enable authentication to services that support Azure AD authentication, without needing to embed credentials into your code.

You learn how to:

- Create a user-assigned managed identity
- Assign your user-assigned identity to your Windows VM
- Grant the user-assigned identity access to a Resource Group in Azure Resource Manager
- Get an access token using the user-assigned identity and use it to call Azure Resource Manager
- Read the properties of a Resource Group

NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- Sign in to Azure portal
- Create a Windows virtual machine
- To perform the required resource creation and role management steps in this tutorial, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Install the latest version of the Azure PowerShell module.
- Run `Connect-AzAccount` to create a connection with Azure.
- Install the latest version of PowerShellGet.
- Run `Install-Module -Name PowerShellGet -AllowPrerelease` to get the pre-release version of the

`PowerShellGet` module (you may need to `Exit` out of the current PowerShell session after you run this command to install the `Az.ManagedServiceIdentity` module).

- Run `Install-Module -Name Az.ManagedServiceIdentity -AllowPrerelease` to install the prerelease version of the `Az.ManagedServiceIdentity` module to perform the user-assigned identity operations in this article.

Create a user-assigned identity

A user-assigned identity is created as a standalone Azure resource. Using the New-AzUserAssignedIdentity, Azure creates an identity in your Azure AD tenant that can be assigned to one or more Azure service instances.

IMPORTANT

When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

```
New-AzUserAssignedIdentity -ResourceGroupName myResourceGroupVM -Name ID1
```

The response contains details for the user-assigned identity created, similar to the following example. Note the `Id` and `ClientId` values for your user-assigned identity, because they are used in subsequent steps:

```
{
Id:
/subscriptions/<SUBSCRIPTIONID>/resourcegroups/myResourceGroupVM/providers/Microsoft.ManagedIdentity/userAssign
edIdentities/ID1
ResourceGroupName : myResourceGroupVM
Name: ID1
Location: westus
TenantId: 733a8f0e-ec41-4e69-8ad8-971fc4b533f8
PrincipalId: e591178e-b785-43c8-95d2-1397559b2fb9
ClientId: af825a31-b0e0-471f-baea-96de555632f9
ClientSecretUrl: https://control-
westus.identity.azure.net/subscriptions/<SUBSCRIPTIONID>/resourcegroups/myResourceGroupVM/providers/Microsoft.M
anagedIdentity/userAssignedIdentities/ID1/credentials?tid=733a8f0e-ec41-4e69-8ad8-971fc4b533f8&oid=e591178e-
b785-43c8-95d2-1397559b2fb9&aid=af825a31-b0e0-471f-baea-96de555632f9
Type: Microsoft.ManagedIdentity/userAssignedIdentities
}
```

Assign the user-assigned identity to a Windows VM

A user-assigned identity can be used by clients on multiple Azure resources. Use the following commands to assign the user-assigned identity to a single VM. Use the `Id` property returned in the previous step for the `-IdentityID` parameter.

```
$vm = Get-AzVM -ResourceGroupName myResourceGroup -Name myVM
Update-AzVM -ResourceGroupName TestRG -VM $vm -IdentityType "UserAssigned" -IdentityID
"/subscriptions/<SUBSCRIPTIONID>/resourcegroups/myResourceGroupVM/providers/Microsoft.ManagedIdentity/userAssig
nedIdentities/ID1"
```

Grant your user-assigned identity access to a Resource Group in Azure Resource Manager

Managed identities for Azure resources provides identities that your code can use to request access tokens to authenticate to resource APIs that support Azure AD authentication. In this tutorial, your code will access the Azure Resource Manager API.

Before your code can access the API, you need to grant the identity access to a resource in Azure Resource Manager. In this case, the Resource Group in which the VM is contained. Update the value for `<SUBSCRIPTION ID>` as appropriate for your environment.

```
$spID = (Get-AzUserAssignedIdentity -ResourceGroupName myResourceGroupVM -Name ID1).principalid
New-AzRoleAssignment -ObjectId $spID -RoleDefinitionName "Reader" -Scope
"/subscriptions/<SUBSCRIPTIONID>/resourcegroups/myResourceGroupVM/"
```

The response contains details for the role assignment created, similar to the following example:

```
RoleAssignmentId: /subscriptions/80c696ff-5efa-4909-a64d-
f1b616f423ca/resourcegroups/myResourceGroupVM/providers/Microsoft.Authorization/roleAssignments/f9cc753d-265e-
4434-ae19-0c3e2ead62ac
Scope: /subscriptions/80c696ff-5efa-4909-a64d-f1b616f423ca/resourcegroups/myResourceGroupVM
DisplayName: ID1
SignInName:
RoleDefinitionName: Reader
RoleDefinitionId: acdd72a7-3385-48ef-bd42-f606fba81ae7
ObjectId: e591178e-b785-43c8-95d2-1397559b2fb9
ObjectType: ServicePrincipal
CanDelegate: False
```

Get an access token using the VM's identity and use it to call Resource Manager

For the remainder of the tutorial, you will work from the VM we created earlier.

1. Sign in to the Azure portal at https://portal.azure.com
2. In the portal, navigate to **Virtual Machines** and go to the Windows virtual machine and in the **Overview**, click **Connect**.
3. Enter the **Username** and **Password** you used when you created the Windows VM.
4. Now that you have created a **Remote Desktop Connection** with the virtual machine, open **PowerShell** in the remote session.
5. Using PowerShell's `Invoke-WebRequest`, make a request to the local managed identities for Azure resources endpoint to get an access token for Azure Resource Manager. The `client_id` value is the value returned when you created the user-assigned managed identity.

   ```
   $response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-
   version=2018-02-01&client_id=af825a31-b0e0-471f-baea-
   96de555632f9&resource=https://management.azure.com/' -Method GET -Headers @{Metadata="true"}
   $content = $response.Content | ConvertFrom-Json
   $ArmToken = $content.access_token
   ```

Read the properties of a Resource Group

Use the access token retrieved in the previous step to access Azure Resource Manager, and read the properties of the Resource Group you granted your user-assigned identity access. Replace `<SUBSCRIPTION ID>` with the subscription id of your environment.

```
(Invoke-WebRequest -Uri https://management.azure.com/subscriptions/80c696ff-5efa-4909-a64d-
f1b616f423ca/resourceGroups/myResourceGroupVM?api-version=2016-06-01 -Method GET -ContentType
"application/json" -Headers @{Authorization ="Bearer $ArmToken"}).content
```

The response contains the specific Resource Group information, similar to the following example:

```
{"id":"/subscriptions/<SUBSCRIPTIONID>/resourceGroups/myResourceGroupVM","name":"myResourceGroupVM","location":
"eastus","properties":{"provisioningState":"Succeeded"}}
```

Next steps

In this tutorial, you learned how to create a user-assigned identity and attach it to an Azure Virtual Machine to access the Azure Resource Manager API. To learn more about Azure Resource Manager see:

Azure Resource Manager

Tutorial: Use a Linux VM system-assigned managed identity to access Azure Data Lake Store

3/26/2019 • 5 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Linux virtual machine (VM) to access Azure Data Lake Store. You learn how to:

In this tutorial, you learn how to:

- Grant your VM access to Azure Data Lake Store.
- Get an access token by using the VM's system-assigned managed identity to access Azure Data Lake Store.

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Grant your VM access to Azure Data Lake Store

Now you can grant your VM access to files and folders in Azure Data Lake Store. For this step, you can use an existing Data Lake Store instance or create a new one. To create a Data Lake Store instance by using the Azure portal, follow the Azure Data Lake Store quickstart. There are also quickstarts that use Azure CLI and Azure PowerShell in the Azure Data Lake Store documentation.

In Data Lake Store, create a new folder and grant our Linux VM system-assigned managed identity permission to read, write, and execute files in that folder:

1. In the Azure portal, select **Data Lake Store** in the left pane.
2. Select the Data Lake Store instance that you want to use.
3. Select **Data Explorer** on the command bar.
4. The root folder of the Data Lake Store instance is selected. Select **Access** on the command bar.
5. Select **Add**. In the **Select** box, enter the name of your VM--for example, **DevTestVM**. Select your VM from the search results, and then click **Select**.
6. Click **Select Permissions**. Select **Read** and **Execute**, add to **This folder**, and add as **An access permission only**. Select **Ok**. The permission should be added successfully.
7. Close the **Access** pane.
8. For this tutorial, create a new folder. Select **New Folder** on the command bar, and give the new folder a name--

for example **TestFolder**. Select **Ok**.

9. Select the folder that you created, and then select **Access** on the command bar.
10. Similar to step 5, select **Add**. In the **Select** box, enter the name of your VM. Select your VM from the search results, and then click **Select**.
11. Similar to step 6, select **Select Permissions**. Select **Read**, **Write**, and **Execute**, add to **This folder**, and add as **An access permission entry and a default permission entry**. Select **Ok**. The permission should be added successfully.

Managed identities for Azure resources can now perform all operations on files in the folder that you created. For more information on managing access to Data Lake Store, see Access Control in Data Lake Store.

Get an access token and call the Data Lake Store file system

Azure Data Lake Store natively supports Azure AD authentication, so it can directly accept access tokens obtained via using managed identities for Azure resources. To authenticate to the Data Lake Store file system, you send an access token issued by Azure AD to your Data Lake Store file system endpoint. The access token is in an authorization header in the format "Bearer <ACCESS_TOKEN_VALUE>". To learn more about Data Lake Store support for Azure AD authentication, see Authentication with Data Lake Store using Azure Active Directory.

In this tutorial, you authenticate to the REST API for the Data Lake Store file system by using cURL to make REST requests.

NOTE

The client SDKs for the Data Lake Store file system do not yet support managed identities for Azure resources.

To complete these steps, you need an SSH client. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to use SSH keys with Windows on Azure or How to create and use an SSH public and private key pair for Linux VMs in Azure.

1. In the portal, browse to your Linux VM. In **Overview**, select **Connect**.

2. Connect to the VM by using the SSH client of your choice.

3. In the terminal window, by using cURL, make a request to the local managed identities Azure for Azure resources endpoint to get an access token for the Data Lake Store file system. The resource identifier for Data Lake Store is `https://datalake.azure.net/`. It's important to include the trailing slash in the resource identifier.

   ```
   curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
   01&resource=https%3A%2F%2Fdatalake.azure.net%2F' -H Metadata:true
   ```

   A successful response returns the access token that you use to authenticate to Data Lake Store:

   ```
   {"access_token":"eyJ0eXAiOiJ...",
    "refresh_token":"",
    "expires_in":"3599",
    "expires_on":"1508119757",
    "not_before":"1508115857",
    "resource":"https://datalake.azure.net/",
    "token_type":"Bearer"}
   ```

4. By using cURL, make a request to your Data Lake Store file system's REST endpoint to list the folders in the root folder. This is a simple way to check that everything is configured correctly. Copy the value of the access token from the previous step. It's important that the string "Bearer" in the Authorization header has a capital

"B." You can find the name of your Data Lake Store instance in the **Overview** section of the **Data Lake Store** pane in the Azure portal.

```
curl https://<YOUR_ADLS_NAME>.azuredatalakestore.net/webhdfs/v1/?op=LISTSTATUS -H "Authorization: Bearer
<ACCESS_TOKEN>"
```

A successful response looks like this:

```
{"FileStatuses":{"FileStatus":
[{"length":0,"pathSuffix":"TestFolder","type":"DIRECTORY","blockSize":0,"accessTime":1507934941392,"modi
ficationTime":1508105430590,"replication":0,"permission":"770","owner":"bd0e76d8-ad45-4fe1-8941-
04a7bf27f071","group":"bd0e76d8-ad45-4fe1-8941-04a7bf27f071"}]}}
```

5. Now you can try uploading a file to your Data Lake Store instance. First, create a file to upload.

```
echo "Test file." > Test1.txt
```

6. By using cURL, make a request to your Data Lake Store file system's REST endpoint to upload the file to the folder that you created earlier. The upload involves a redirect, and cURL follows the redirect automatically.

```
curl -i -X PUT -L -T Test1.txt -H "Authorization: Bearer <ACCESS_TOKEN>"
'https://<YOUR_ADLS_NAME>.azuredatalakestore.net/webhdfs/v1/<FOLDER_NAME>/Test1.txt?op=CREATE'
```

A successful response looks like this:

```
HTTP/1.1 100 Continue
HTTP/1.1 307 Temporary Redirect
Cache-Control: no-cache, no-cache, no-store, max-age=0
Pragma: no-cache
Expires: -1
Location: https://mytestadls.azuredatalakestore.net/webhdfs/v1/TestFolder/Test1.txt?op=CREATE&write=true
x-ms-request-id: 756f6b24-0cca-47ef-aa12-52c3b45b954c
ContentLength: 0
x-ms-webhdfs-version: 17.04.22.00
Status: 0x0
X-Content-Type-Options: nosniff
Strict-Transport-Security: max-age=15724800; includeSubDomains
Date: Sun, 15 Oct 2017 22:10:30 GMT
Content-Length: 0

HTTP/1.1 100 Continue

HTTP/1.1 201 Created
Cache-Control: no-cache, no-cache, no-store, max-age=0
Pragma: no-cache
Expires: -1
Location: https://mytestadls.azuredatalakestore.net/webhdfs/v1/TestFolder/Test1.txt?op=CREATE&write=true
x-ms-request-id: af5baa07-3c79-43af-a01a-71d63d53e6c4
ContentLength: 0
x-ms-webhdfs-version: 17.04.22.00
Status: 0x0
X-Content-Type-Options: nosniff
Strict-Transport-Security: max-age=15724800; includeSubDomains
Date: Sun, 15 Oct 2017 22:10:30 GMT
Content-Length: 0
```

By using other APIs for the Data Lake Store file system, you can append to files, download files, and more.

Next steps

In this tutorial, you learned how to use a Linux VM system-assigned managed identity to access an Azure Data Lake Store. To learn more about Azure Data Lake Store see:

Azure Data Lake Store

Tutorial: Use a Linux VM system-assigned managed identity to access Azure Storage

4/1/2019 • 4 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Linux virtual machine (VM) to access Azure Storage. You learn how to:

- Create a storage account
- Create a blob container in a storage account
- Grant the Linux VM's Managed Identity access to an Azure Storage container
- Get an access token and use it to call Azure Storage

NOTE

Azure Active Directory authentication for Azure Storage is in public preview.

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

To run the CLI script examples in this tutorial, you have two options:

- Use Azure Cloud Shell either from the Azure portal, or via the **Try It** button, located in the top right corner of each code block.
- Install the latest version of CLI 2.0 (2.0.23 or later) if you prefer to use a local CLI console.

Create a storage account

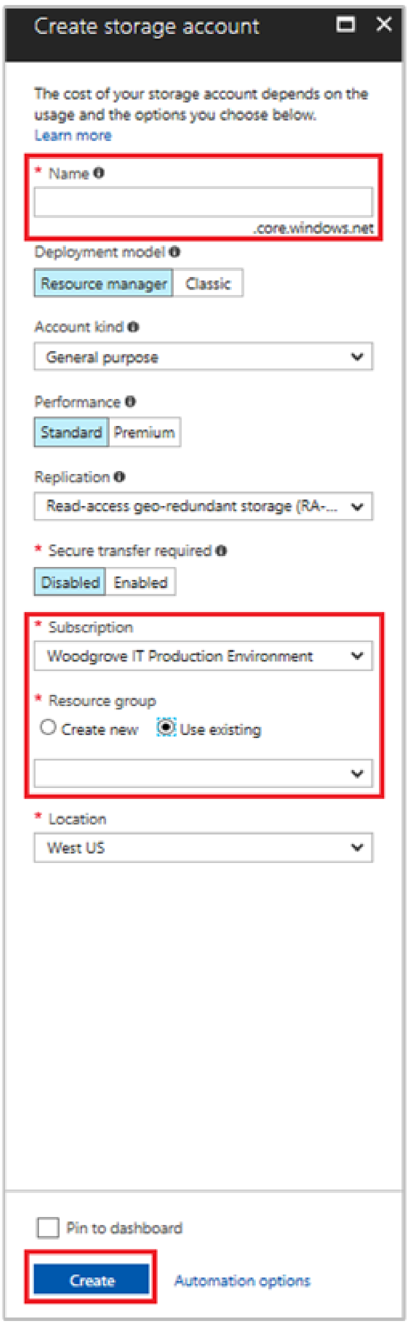
In this section, you create a storage account.

1. Click the **+ Create a resource** button found on the upper left-hand corner of the Azure portal.
2. Click **Storage**, then **Storage account - blob, file, table, queue**.
3. Under **Name**, enter a name for the storage account.
4. **Deployment model** and **Account kind** should be set to **Resource manager** and **Storage (general**

purpose v1).

5. Ensure the **Subscription** and **Resource Group** match the ones you specified when you created your VM in the previous step.

6. Click **Create**.



Create storage account

The cost of your storage account depends on the usage and the options you choose below.
Learn more

* Name

.core.windows.net

Deployment model: [✓ Resource manager] Classic

Account kind: General purpose

Performance: [✓ Standard] Premium

Replication: Read-access geo-redundant storage (RA-...

* Secure transfer required: [✓ Disabled] Enabled

* Subscription: Woodgrove IT Production Environment

* Resource group: Create new [✓ Use existing]

* Location: West US

Pin to dashboard

Create   Automation options

Create a blob container and upload a file to the storage account

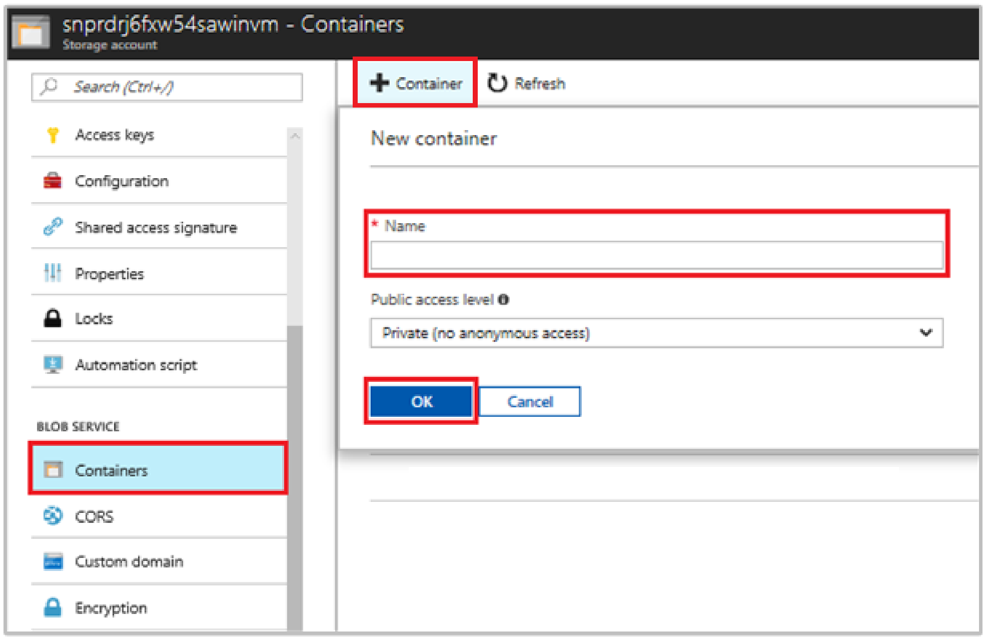
Files require blob storage so you need to create a blob container in which to store the file. You then upload a file to the blob container in the new storage account.

1. Navigate back to your newly created storage account.

2. Under **Blob Service**, click **Containers**.

3. Click **+ Container** on the top of the page.

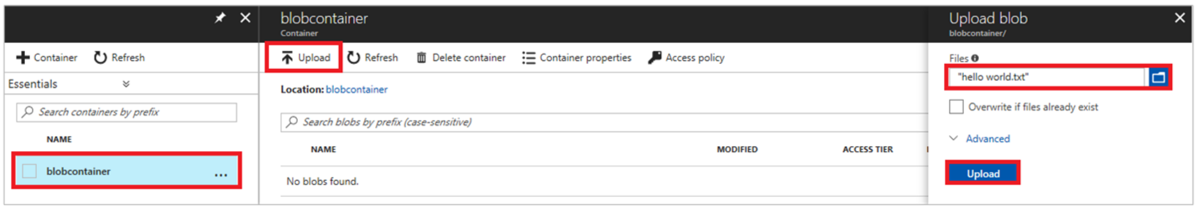
4. Under **New container**, enter a name for the container and under **Public access level** keep the default value .



5. Using an editor of your choice, create a file titled *hello world.txt* on your local machine. Open the file and add the text (without the quotes) "Hello world! :)" and then save it.

6. Upload the file to the newly created container by clicking on the container name, then **Upload**

7. In the **Upload blob** pane, under **Files**, click the folder icon and browse to the file **hello_world.txt** on your local machine, select the file, then click **Upload**.



Grant your VM access to an Azure Storage container

You can use the VM's managed identity to retrieve the data in the Azure storage blob.

1. Navigate back to your newly created storage account.

2. Click the **Access control (IAM)** link in the left panel.

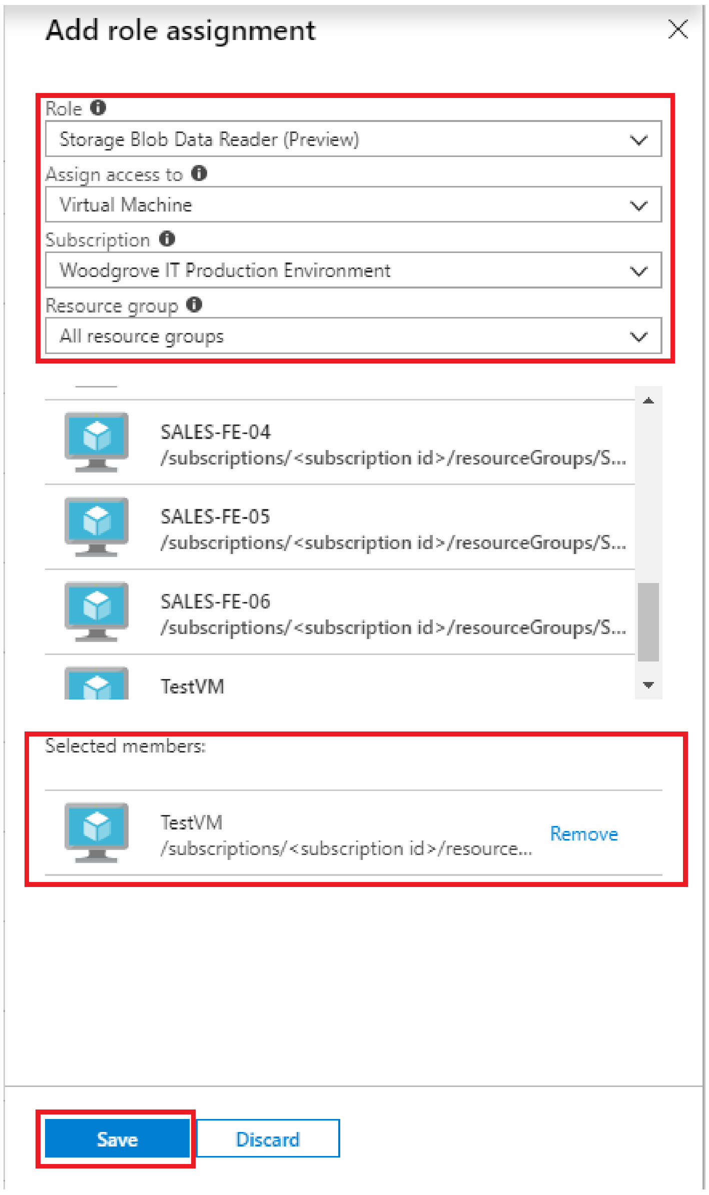
3. Click **+ Add role assignment** on top of the page to add a new role assignment for your VM.

4. Under **Role**, from the dropdown, select **Storage Blob Data Reader**.

5. In the next dropdown, under **Assign access to**, choose **Virtual Machine**.

6. Next, ensure the proper subscription is listed in **Subscription** dropdown and then set **Resource Group** to **All resource groups**.

7. Under **Select**, choose your VM and then click **Save**.


Add role assignment

Role
Storage Blob Data Reader (Preview)
Assign access to
Virtual Machine
Subscription
Woodgrove IT Production Environment
Resource group
All resource groups

SALES-FE-04
/subscriptions/<subscription id>/resourceGroups/S...
SALES-FE-05
/subscriptions/<subscription id>/resourceGroups/S...
SALES-FE-06
/subscriptions/<subscription id>/resourceGroups/S...
TestVM

Selected members:
TestVM
/subscriptions/<subscription id>/resource...
Remove

Get an access token and use it to call Azure Storage

Azure Storage natively supports Azure AD authentication, so it can directly accept access tokens obtained using a Managed Identity. This is part of Azure Storage's integration with Azure AD, and is different from supplying credentials on the connection string.

To complete the following steps, you need to work from the VM created earlier and you need an SSH client to connect to it. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to Use SSH keys with Windows on Azure, or How to create and use an SSH public and private key pair for Linux VMs in Azure.

1. In the Azure portal, navigate to **Virtual Machines**, go to your Linux virtual machine, then from the **Overview** page click **Connect**. Copy the string to connect to your VM.

2. **Connect** to the VM with the SSH client of your choice.

3. In the terminal window, using CURL, make a request to the local Managed Identity endpoint to get an access token for Azure Storage.

```
curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
01&resource=https%3A%2F%2Fstorage.azure.com%2F' -H Metadata:true
```

4. Now use the access token to access Azure Storage, for example to read the contents of the sample file which

you previously uploaded to the container. Replace the values of `<STORAGE ACCOUNT>`, `<CONTAINER NAME>`, and `<FILE NAME>` with the values you specified earlier, and `<ACCESS TOKEN>` with the token returned in the previous step.

```
curl https://<STORAGE ACCOUNT>.blob.core.windows.net/<CONTAINER NAME>/<FILE NAME> -H "x-ms-version:
2017-11-09" -H "Authorization: Bearer <ACCESS TOKEN>"
```

The response contains the contents of the file:

```
Hello world! :)
```

Next steps

In this tutorial, you learned how enable a Linux VM system-assigned managed identity to access Azure Storage. To learn more about Azure Storage see:

Azure Storage

Tutorial: Use a Linux VM system-assigned managed identity to access Azure Storage via access key

3/26/2019 • 6 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Linux virtual machine (VM) to retrieve storage account access keys. You can use a storage access key as usual when doing storage operations, for example when using the Storage SDK. For this tutorial, we upload and download blobs using Azure CLI. You will learn how to:

- Grant your VM access to storage account access keys in Resource Manager
- Get an access token using your VM's identity, and use it to retrieve the storage access keys from Resource Manager

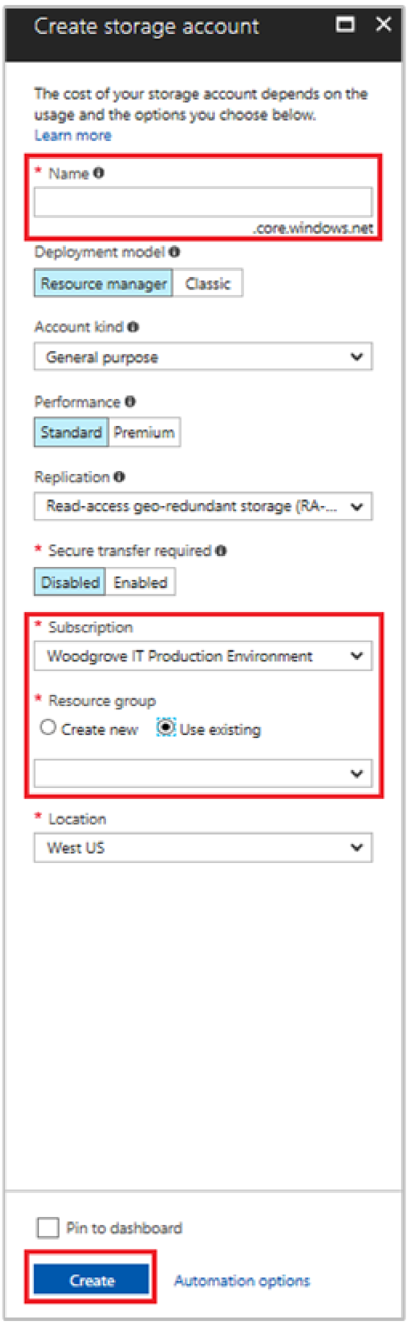
Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Create a storage account

If you don't already have one, you will now create a storage account. You can also skip this step and grant your VM system-assigned managed identity access to the keys of an existing storage account.

1. Click the **+/Create new service** button found on the upper left-hand corner of the Azure portal.
2. Click **Storage**, then **Storage Account**, and a new "Create storage account" panel will display.
3. Enter a **Name** for the storage account, which you will use later.
4. **Deployment model** and **Account kind** should be set to "Resource manager" and "General purpose", respectively.
5. Ensure the **Subscription** and **Resource Group** match the ones you specified when you created your VM in the previous step.
6. Click **Create**.



Create storage account

The cost of your storage account depends on the usage and the options you choose below.
Learn more

* Name
.core.windows.net

Deployment model
[✓ Resource manager] Classic

Account kind
General purpose

Performance
[✓ Standard] Premium

Replication
Read-access geo-redundant storage (RA-...

* Secure transfer required
[✓ Disabled] Enabled

* Subscription
Woodgrove IT Production Environment

* Resource group
Create new [✓ Use existing]

* Location
West US

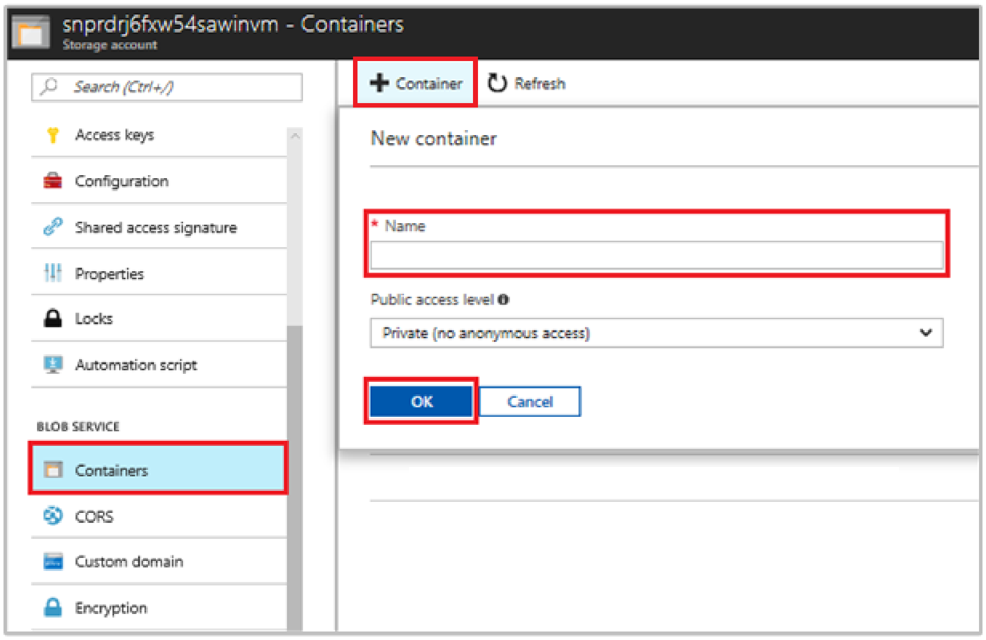
Pin to dashboard

Create Automation options

Create a blob container in the storage account

Later we will upload and download a file to the new storage account. Because files require blob storage, we need to create a blob container in which to store the file.

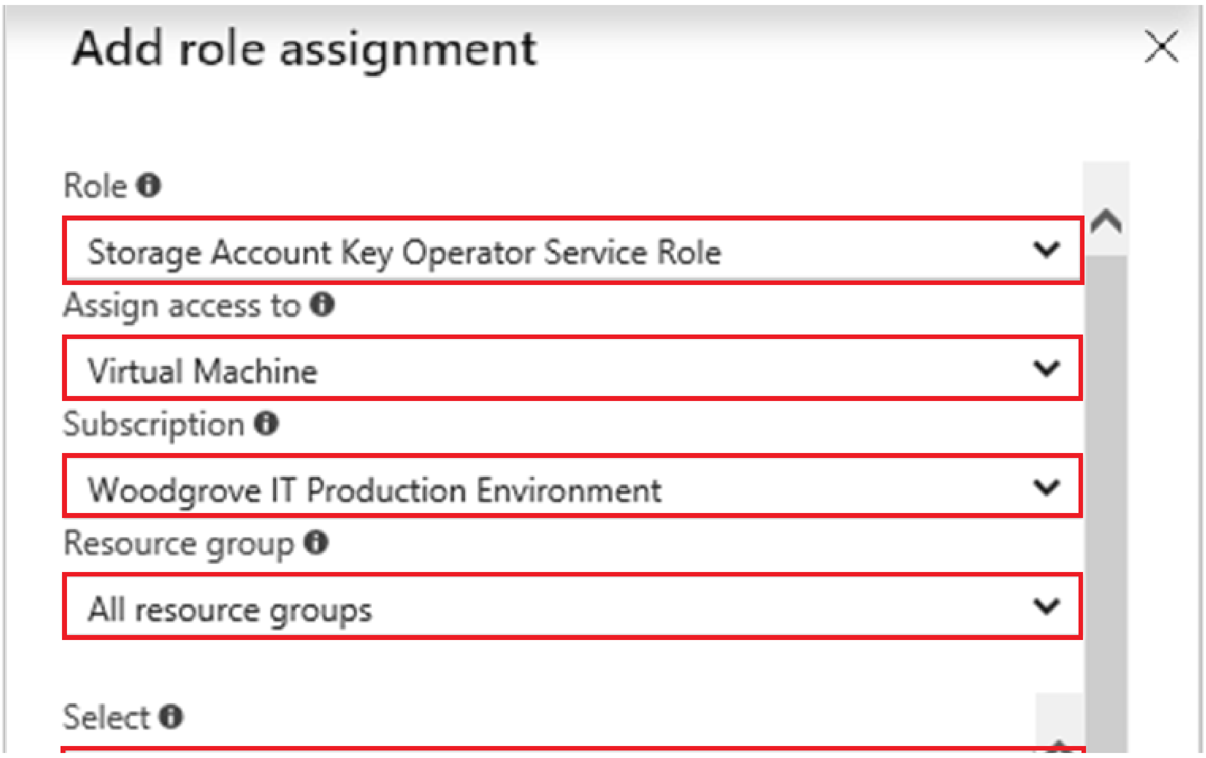
1. Navigate back to your newly created storage account.
2. Click the **Containers** link in the left, under "Blob service."
3. Click **+ Container** on the top of the page, and a "New container" panel slides out.
4. Give the container a name, select an access level, then click **OK**. The name you specified will be used later in the tutorial.



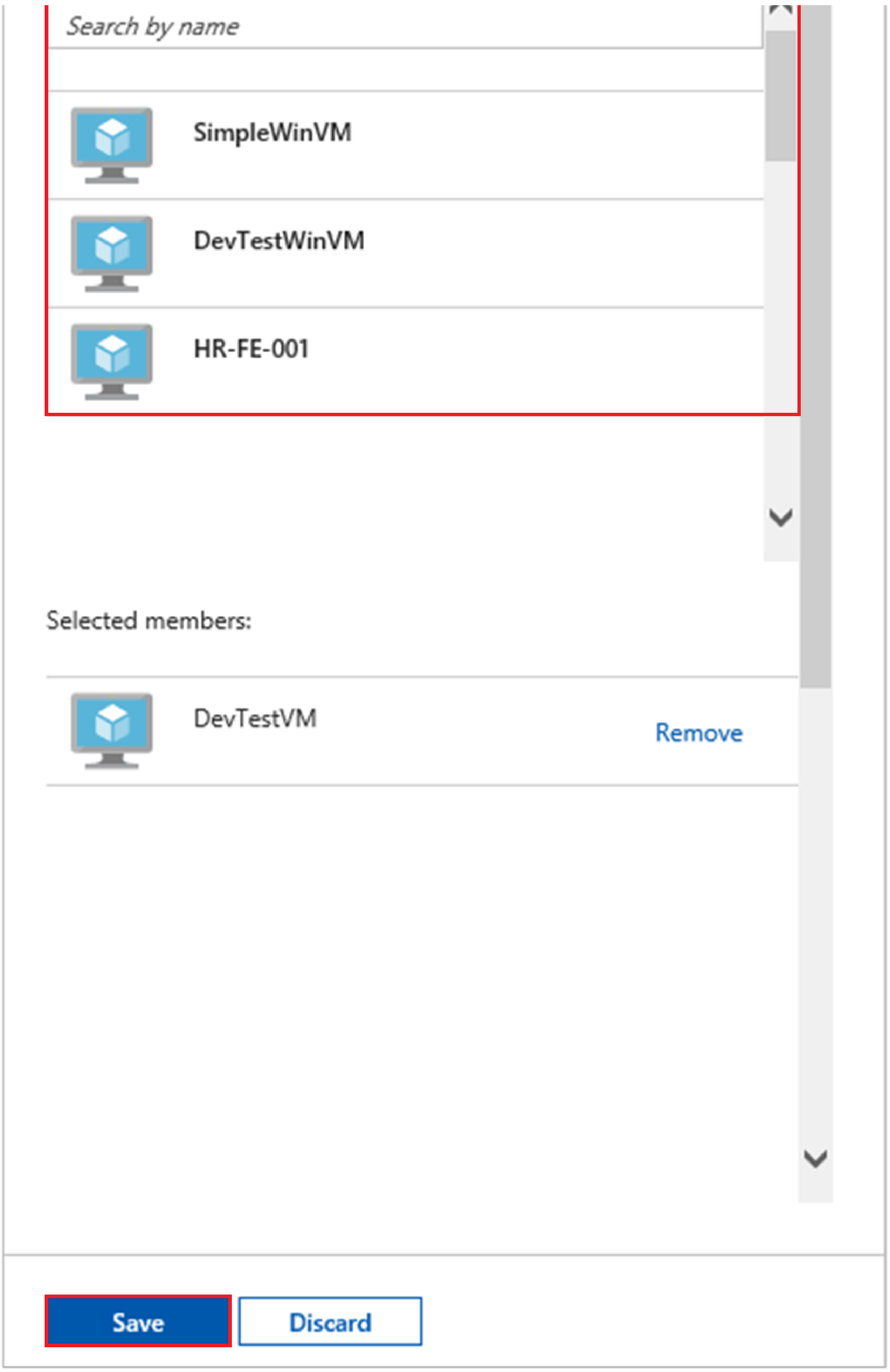
Grant your VM's system-assigned managed identity access to use storage account access keys

In this step, you grant your VM's system-assigned managed identity access to the keys to your storage account.

1. Navigate back to your newly created storage account.
2. Click the **Access control (IAM)** link in the left panel.
3. Click **+ Add role assignment** on top of the page to add a new role assignment for your VM
4. Set **Role** to "Storage Account Key Operator Service Role", on the right side of the page.
5. In the next dropdown, set **Assign access to** the resource "Virtual Machine".
6. Next, ensure the proper subscription is listed in **Subscription** dropdown, then set **Resource Group** to "All resource groups".
7. Finally, under **Select** choose your Linux Virtual Machine in the dropdown, then click **Save**.



Role: Storage Account Key Operator Service Role
Assign access to: Virtual Machine
Subscription: Woodgrove IT Production Environment
Resource group: All resource groups
Select



Get an access token using the VM's identity and use it to call Azure Resource Manager

For the remainder of the tutorial, we will work from the VM we created earlier.

To complete these steps, you will need an SSH client. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to Use SSH keys with Windows on Azure, or How to create and use an SSH public and private key pair for Linux VMs in Azure.

1. In the Azure portal, navigate to **Virtual Machines**, go to your Linux virtual machine, then from the **Overview** page click **Connect** at the top. Copy the string to connect to your VM.

2. Connect to your VM using your SSH client.

3. Next, you will be prompted to enter in your **Password** you added when creating the **Linux VM**. You should then be successfully signed in.

4. Use CURL to get an access token for Azure Resource Manager.

   The CURL request and response for the access token is below:

   ```
   curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -H Metadata:true
   ```

   NOTE

   In the previous request, the value of the "resource" parameter must be an exact match for what is expected by Azure AD. When using the Azure Resource Manager resource ID, you must include the trailing slash on the URI. In the following response, the access_token element as been shortened for brevity.

   ```
   {"access_token":"eyJ0eXAiOiJ...",
   "refresh_token":"",
   "expires_in":"3599",
   "expires_on":"1504130527",
   "not_before":"1504126627",
   "resource":"https://management.azure.com",
   "token_type":"Bearer"}
   ```

Get storage account access keys from Azure Resource Manager to make storage calls

Now use CURL to call Resource Manager using the access token we retrieved in the previous section, to retrieve the storage access key. Once we have the storage access key, we can call storage upload/download operations. Be sure to replace the `<SUBSCRIPTION ID>`, `<RESOURCE GROUP>`, and `<STORAGE ACCOUNT NAME>` parameter values with your own values. Replace the `<ACCESS TOKEN>` value with the access token you retrieved earlier:

```
curl https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE GROUP>/providers/Microsoft.Storage/storageAccounts/<STORAGE ACCOUNT NAME>/listKeys?api-version=2016-12-01 --request POST -d "" -H "Authorization: Bearer <ACCESS TOKEN>"
```

NOTE

The text in the prior URL is case sensitive, so ensure if you are using upper-lowercase for your Resource Groups to reflect it accordingly. Additionally, it’s important to know that this is a POST request not a GET request and ensure you pass a value to capture a length limit with -d that can be NULL.

The CURL response gives you the list of Keys:

```
{"keys":[{"keyName":"key1","permissions":"Full","value":"iqDPNt..."},
{"keyName":"key2","permissions":"Full","value":"U+uI0B..."}]}
```

Create a sample blob file to upload to your blob storage container. On a Linux VM you can do this with the following command.

```
echo "This is a test file." > test.txt
```

Next, authenticate with the CLI `az storage` command using the storage access key, and upload the file to the blob container. For this step, you will need to install the latest Azure CLI on your VM, if you haven't already.

```
az storage blob upload -c <CONTAINER NAME> -n test.txt -f test.txt --account-name <STORAGE ACCOUNT NAME> --
account-key <STORAGE ACCOUNT KEY>
```

Response:

```
Finished[#############################################################]  100.0000%
{
  "etag": "\"0x8D4F9929765C139\"",
  "lastModified": "2017-09-12T03:58:56+00:00"
}
```

Additionally, you can download the file using the Azure CLI and authenticating with the storage access key.

Request:

```
az storage blob download -c <CONTAINER NAME> -n test.txt -f test-download.txt --account-name <STORAGE ACCOUNT
NAME> --account-key <STORAGE ACCOUNT KEY>
```

Response:

```
{
  "content": null,
  "metadata": {},
  "name": "test.txt",
  "properties": {
    "appendBlobCommittedBlockCount": null,
    "blobType": "BlockBlob",
    "contentLength": 21,
    "contentRange": "bytes 0-20/21",
    "contentSettings": {
      "cacheControl": null,
      "contentDisposition": null,
      "contentEncoding": null,
      "contentLanguage": null,
      "contentMd5": "LSghAvpnElYyfUdn7CO8aw==",
      "contentType": "text/plain"
    },
    "copy": {
      "completionTime": null,
      "id": null,
      "progress": null,
      "source": null,
      "status": null,
      "statusDescription": null
    },
    "etag": "\"0x8D5067F30D0C283\"",
    "lastModified": "2017-09-28T14:42:49+00:00",
    "lease": {
      "duration": null,
      "state": "available",
      "status": "unlocked"
    },
    "pageBlobSequenceNumber": null,
    "serverEncrypted": false
  },
  "snapshot": null
}
```

Next steps

In this tutorial, you learned how to use a Linux VM system-assigned managed identity to access Azure Storage using an access key. To learn more about Azure Storage access keys see:

Manage your storage access keys

Tutorial: Use a Linux VM system-assigned identity to access Azure Storage via a SAS credential

4/5/2019 • 7 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Linux virtual machine (VM) to obtain a storage Shared Access Signature (SAS) credential. Specifically, a Service SAS credential.

NOTE

The SAS key generated in this tutorial will not be restricted/bound to the VM.

A Service SAS provides the ability to grant limited access to objects in a storage account, for a limited time and a specific service (in our case, the blob service), without exposing an account access key. You can use a SAS credential as usual when doing storage operations, for example when using the Storage SDK. For this tutorial, we demonstrate uploading and downloading a blob using Azure Storage CLI. You will learn how to:

- Create a storage account
- Create a blob container in the storage account
- Grant your VM access to a storage account SAS in Resource Manager
- Get an access token using your VM's identity, and use it to retrieve the SAS from Resource Manager

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Create a storage account

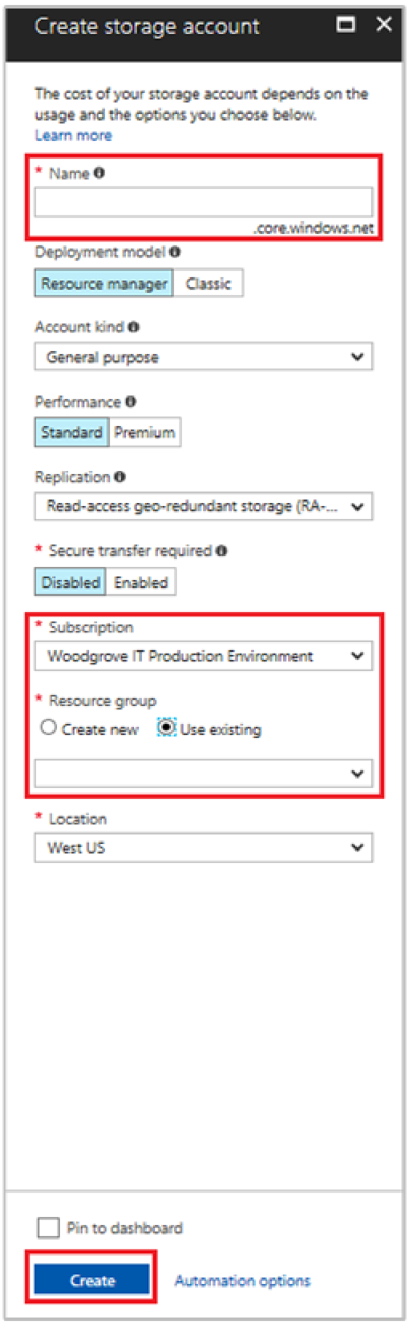
If you don't already have one, you will now create a storage account. You can also skip this step and grant your VM system-assigned managed identity access to the keys of an existing storage account.

1. Click the **+/Create new service** button found on the upper left-hand corner of the Azure portal.
2. Click **Storage**, then **Storage Account**, and a new "Create storage account" panel will display.
3. Enter a **Name** for the storage account, which you will use later.
4. **Deployment model** and **Account kind** should be set to "Resource manager" and "General purpose",

respectively.

5. Ensure the **Subscription** and **Resource Group** match the ones you specified when you created your VM in the previous step.

6. Click **Create**.



Create storage account

The cost of your storage account depends on the usage and the options you choose below.
Learn more

* Name
.core.windows.net

Deployment model: [✓ Resource manager] Classic

Account kind: General purpose

Performance: [✓ Standard] Premium

Replication: Read-access geo-redundant storage (RA-...

* Secure transfer required: [✓ Disabled] Enabled

* Subscription: Woodgrove IT Production Environment

* Resource group: Create new [✓ Use existing]

* Location: West US

Pin to dashboard

Create a blob container in the storage account

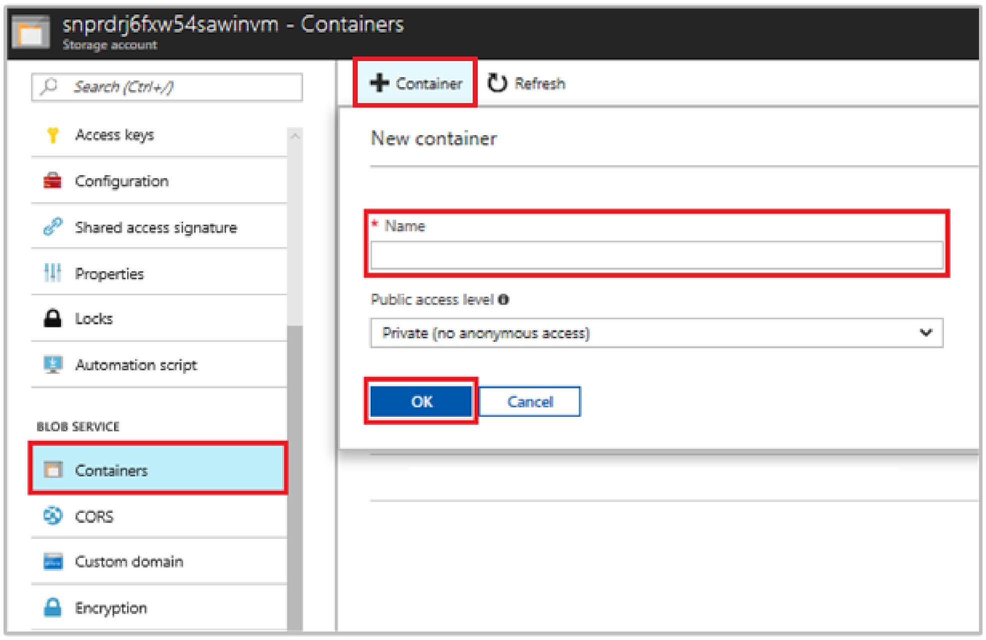
Later we will upload and download a file to the new storage account. Because files require blob storage, we need to create a blob container in which to store the file.

1. Navigate back to your newly created storage account.

2. Click the **Containers** link in the left panel, under "Blob service."

3. Click **+ Container** on the top of the page, and a "New container" panel slides out.

4. Give the container a name, select an access level, then click **OK**. The name you specified will be used later in the tutorial.


snprdrj6fxw54sawinvm - Containers
Storage account

New container

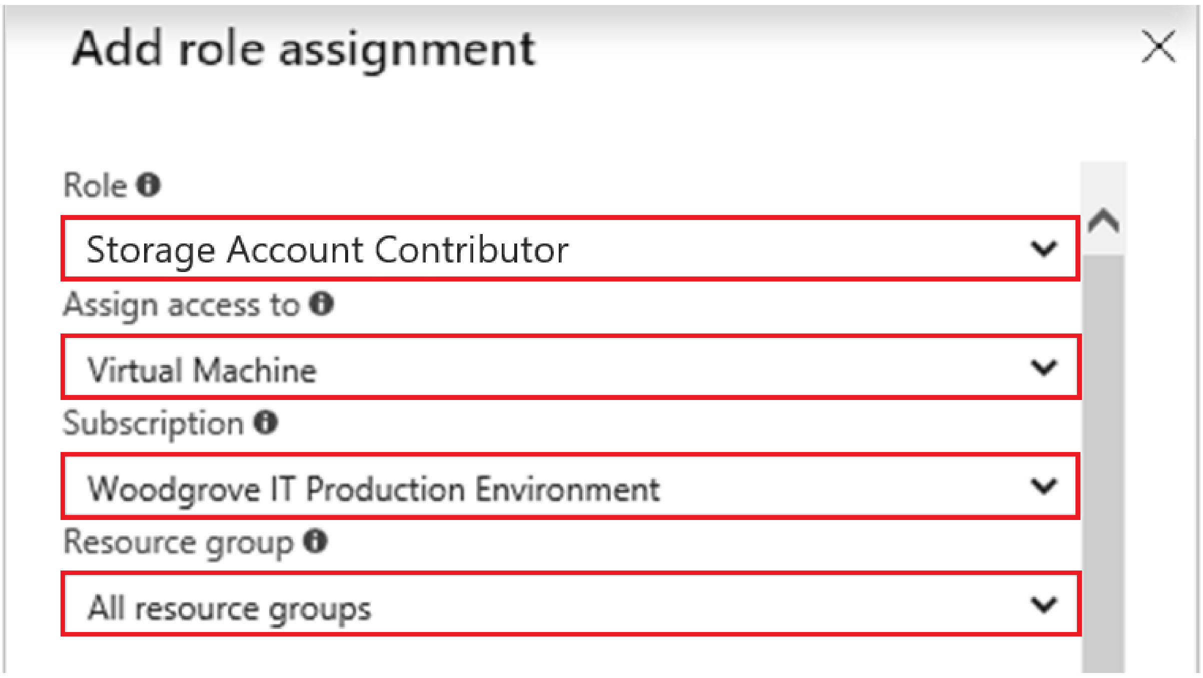
Name

Public access level: Private (no anonymous access)

Grant your VM's system-assigned managed identity access to use a storage SAS

Azure Storage does not natively support Azure AD authentication. However, you can use your VM's system-assigned managed identity to retrieve a storage SAS from the Resource Manager, then use the SAS to access storage. In this step, you grant your VM's system-assigned managed identity access to your storage account SAS.

1. Navigate back to your newly created storage account.
2. Click the **Access control (IAM)** link in the left panel.
3. Click **+ Add role assignment** on top of the page to add a new role assignment for your VM
4. Set **Role** to "Storage Account Contributor", on the right side of the page.
5. In the next dropdown, set **Assign access to** the resource "Virtual Machine".
6. Next, ensure the proper subscription is listed in **Subscription** dropdown, then set **Resource Group** to "All resource groups".
7. Finally, under **Select** choose your Linux Virtual Machine in the dropdown, then click **Save**.

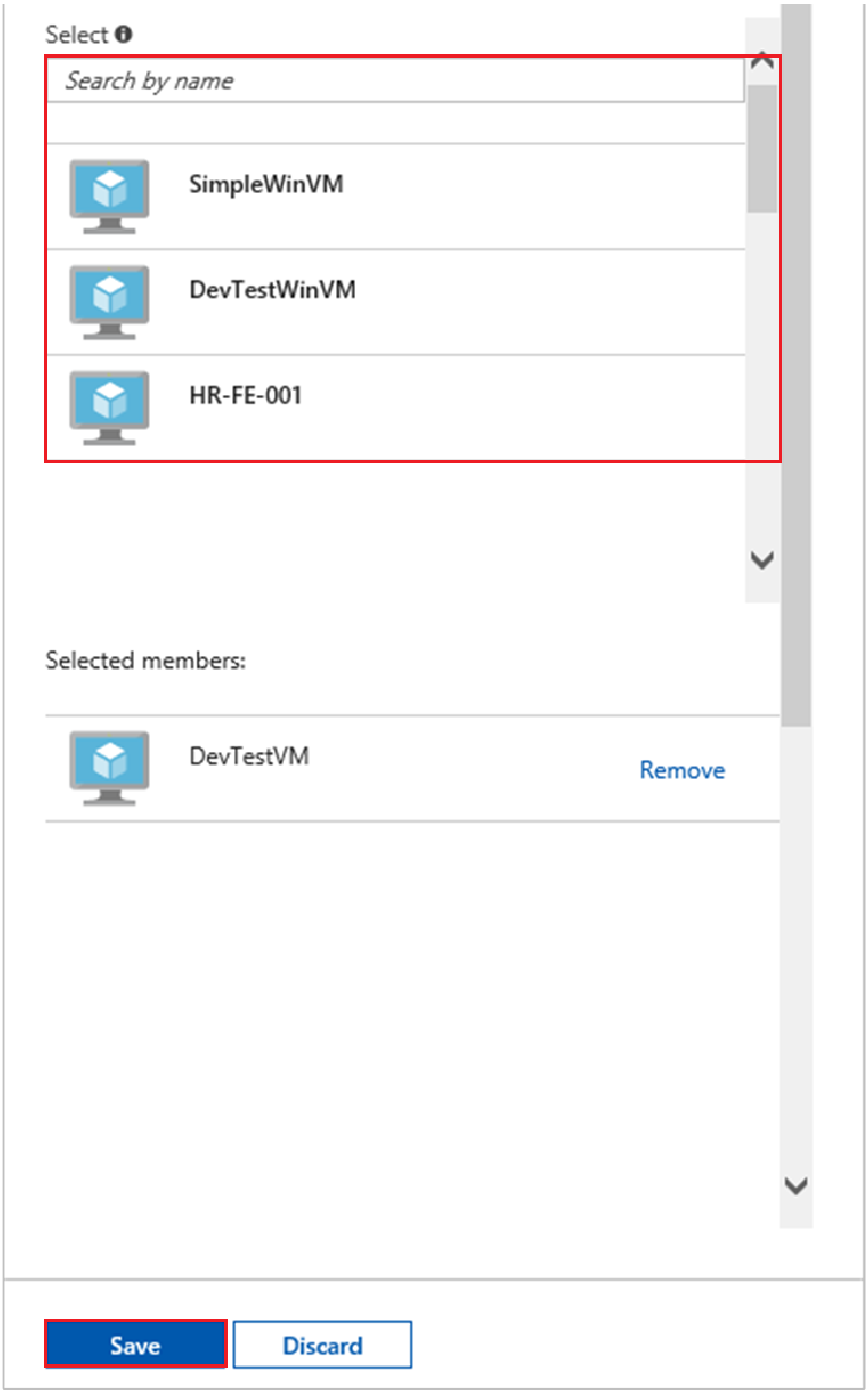

Add role assignment

Role: Storage Account Contributor

Assign access to: Virtual Machine

Subscription: Woodgrove IT Production Environment

Resource group: All resource groups



Get an access token using the VM's identity and use it to call Azure Resource Manager

For the remainder of the tutorial, we will work from the VM we created earlier.

To complete these steps, you will need an SSH client. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to Use SSH keys with Windows on Azure, or How to create and use an SSH public and private key pair for Linux VMs in Azure.

1. In the Azure portal, navigate to **Virtual Machines**, go to your Linux virtual machine, then from the **Overview** page click **Connect** at the top. Copy the string to connect to your VM.

2. Connect to your VM using your SSH client.

3. Next, you will be prompted to enter in your **Password** you added when creating the **Linux VM**. You should then be successfully signed in.

4. Use CURL to get an access token for Azure Resource Manager.

   The CURL request and response for the access token is below:

   ```
   curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
   01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -H Metadata:true
   ```

   > **NOTE**
   > In the previous request, the value of the "resource" parameter must be an exact match for what is expected by Azure AD. When using the Azure Resource Manager resource ID, you must include the trailing slash on the URI. In the following response, the access_token element as been shortened for brevity.

   ```
   {"access_token":"eyJ0eXAiOiJ...",
   "refresh_token":"",
   "expires_in":"3599",
   "expires_on":"1504130527",
   "not_before":"1504126627",
   "resource":"https://management.azure.com",
   "token_type":"Bearer"}
   ```

Get a SAS credential from Azure Resource Manager to make storage calls

Now use CURL to call Resource Manager using the access token we retrieved in the previous section, to create a storage SAS credential. Once we have the SAS credential, we can call storage upload/download operations.

For this request we'll use the follow HTTP request parameters to create the SAS credential:

```
{
    "canonicalizedResource":"/blob/<STORAGE ACCOUNT NAME>/<CONTAINER NAME>",
    "signedResource":"c",              // The kind of resource accessible with the SAS, in this case a
container (c).
    "signedPermission":"rcw",          // Permissions for this SAS, in this case (r)ead, (c)reate, and
(w)rite.  Order is important.
    "signedProtocol":"https",          // Require the SAS be used on https protocol.
    "signedExpiry":"<EXPIRATION TIME>" // UTC expiration time for SAS in ISO 8601 format, for example 2017-09-
22T00:06:00Z.
}
```

These parameters are included in the POST body of the request for the SAS credential. For more information on the parameters for creating a SAS credential, see the List Service SAS REST reference.

Use the following CURL request to get the SAS credential. Be sure to replace the `<SUBSCRIPTION ID>`, `<RESOURCE GROUP>`, `<STORAGE ACCOUNT NAME>`, `<CONTAINER NAME>`, and `<EXPIRATION TIME>` parameter values with your own values. Replace the `<ACCESS TOKEN>` value with the access token you retrieved earlier:

```
curl https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Storage/storageAccounts/<STORAGE ACCOUNT NAME>/listServiceSas/?api-version=2017-06-
01 -X POST -d "{\"canonicalizedResource\":\"/blob/<STORAGE ACCOUNT NAME>/<CONTAINER
NAME>\",\"signedResource\":\"c\",\"signedPermission\":\"rcw\",\"signedProtocol\":\"https\",\"signedExpiry\":\"
<EXPIRATION TIME>\"}" -H "Authorization: Bearer <ACCESS TOKEN>"
```

NOTE

The text in the prior URL is case sensitive, so ensure if you are using upper-lowercase for your Resource Groups to reflect it accordingly. Additionally, it's important to know that this is a POST request not a GET request.

The CURL response returns the SAS credential:

```
{"serviceSasToken":"sv=2015-04-05&sr=c&spr=https&st=2017-09-22T00%3A10%3A00Z&se=2017-09-
22T02%3A00%3A00Z&sp=rcw&sig=QcVwljccgWcNMbe9roAJbD8J5oEkYoq%2F0cUPlgriBn0%3D"}
```

Create a sample blob file to upload to your blob storage container. On a Linux VM you can do this with the following command.

```
echo "This is a test file." > test.txt
```

Next, authenticate with the CLI `az storage` command using the SAS credential, and upload the file to the blob container. For this step, you will need to install the latest Azure CLI on your VM, if you haven't already.

```
 az storage blob upload --container-name
                        --file
                        --name
                        --account-name
                        --sas-token
```

Response:

```
Finished[#############################################################]  100.0000%
{
  "etag": "\"0x8D4F9929765C139\"",
  "lastModified": "2017-09-21T03:58:56+00:00"
}
```

Additionally, you can download the file using the Azure CLI and authenticating with the SAS credential.

Request:

```
az storage blob download --container-name
                         --file
                         --name
                         --account-name
                         --sas-token
```

Response:

```
{
  "content": null,
  "metadata": {},
  "name": "testblob",
  "properties": {
    "appendBlobCommittedBlockCount": null,
    "blobType": "BlockBlob",
    "contentLength": 16,
    "contentRange": "bytes 0-15/16",
    "contentSettings": {
      "cacheControl": null,
      "contentDisposition": null,
      "contentEncoding": null,
      "contentLanguage": null,
      "contentMd5": "Aryr///Rb+D8JQ8IytleDA==",
      "contentType": "text/plain"
    },
    "copy": {
      "completionTime": null,
      "id": null,
      "progress": null,
      "source": null,
      "status": null,
      "statusDescription": null
    },
    "etag": "\"0x8D4F9929765C139\"",
    "lastModified": "2017-09-21T03:58:56+00:00",
    "lease": {
      "duration": null,
      "state": "available",
      "status": "unlocked"
    },
    "pageBlobSequenceNumber": null,
    "serverEncrypted": false
  },
  "snapshot": null
}
```

Next steps

In this tutorial, you learned how to use a Linux VM system-assigned managed identity to access Azure Storage using a SAS credential. To learn more about Azure Storage SAS see:

Using shared access signatures (SAS)

Tutorial: Use a Linux VM system-assigned managed identity to access Azure Key Vault

3/26/2019 • 4 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Linux virtual machine (VM) to access Azure Key Vault. Serving as a bootstrap, Key Vault makes it possible for your client application to then use the secret to access resources not secured by Azure Active Directory (AD). Managed identities for Azure resources are automatically managed by Azure and enable you to authenticate to services that support Azure AD authentication, without needing to insert credentials into your code.

You learn how to:

- Grant your VM access to a secret stored in a Key Vault
- Get an access token using the VM's identity and use it to retrieve the secret from the Key Vault

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

Grant your VM access to a Secret stored in a Key Vault

Using managed service identities for Azure resources your code can get access tokens to authenticate to resources that support Azure Active Directory authentication.However, not all Azure services support Azure AD authentication. To use managed identities for Azure resources with those services, store the service credentials in Azure Key Vault, and use managed identities for Azure resources to access Key Vault to retrieve the credentials.

First, we need to create a Key Vault and grant our VM’s system-assigned managed identity access to the Key Vault.

1. At the top of the left navigation bar, select **Create a resource** > **Security + Identity** > **Key Vault**.

2. Provide a **Name** for the new Key Vault.

3. Locate the Key Vault in the same subscription and resource group as the VM you created earlier.

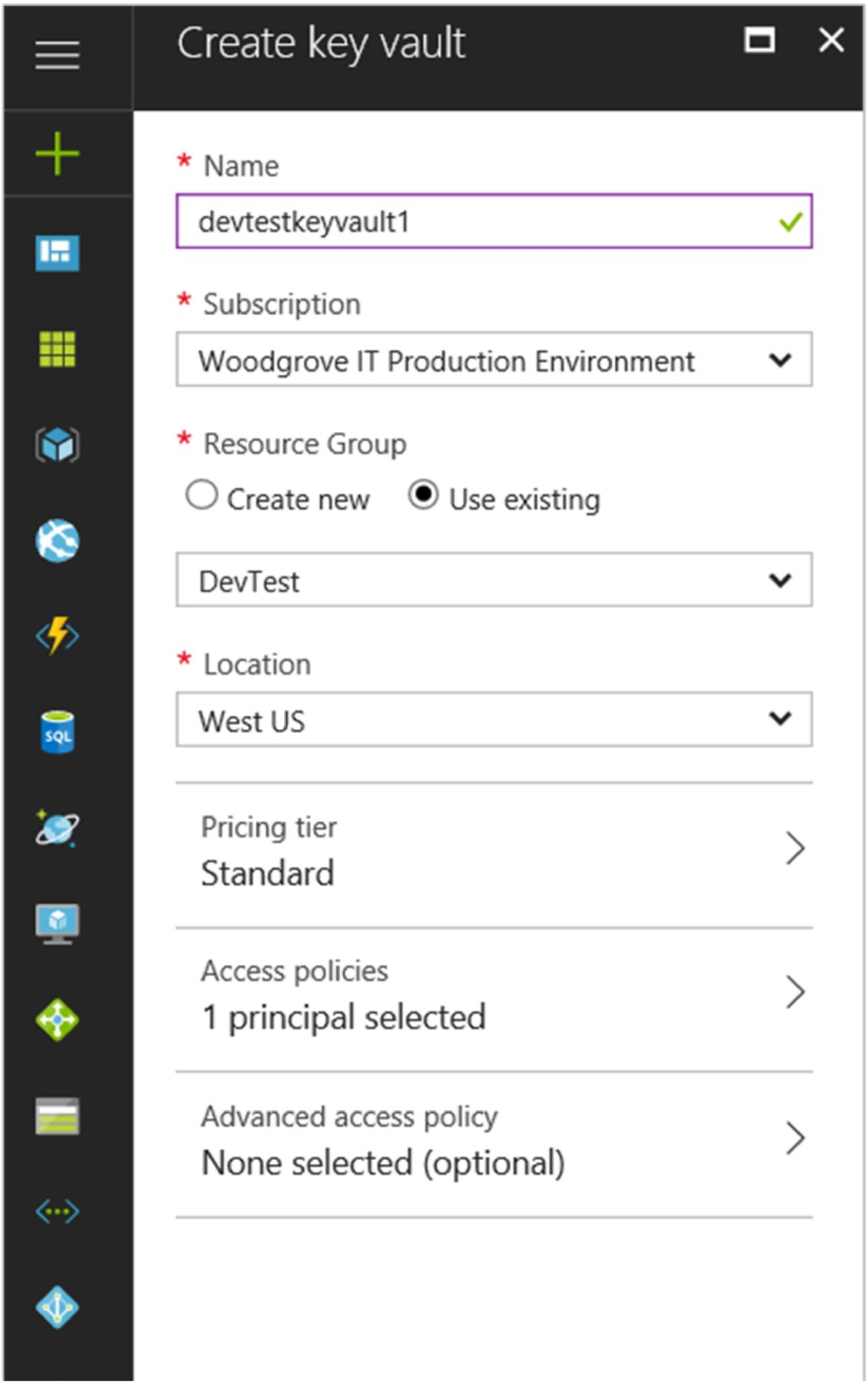
4. Select **Access policies** and click **Add new**.

5. In Configure from template, select **Secret Management**.

6. Choose **Select Principal**, and in the search field enter the name of the VM you created earlier. Select the

VM in the result list and click **Select**.

7. Click **OK** to finishing adding the new access policy, and **OK** to finish access policy selection.
8. Click **Create** to finish creating the Key Vault.



Create key vault

* Name
devtestkeyvault1 ✓

* Subscription
Woodgrove IT Production Environment

* Resource Group
Create new [✓ Use existing]

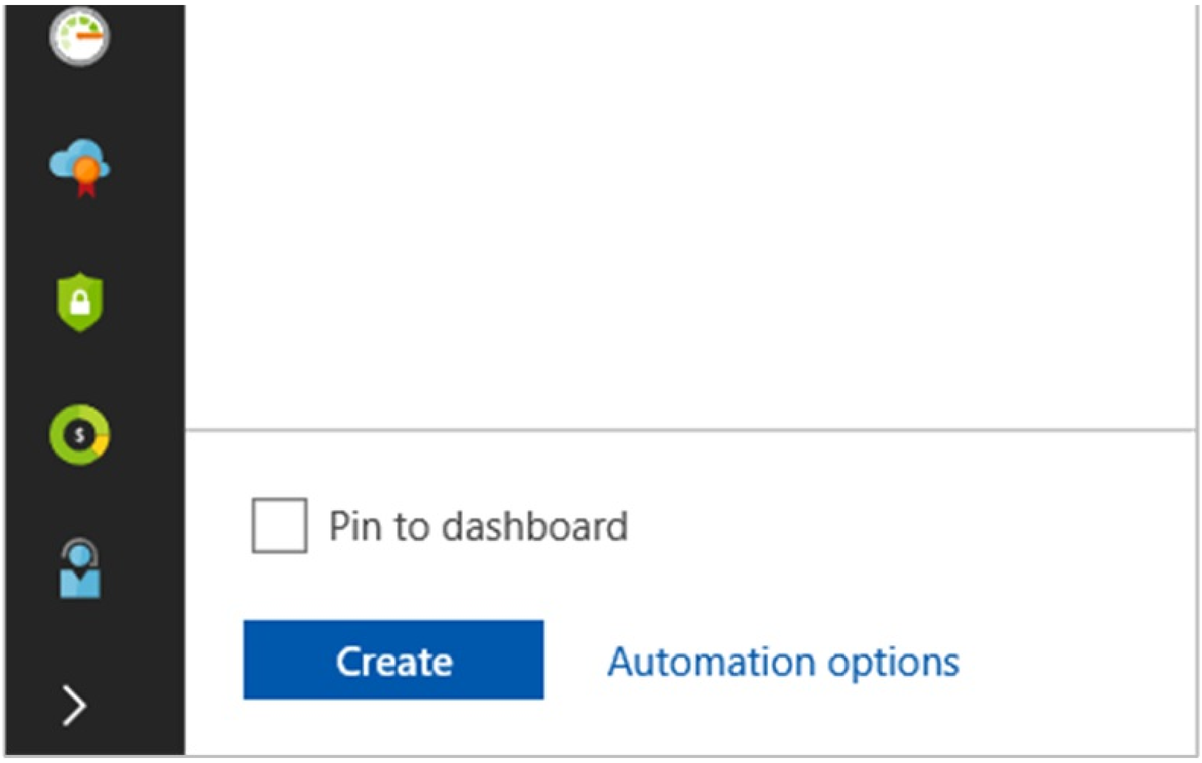
DevTest

* Location
West US

Pricing tier
Standard

Access policies
1 principal selected

Advanced access policy
None selected (optional)



Next, add a secret to the Key Vault, so that later you can retrieve the secret using code running in your VM:

1. Select **All Resources**, and find and select the Key Vault you created.
2. Select **Secrets**, and click **Add**.
3. Select **Manual**, from **Upload options**.
4. Enter a name and value for the secret. The value can be anything you want.
5. Leave the activation date and expiration date clear, and leave **Enabled** as **Yes**.
6. Click **Create** to create the secret.

Get an access token using the VM's identity and use it to retrieve the secret from the Key Vault

To complete these steps, you need an SSH client. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to Use SSH keys with Windows on Azure, or How to create and use an SSH public and private key pair for Linux VMs in Azure.

1. In the portal, navigate to your Linux VM and in the **Overview**, click **Connect**.
2. **Connect** to the VM with the SSH client of your choice.
3. In the terminal window, using CURL, make a request to the local managed identities for Azure resources endpoint to get an access token for Azure Key Vault. The CURL request for the access token is below.

   ```
   curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
   01&resource=https%3A%2F%2Fvault.azure.net' -H Metadata:true
   ```

   The response includes the access token you need to access Resource Manager.

   Response:

```
{"access_token":"eyJ0eXAi...",
"refresh_token":"",
"expires_in":"3599",
"expires_on":"1504130527",
"not_before":"1504126627",
"resource":"https://vault.azure.net",
"token_type":"Bearer"}
```

You can use this access token to authenticate to Azure Key Vault. The next CURL request shows how to read a secret from Key Vault using CURL and the Key Vault REST API. You'll need the URL of your Key Vault, which is in the **Essentials** section of the **Overview** page of the Key Vault. You will also need the access token you obtained on the previous call.

```
curl https://<YOUR-KEY-VAULT-URL>/secrets/<secret-name>?api-version=2016-10-01 -H "Authorization: Bearer
<ACCESS TOKEN>"
```

The response will look like this:

```
{"value":"p@ssw0rd!","id":"https://mytestkeyvault.vault.azure.net/secrets/MyTestSecret/7c2204c6093c4d859
bc5b9eff8f29050","attributes":
{"enabled":true,"created":1505088747,"updated":1505088747,"recoveryLevel":"Purgeable"}}
```

Once you've retrieved the secret from the Key Vault, you can use it to authenticate to a service that requires a name and password.

Next steps

In this tutorial, you learned how to use a Linux VM system-assigned managed identity to access Azure Key Vault. To learn more about Azure Key Vault see:

Azure Key Vault

Tutorial: Use a Linux VM system-assigned managed identity to access Azure Cosmos DB

4/11/2019 • 7 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Linux virtual machine (VM) to access Azure Cosmos DB. You learn how to:

- Create a Cosmos DB account
- Create a collection in the Cosmos DB account
- Grant the system-assigned managed identity access to an Azure Cosmos DB instance
- Retrieve the `principalID` of the of the Linux VM's system-assigned managed identity
- Get an access token and use it to call Azure Resource Manager
- Get access keys from Azure Resource Manager to make Cosmos DB calls

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled

To run the CLI script examples in this tutorial, you have two options:

- Use Azure Cloud Shell either from the Azure portal, or via the **Try It** button, located in the top right corner of each code block.
- Install the latest version of CLI 2.0 (2.0.23 or later) if you prefer to use a local CLI console.

Create a Cosmos DB account

If you don't already have one, create a Cosmos DB account. You can skip this step and use an existing Cosmos DB account.

1. Click the **+/Create new service** button found on the upper left-hand corner of the Azure portal.
2. Click **Databases**, then **Azure Cosmos DB**, and a new "New account" panel displays.
3. Enter an **ID** for the Cosmos DB account, which you use later.
4. **API** should be set to "SQL." The approach described in this tutorial can be used with the other available API types, but the steps in this tutorial are for the SQL API.
5. Ensure the **Subscription** and **Resource Group** match the ones you specified when you created your VM in the previous step. Select a **Location** where Cosmos DB is available.

6. Click **Create**.

Create a collection in the Cosmos DB account

Next, add a data collection in the Cosmos DB account that you can query in later steps.

1. Navigate to your newly created Cosmos DB account.
2. On the **Overview** tab click the **+/Add Collection** button, and an "Add Collection" panel slides out.
3. Give the collection a database ID, collection ID, select a storage capacity, enter a partition key, enter a throughput value, then click **OK**. For this tutorial, it is sufficient to use "Test" as the database ID and collection ID, select a fixed storage capacity and lowest throughput (400 RU/s).

Retrieve the `principalID` of the Linux VM's system-assigned managed identity

To gain access to the Cosmos DB account access keys from the Resource Manager in the following section, you need to retrieve the `principalID` of the Linux VM's system-assigned managed identity. Be sure to replace the `<SUBSCRIPTION ID>`, `<RESOURCE GROUP>` (resource group in which you VM resides), and `<VM NAME>` parameter values with your own values.

```
az resource show --id /subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Compute/virtualMachines/<VM NAMe> --api-version 2017-12-01
```

The response includes the details of the system-assigned managed identity (note the principalID as it is used in the next section):

```
{
    "id": "/subscriptions/<SUBSCRIPTION ID>/<RESOURCE GROUP>/providers/Microsoft.Compute/virtualMachines/<VM
NAMe>",
  "identity": {
    "principalId": "6891c322-314a-4e85-b129-52cf2daf47bd",
    "tenantId": "733a8f0e-ec41-4e69-8ad8-971fc4b533f8",
    "type": "SystemAssigned"
 }
```

Grant your Linux VM's system-assigned identity access to the Cosmos DB account access keys

Cosmos DB does not natively support Azure AD authentication. However, you can use a managed identity to retrieve a Cosmos DB access key from the Resource Manager, then use the key to access Cosmos DB. In this step, you grant your system-assigned managed identity access to the keys to the Cosmos DB account.

To grant the system-assigned managed identity access to the Cosmos DB account in Azure Resource Manager using the Azure CLI, update the values for `<SUBSCRIPTION ID>`, `<RESOURCE GROUP>`, and `<COSMOS DB ACCOUNT NAME>` for your environment. Replace `<MI PRINCIPALID>` with the `principalId` property returned by the `az resource show` command in Retrieve the principalID of the Linux VM's MI. Cosmos DB supports two levels of granularity when using access keys: read/write access to the account, and read-only access to the account. Assign the `DocumentDB Account Contributor` role if you want to get read/write keys for the account, or assign the `Cosmos DB Account Reader Role` role if you want to get read-only keys for the account:

```
az role assignment create --assignee <MI PRINCIPALID> --role '<ROLE NAME>' --scope
"/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.DocumentDB/databaseAccounts/<COSMODS DB ACCOUNT NAME>"
```

The response includes the details for the role assignment created:

```
{
  "id": "/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.DocumentDB/databaseAccounts/<COSMOS DB
ACCOUNT>/providers/Microsoft.Authorization/roleAssignments/5b44e628-394e-4e7b-bbc3-d6cd4f28f15b",
  "name": "5b44e628-394e-4e7b-bbc3-d6cd4f28f15b",
  "properties": {
    "principalId": "c0833082-6cc3-4a26-a1b1-c4b5f90a981f",
    "roleDefinitionId": "/subscriptions/<SUBSCRIPTION
ID>/providers/Microsoft.Authorization/roleDefinitions/fbdf93bf-df7d-467e-a4d2-9458aa1360c8",
    "scope": "/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.DocumentDB/databaseAccounts/<COSMOS DB ACCOUNT>"
  },
  "resourceGroup": "<RESOURCE GROUP>",
  "type": "Microsoft.Authorization/roleAssignments"
}
```

Get an access token using the Linux VM's system-assigned managed identity and use it to call Azure Resource Manager

For the remainder of the tutorial, work from the VM created earlier.

To complete these steps, you need an SSH client. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to Use SSH keys with Windows on Azure, or How to create and use an SSH public and private key pair for Linux VMs in Azure.

1. In the Azure portal, navigate to **Virtual Machines**, go to your Linux virtual machine, then from the **Overview** page click **Connect** at the top. Copy the string to connect to your VM.

2. Connect to your VM using your SSH client.

3. Next, you are prompted to enter in your **Password** you added when creating the **Linux VM**. You should then be successfully signed in.

4. Use CURL to get an access token for Azure Resource Manager:

   ```
   curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
   01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -H Metadata:true
   ```

   NOTE

   In the previous request, the value of the "resource" parameter must be an exact match for what is expected by Azure AD. When using the Azure Resource Manager resource ID, you must include the trailing slash on the URI. In the following response, the access_token element as been shortened for brevity.

```
{"access_token":"eyJ0eXAiOi...",
 "expires_in":"3599",
 "expires_on":"1518503375",
 "not_before":"1518499475",
 "resource":"https://management.azure.com/",
 "token_type":"Bearer",
 "client_id":"1ef89848-e14b-465f-8780-bf541d325cd5"}
```

Get access keys from Azure Resource Manager to make Cosmos DB calls

Now use CURL to call Resource Manager using the access token retrieved in the previous section to retrieve the Cosmos DB account access key. Once we have the access key, we can query Cosmos DB. Be sure to replace the `<SUBSCRIPTION ID>`, `<RESOURCE GROUP>`, and `<COSMOS DB ACCOUNT NAME>` parameter values with your own values. Replace the `<ACCESS TOKEN>` value with the access token you retrieved earlier. If you want to retrieve read/write keys, use key operation type `listKeys`. If you want to retrieve read-only keys, use the key operation type `readonlykeys`:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.DocumentDB/databaseAccounts/<COSMOS DB ACCOUNT NAME>/<KEY OPERATION TYPE>?api-
version=2016-03-31' -X POST -d "" -H "Authorization: Bearer <ACCESS TOKEN>"
```

NOTE

The text in the prior URL is case-sensitive, so ensure if you are using upper-lowercase for your Resource Groups to reflect it accordingly. Additionally, it's important to know that this is a POST request not a GET request and ensure you pass a value to capture a length limit with -d that can be NULL.

The CURL response gives you the list of Keys. For example, if you get the read-only keys:

```
{"primaryReadonlyMasterKey":"bWpDxS...dzQ==",
"secondaryReadonlyMasterKey":"38v5ns...7bA=="}
```

Now that you have the access key for the Cosmos DB account you can pass it to a Cosmos DB SDK and make calls to access the account. For a quick example, you can pass the access key to the Azure CLI. You can get the `<COSMOS DB CONNECTION URL>` from the **Overview** tab on the Cosmos DB account blade in the Azure portal. Replace the `<ACCESS KEY>` with the value you obtained above:

```
az cosmosdb collection show -c <COLLECTION ID> -d <DATABASE ID> --url-connection "<COSMOS DB CONNECTION URL>" -
-key <ACCESS KEY>
```

This CLI command returns details about the collection:

```
{
  "collection": {
    "_conflicts": "conflicts/",
    "_docs": "docs/",
    "_etag": "\"00006700-0000-0000-0000-5a8271e90000\"",
    "_rid": "Es5SAM2FDwA=",
    "_self": "dbs/Es5SAA==/colls/Es5SAM2FDwA=/",
    "_sprocs": "sprocs/",
    "_triggers": "triggers/",
    "_ts": 1518498281,
    "_udfs": "udfs/",
    "id": "Test",
    "indexingPolicy": {
      "automatic": true,
      "excludedPaths": [],
      "includedPaths": [
        {
          "indexes": [
            {
              "dataType": "Number",
              "kind": "Range",
              "precision": -1
            },
            {
              "dataType": "String",
              "kind": "Range",
              "precision": -1
            },
            {
              "dataType": "Point",
              "kind": "Spatial"
            }
          ],
          "path": "/*"
        }
      ],
      "indexingMode": "consistent"
    }
  },
  "offer": {
    "_etag": "\"00006800-0000-0000-0000-5a8271ea0000\"",
    "_rid": "f4V+",
    "_self": "offers/f4V+/",
    "_ts": 1518498282,
    "content": {
      "offerIsRUPerMinuteThroughputEnabled": false,
      "offerThroughput": 400
    },
    "id": "f4V+",
    "offerResourceId": "Es5SAM2FDwA=",
    "offerType": "Invalid",
    "offerVersion": "V2",
    "resource": "dbs/Es5SAA==/colls/Es5SAM2FDwA=/"
  }
}
```

Next steps

In this tutorial, you learned how to use a system-assigned managed identity on a Linux virtual machine to access Cosmos DB. To learn more about Cosmos DB see:

Azure Cosmos DB overview

Tutorial: Use a Linux VM system-assigned managed identity to access Azure AD Graph API

3/26/2019 • 5 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This tutorial shows you how to use a system-assigned managed identity for a Linux virtual machine (VM) to access the Azure AD Graph API to retrieve its group memberships. Managed identities for Azure resources are automatically managed by Azure and enable you to authenticate to services that support Azure AD authentication without needing to insert credentials into your code.

For this tutorial, you will query your VM identity's membership in Azure AD groups. Group information is often used in authorization decisions. Under the covers, your VM's managed identity is represented by a **Service Principal** in Azure AD.

- Connect to Azure AD
- Add your VM identity to a group in Azure AD
- Grant the VM's identity access to Azure AD Graph
- Get an access token using the VM identity and use it to call Azure AD Graph

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- To perform the required resource creation and role management, your account needs "Owner" permissions at the appropriate scope (your subscription or resource group). If you need assistance with role assignment, see Use Role-Based Access Control to manage access to your Azure subscription resources.
- Sign in to Azure portal
- Create a virtual machine with system-assigned identity enabled
- Install the latest version of the Azure CLI
- To grant a VM identity access to Azure AD Graph, your account needs to be assigned the **Global Admin** role in Azure AD.

Connect to Azure AD

You need to connect to Azure AD to assign the VM to a group as well as grant the VM permission to retrieve its group memberships.

```
az login
```

Add your VM's identity to a group in Azure AD

When you enabled system-assigned managed identity on the Linux VM, it created a service principal in Azure AD. You need to add the VM to a group. Refer to the following article for instructions on how to add your VM to a group in Azure AD:

- Add group members

Grant your VM access to the Azure AD Graph API

Using managed identities for Azure resources, your code can get access tokens to authenticate to resources that support Azure AD authentication. The Microsoft Azure AD Graph API supports Azure AD authentication. In this step, you will grant your VM identity's service principal access to the Azure AD Graph so that it can query group memberships. Service principals are granted access to the Microsoft or Azure AD Graph through **Application Permissions**. The type of application permission you need to grant depends on the entity you want to access in the MS or Azure AD Graph.

For this tutorial, you will grant your VM identity the ability to query group memberships using the `Directory.Read.All` application permission. To grant this permission, you will need a user account that is assigned the Global Admin role in Azure AD. Normally you would grant an application permission by visiting your application's registration in the Azure portal and adding the permission there. However, managed identities for Azure resources does not register application objects in Azure AD, it only registers service principals. To register the application permission you will use the Azure AD PowerShell command line tool.

Azure AD Graph:

- Service Principal appId (used when granting app permission): 00000002-0000-0000-c000-000000000000
- Resource ID (used when requesting access token from managed identities for Azure resources): https://graph.windows.net
- Permission scope reference: Azure AD Graph Permissions Reference

Grant application permissions using CURL

1. Retrieve a token to make CURL requests:

```
az account get-access-token --resource "https://graph.windows.net/"
```

2. You will need to retrieve and note the `objectId` of your VM. It's used in subsequent steps to grant permissions to the VM to read its group membership. Replace `<ACCESS TOKEN>` with the access token you retrieved in the preceding step.

```
curl 'https://graph.windows.net/myorganization/servicePrincipals?
$filter=startswith%28displayName%2C%27myVM%27%29&api-version=1.6' -H "Authorization: Bearer <ACCESS
TOKEN>"
```

3. Using the Azure AD Graph appID, 00000002-0000-0000-c000-000000000000, retrieve and note the `objectId` for `odata.type: Microsoft.DirectoryServices.ServicePrincipal` and the `id` for the `Directory.Read.All` app role permission. Replace `<ACCESS TOKEN>` with the access token you retrieved earlier.

```
curl "https://graph.windows.net/myorganization/servicePrincipals?api-
version=1.6&%24filter=appId%20eq%20'00000002-0000-0000-c000-000000000000'" -H "Authorization: Bearer
<ACCESS TOKEN>"
```

Response:

```
"odata.metadata":"https://graph.windows.net/myorganization/$metadata#directoryObjects",
"value":[
   {
      "odata.type":"Microsoft.DirectoryServices.ServicePrincipal",
      "objectType":"ServicePrincipal",
      "objectId":"81789304-ff96-402b-ae73-07ec0db26721",
      "deletionTimestamp":null,
      "accountEnabled":true,
      "addIns":[
      ],
      "alternativeNames":[
      ],
      "appDisplayName":"Windows Azure Active Directory",
      "appId":"00000002-0000-0000-c000-000000000000",
      "appOwnerTenantId":"f8cdef31-a31e-4b4a-93e4-5f571e91255a",
      "appRoleAssignmentRequired":false,
      "appRoles":[
         {
            "allowedMemberTypes":[
               "Application"
            ],
            "description":"Allows the app to read data in your company or school directory, such as
users, groups, and apps.",
            "displayName":"Read directory data",
            "id":"5778995a-e1bf-45b8-affa-663a9f3f4d04",
            "isEnabled":true,
            "value":"Directory.Read.All"
         },
         {
            //other appRoles values
         }
```

4. Now, grant the VM's service principal read access to Azure AD directory objects using the Azure AD Graph API. The `id` value is the value for the `Directory.Read.All` app role permission and the `resourceId` is the `objectId` for the service principal `odata.type:Microsoft.DirectoryServices.ServicePrincipal` (the values you noted in the previous step).

```
curl "https://graph.windows.net/myorganization/servicePrincipals/<VM Object ID>/appRoleAssignments?api-
version=1.6" -X POST -d '{"id":"5778995a-e1bf-45b8-affa-663a9f3f4d04","principalId":"<VM Object
ID>","resourceId":"81789304-ff96-402b-ae73-07ec0db26721"}'-H "Content-Type: application/json" -H
"Authorization: Bearer <ACCESS TOKEN>"
```

Get an access token using the VM's identity to call Azure AD Graph

To complete these steps, you will need an SSH client. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to Use SSH keys with Windows on Azure, or How to create and use an SSH public and private key pair for Linux VMs in Azure.

1. In the portal, navigate to your Linux VM and in the **Overview**, click **Connect**.

2. **Connect** to the VM with the SSH client of your choice.

3. In the terminal window, using CURL, make a request to the local managed identities for Azure resources endpoint to get an access token for the Azure AD Graph.

```
curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
01&resource=https://graph.windows.net' -H Metadata:true
```

The response includes the access token you need to access Azure AD Graph.

Response:

```
{
    "access_token":"eyJ0eXAiOiJKV...",
    "expires_in":"3599",
    "expires_on":"1519622535",
    "not_before":"1519618635",
    "resource":"https://graph.windows.net",
    "token_type":"Bearer"
}
```

4. Using the object ID of your VM's service principal (the value you retrieved in earlier steps), you can query the Azure AD Graph API to retrieve its group memberships. Replace `<OBJECT-ID>` with the Object ID of your VM's service principal and `<ACCESS-TOKEN>` with the previously obtained access token:

```
curl 'https://graph.windows.net/myorganization/servicePrincipals/<OBJECT-ID>/getMemberGroups?api-
version=1.6' -X POST -d "{\"securityEnabledOnly\": false}" -H "Content-Type: application/json" -H
"Authorization: Bearer <ACCESS-TOKEN>"
```

Response:

```
Content :
{"odata.metadata":"https://graph.windows.net/myorganization/$metadata#Collection(Edm.String)","value":["
<ObjectID of VM's group membership>"]}
```

Next steps

In this tutorial, you learned how to use a Linux VM system-assigned managed identity to access Azure AD Graph. To learn more about Azure AD Graph see:

Azure AD Graph

Tutorial: Use a user-assigned managed identity on a Linux VM to access Azure Resource Manager

3/26/2019 • 5 minutes to read • Edit Online

User assigned managed identities are a preview feature of Azure Active Directory. Make sure you review the known issues before you begin. For more information about previews, see Supplemental Terms of Use for Microsoft Azure Previews.

This tutorial explains how to create a user-assigned managed identity, assign it to a Linux Virtual Machine (VM), and then use that identity to access the Azure Resource Manager API. Managed identities for Azure resources are automatically managed by Azure. They enable authentication to services that support Azure AD authentication, without needing to embed credentials into your code.

In this tutorial, you learn how to:

- Create a user-assigned managed identity
- Assign the user-assigned managed identity to a Linux VM
- Grant the user-assigned managed identity access to a Resource Group in Azure Resource Manager
- Get an access token using the user-assigned managed identity and use it to call Azure Resource Manager

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.
- Sign in to Azure portal
- Create a Linux virtual machine
- If you choose to install and use the CLI locally, this quickstart requires that you are running the Azure CLI version 2.0.4 or later. Run `az --version` to find the version. If you need to install or upgrade, see Install Azure CLI 2.0.

Create a user-assigned managed identity

1. If you are using the CLI console (instead of an Azure Cloud Shell session), start by signing in to Azure. Use an account that is associated with the Azure subscription under which you would like to create the new user-assigned managed identity:

   ```
   az login
   ```

2. Create a user-assigned managed identity using az identity create. The `-g` parameter specifies the resource group where the user-assigned managed identity is created, and the `-n` parameter specifies its name. Be sure to replace the `<RESOURCE GROUP>` and `<UAMI NAME>` parameter values with your own values:

IMPORTANT

When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

```
az identity create -g <RESOURCE GROUP> -n <UAMI NAME>
```

The response contains details for the user-assigned managed identity created, similar to the following example. Note the `id` value for your user-assigned managed identity, as it will be used in the next step:

```
{
"clientId": "73444643-8088-4d70-9532-c3a0fdc190fz",
"clientSecretUrl": "https://control-westcentralus.identity.azure.net/subscriptions/<SUBSCRIPTON
ID>/resourcegroups/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<UAMI
NAME>/credentials?tid=5678&oid=9012&aid=12344643-8088-4d70-9532-c3a0fdc190fz",
"id": "/subscriptions/<SUBSCRIPTON ID>/resourcegroups/<RESOURCE
GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<UAMI NAME>",
"location": "westcentralus",
"name": "<UAMI NAME>",
"principalId": "9012",
"resourceGroup": "<RESOURCE GROUP>",
"tags": {},
"tenantId": "733a8f0e-ec41-4e69-8ad8-971fc4b533bl",
"type": "Microsoft.ManagedIdentity/userAssignedIdentities"
}
```

Assign a user-assigned managed identity to your Linux VM

A user-assigned managed identity can be used by clients on multiple Azure resources. Use the following commands to assign the user-assigned managed identity to a single VM. Use the `Id` property returned in the previous step for the `-IdentityID` parameter.

Assign the user-assigned managed identity to your Linux VM using az vm identity assign. Be sure to replace the `<RESOURCE GROUP>` and `<VM NAME>` parameter values with your own values. Use the `id` property returned in the previous step for the `--identities` parameter value.

```
az vm identity assign -g <RESOURCE GROUP> -n <VM NAME> --identities "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<UAMI NAME>"
```

Grant your user-assigned managed identity access to a Resource Group in Azure Resource Manager

Managed identities for Azure resources provides identities that your code can use to request access tokens to authenticate to resource APIs that support Azure AD authentication. In this tutorial, your code will access the Azure Resource Manager API.

Before your code can access the API, you need to grant the identity access to a resource in Azure Resource Manager. In this case, the Resource Group in which the VM is contained. Update the value for `<SUBSCRIPTION ID>` and `<RESOURCE GROUP>` as appropriate for your environment. Additionally, replace `<UAMI PRINCIPALID>` with the `principalId` property returned by the `az identity create` command in Create a user-assigned managed identity:

```
az role assignment create --assignee <UAMI PRINCIPALID> --role 'Reader' --scope "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/<RESOURCE GROUP> "
```

The response contains details for the role assignment created, similar to the following example:

```
{
  "id": "/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Authorization/roleAssignments/b402bd74-157f-425c-bf7d-zed3a3a581ll",
  "name": "b402bd74-157f-425c-bf7d-zed3a3a581ll",
  "properties": {
    "principalId": "f5fdfdc1-ed84-4d48-8551-999fb9dedfbl",
    "roleDefinitionId": "/subscriptions/<SUBSCRIPTION
ID>/providers/Microsoft.Authorization/roleDefinitions/acdd72a7-3385-48ef-bd42-f606fba81ae7",
    "scope": "/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE GROUP>"
  },
  "resourceGroup": "<RESOURCE GROUP>",
  "type": "Microsoft.Authorization/roleAssignments"
}
```

Get an access token using the VM's identity and use it to call Resource Manager

For the remainder of the tutorial, we will work from the VM we created earlier.

To complete these steps, you need an SSH client. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux.

1. Sign in to the Azure portal.

2. In the portal, navigate to **Virtual Machines** and go to the Linux virtual machine and in the **Overview**, click **Connect**. Copy the string to connect to your VM.

3. Connect to the VM with the SSH client of your choice. If you are using Windows, you can use the SSH client in the Windows Subsystem for Linux. If you need assistance configuring your SSH client's keys, see How to Use SSH keys with Windows on Azure, or How to create and use an SSH public and private key pair for Linux VMs in Azure.

4. In the terminal window, using CURL, make a request to the Azure Instance Metadata Service (IMDS) identity endpoint to get an access token for Azure Resource Manager.

   The CURL request to acquire an access token is shown in the following example. Be sure to replace `<CLIENT ID>` with the `clientId` property returned by the `az identity create` command in Create a user-assigned managed identity:

   ```
   curl -H Metadata:true "http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
   01&resource=https%3A%2F%2Fmanagement.azure.com/&client_id=<UAMI CLIENT ID>"
   ```

   NOTE

   The value of the `resource` parameter must be an exact match for what is expected by Azure AD. When using the Resource Manager resource ID, you must include the trailing slash on the URI.

   The response includes the access token you need to access Azure Resource Manager.

   Response example:

```
{
"access_token":"eyJ0eXAiOi...",
"refresh_token":"",
"expires_in":"3599",
"expires_on":"1504130527",
"not_before":"1504126627",
"resource":"https://management.azure.com",
"token_type":"Bearer"
}
```

5. Use the access token to access Azure Resource Manager, and read the properties of the Resource Group to which you previously granted your user-assigned managed identity access. Be sure to replace `<SUBSCRIPTION ID>`, `<RESOURCE GROUP>` with the values you specified earlier, and `<ACCESS TOKEN>` with the token returned in the previous step.

   NOTE

   The URL is case-sensitive, so be sure to use the exact same case you used earlier when you named the Resource Group, and the uppercase "G" in `resourceGroups`.

   ```
   curl https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE GROUP>?api-version=2016-09-01 -H "Authorization: Bearer <ACCESS TOKEN>"
   ```

   The response contains the specific Resource Group information, similar to the following example:

   ```
   {
   "id":"/subscriptions/<SUBSCRIPTION ID>/resourceGroups/DevTest",
   "name":"DevTest",
   "location":"westus",
   "properties":{"provisioningState":"Succeeded"}
   }
   ```

Next steps

In this tutorial, you learned how to create a user-assigned managed identity and attach it to a Linux virtual machine to access the Azure Resource Manager API. To learn more about Azure Resource Manager see:

Azure Resource Manager

Configure managed identities for Azure resources on a VM using the Azure portal

3/26/2019 • 3 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, you learn how to enable and disable system and user-assigned managed identities for an Azure Virtual Machine (VM), using the Azure portal.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section.
- If you don't already have an Azure account, sign up for a free account before continuing.

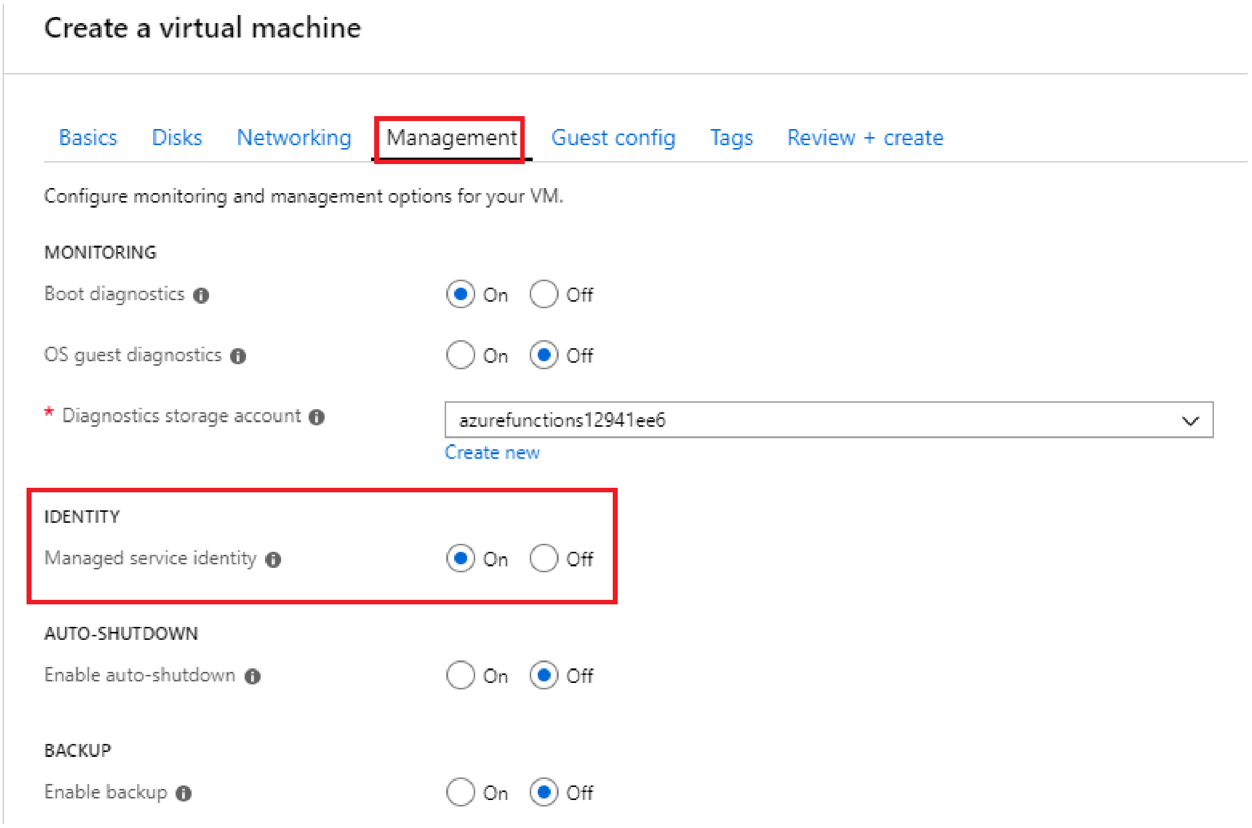
System-assigned managed identity

In this section, you learn how to enable and disable the system-assigned managed identity for VM using the Azure portal.

Enable system-assigned managed identity during creation of a VM

To enable system-assigned managed identity on a VM during its creation, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

- Under the **Management** tab in the **Identity** section, switch **Managed service identity** to **On**.


Create a virtual machine

Basics | Disks | Networking | Management | Guest config | Tags | Review + create

Configure monitoring and management options for your VM.

MONITORING

Boot diagnostics: [✓ On] Off

OS guest diagnostics: On [✓ Off]

* Diagnostics storage account: azurefunctions12941ee6

Create new

IDENTITY

Managed service identity: [✓ On] Off

AUTO-SHUTDOWN

Enable auto-shutdown: On [✓ Off]

BACKUP

Enable backup: On [✓ Off]

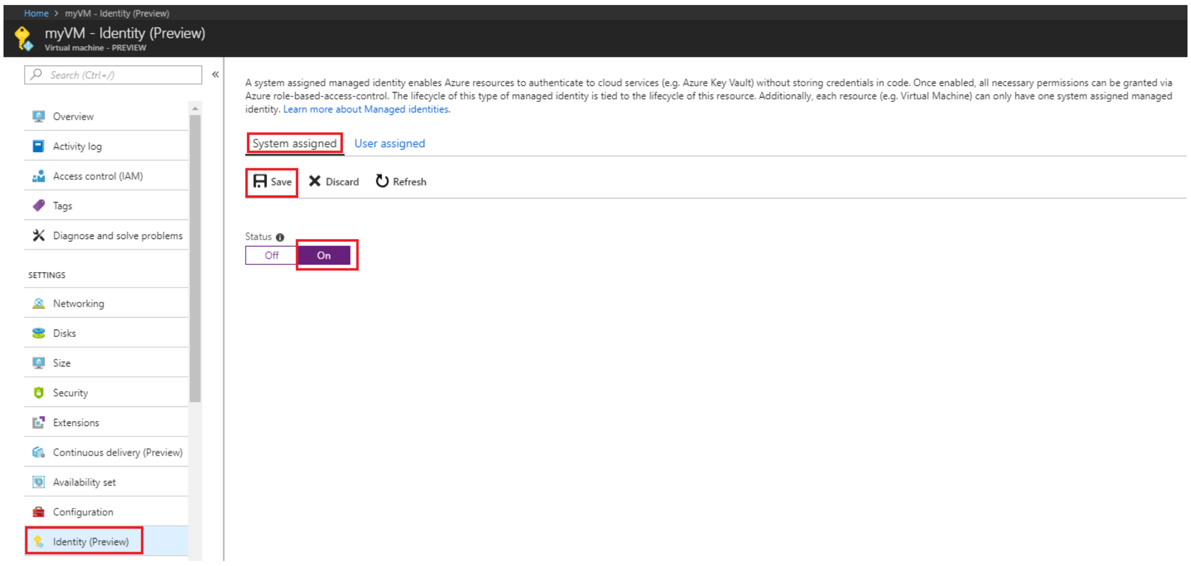
Refer to the following Quickstarts to create a VM:

- Create a Windows virtual machine with the Azure portal
- Create a Linux virtual machine with the Azure portal

Enable system-assigned managed identity on an existing VM

To enable system-assigned managed identity on a VM that was originally provisioned without it, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Sign in to the Azure portal using an account associated with the Azure subscription that contains the VM.
2. Navigate to the desired Virtual Machine and select **Identity**.
3. Under **System assigned**, **Status**, select **On** and then click **Save**:


Home > myVM - Identity (Preview)

myVM - Identity (Preview)

Virtual machine - PREVIEW

A system assigned managed identity enables Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. Once enabled, all necessary permissions can be granted via Azure role-based-access-control. The lifecycle of this type of managed identity is tied to the lifecycle of this resource. Additionally, each resource (e.g. Virtual Machine) can only have one system assigned managed identity. Learn more about Managed identities.

System assigned | User assigned

Save | Discard | Refresh

Status: Off [✓ On]

Remove system-assigned managed identity from a VM

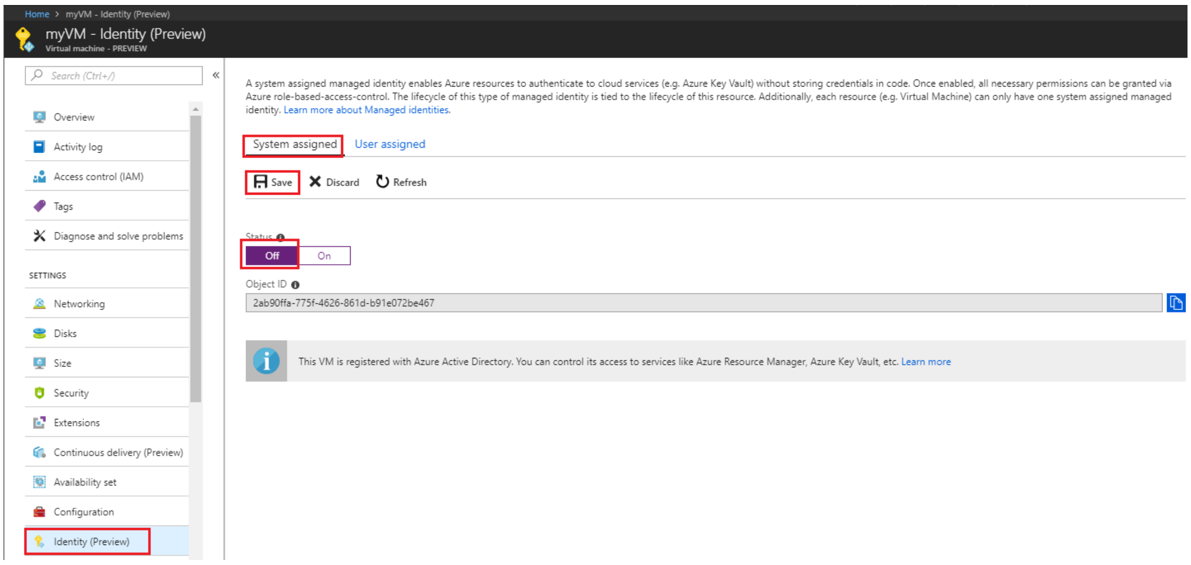
To remove system-assigned managed identity from a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

If you have a Virtual Machine that no longer needs system-assigned managed identity:

1. Sign in to the Azure portal using an account associated with the Azure subscription that contains the VM.

2. Navigate to the desired Virtual Machine and select **Identity**.

3. Under **System assigned**, **Status**, select **Off** and then click **Save**:



A system assigned managed identity enables Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. Once enabled, all necessary permissions can be granted via Azure role-based-access-control. The lifecycle of this type of managed identity is tied to the lifecycle of this resource. Additionally, each resource (e.g. Virtual Machine) can only have one system assigned managed identity. Learn more about Managed identities.

Status: Off

Object ID: 2ab90ffa-775f-4626-861d-b91e072be467

This VM is registered with Azure Active Directory. You can control its access to services like Azure Resource Manager, Azure Key Vault, etc. Learn more

User-assigned managed identity

In this section, you learn how to add and remove a user-assigned managed identity from a VM using the Azure portal.

Assign a user-assigned identity during the creation of a VM

To assign a user-assigned identity to a VM, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

Currently, the Azure portal does not support assigning a user-assigned managed identity during the creation of a VM. Instead, refer to one of the following VM creation Quickstart articles to first create a VM, and then proceed to the next section for details on assigning a user-assigned managed identity to the VM:

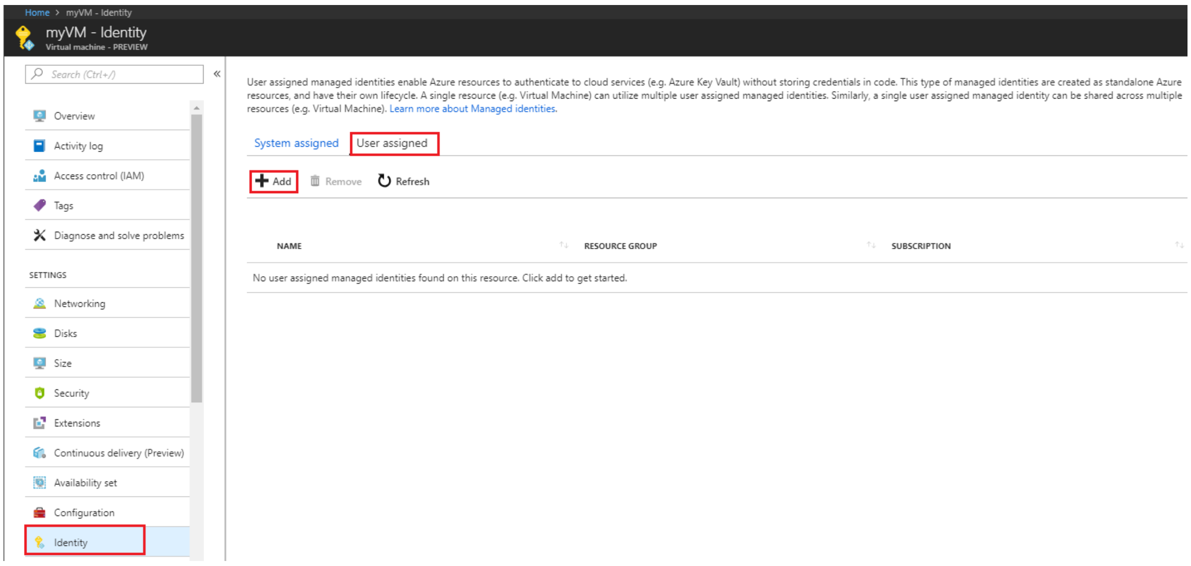
- Create a Windows virtual machine with the Azure portal
- Create a Linux virtual machine with the Azure portal

Assign a user-assigned managed identity to an existing VM

To assign a user-assigned identity to a VM, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

1. Sign in to the Azure portal using an account associated with the Azure subscription that contains the VM.

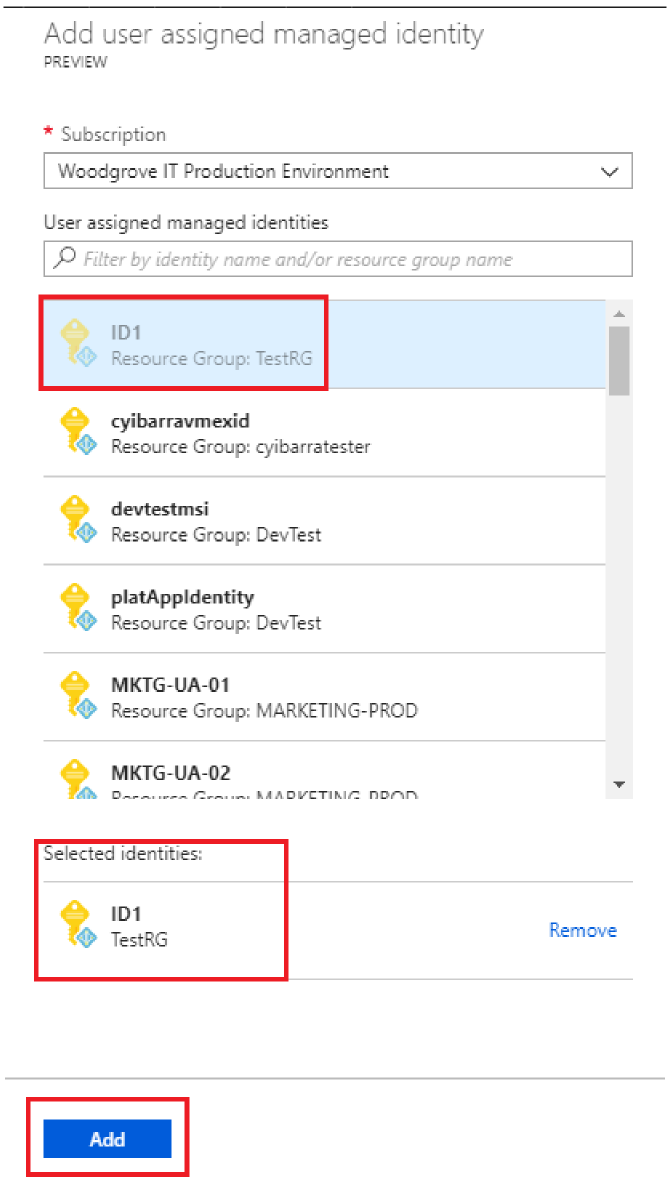
2. Navigate to the desired VM and click **Identity**, **User assigned** and then **+Add**.



User assigned managed identities enable Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. This type of managed identities are created as standalone Azure resources, and have their own lifecycle. A single resource (e.g. Virtual Machine) can utilize multiple user assigned managed identities. Similarly, a single user assigned managed identity can be shared across multiple resources (e.g. Virtual Machine). Learn more about Managed identities.

No user assigned managed identities found on this resource. Click add to get started.

3. Click the user-assigned identity you want to add to the VM and then click **Add**.

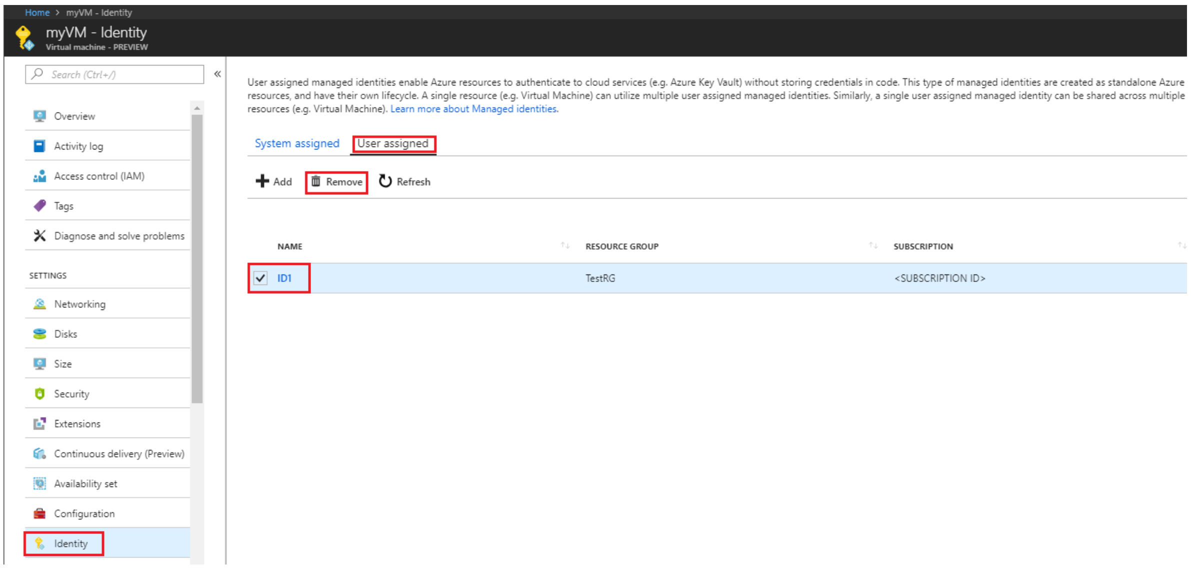

Add user assigned managed identity
PREVIEW
* Subscription
Woodgrove IT Production Environment
User assigned managed identities
Filter by identity name and/or resource group name
ID1
Resource Group: TestRG
cyibarravmexid
Resource Group: cyibarratester
devtestmsi
Resource Group: DevTest
platAppIdentity
Resource Group: DevTest
MKTG-UA-01
Resource Group: MARKETING-PROD
MKTG-UA-02
Selected identities:
ID1
TestRG
Remove
Add

Remove a user-assigned managed identity from a VM

To remove a user-assigned identity from a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Sign in to the Azure portal using an account associated with the Azure subscription that contains the VM.

2. Navigate to the desired VM and click **Identity**, **User assigned**, the name of the user-assigned managed identity you want to delete and then click **Remove** (click **Yes** in the confirmation pane).



User assigned managed identities enable Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. This type of managed identities are created as standalone Azure resources, and have their own lifecycle. A single resource (e.g. Virtual Machine) can utilize multiple user assigned managed identities. Similarly, a single user assigned managed identity can be shared across multiple resources (e.g. Virtual Machine). Learn more about Managed identities.

NAME	RESOURCE GROUP	SUBSCRIPTION
✓ ID1	TestRG	<SUBSCRIPTION ID>

Next steps

- Using the Azure portal, give an Azure VM's managed identity access to another Azure resource.

Configure managed identities for Azure resources on an Azure VM using Azure CLI

3/26/2019 • 8 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, using the Azure CLI, you learn how to perform the following managed identities for Azure resources operations on an Azure VM:

- Enable and disable the system-assigned managed identity on an Azure VM
- Add and remove a user-assigned managed identity on an Azure VM

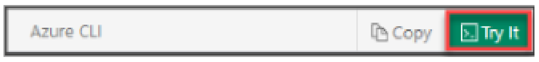
Prerequisites

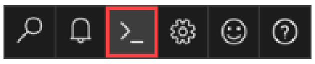
- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- To run the CLI script examples, you have three options:
  - Use Azure Cloud Shell from the Azure portal (see next section).
  - Use the embedded Azure Cloud Shell via the "Try It" button, located in the top right corner of each code block.
  - Install the latest version of the Azure CLI if you prefer to use a local CLI console.

> **NOTE**
>
> The commands have been updated to reflect the latest release of the Azure CLI.

Open Azure Cloud Shell

Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Select **Copy** to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.	
Open Cloud Shell in your browser.	

Select the **Cloud Shell** button on the menu in the upper-right corner of the Azure portal.	

System-assigned managed identity

In this section, you learn how to enable and disable the system-assigned managed identity on an Azure VM using Azure CLI.

Enable system-assigned managed identity during creation of an Azure VM

To create an Azure VM with the system-assigned managed identity enabled,your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. If you're using the Azure CLI in a local console, first sign in to Azure using az login. Use an account that is associated with the Azure subscription under which you would like to deploy the VM:

```
az login
```

2. Create a resource group for containment and deployment of your VM and its related resources, using az group create. You can skip this step if you already have resource group you would like to use instead:

```
az group create --name myResourceGroup --location westus
```

3. Create a VM using az vm create. The following example creates a VM named *myVM* with a system-assigned managed identity, as requested by the `--assign-identity` parameter. The `--admin-username` and `--admin-password` parameters specify the administrative user name and password account for virtual machine sign-in. Update these values as appropriate for your environment:

```
az vm create --resource-group myResourceGroup --name myVM --image win2016datacenter --generate-ssh-keys
--assign-identity --admin-username azureuser --admin-password myPassword12
```

Enable system-assigned managed identity on an existing Azure VM

To enable system-assigned managed identity on a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. If you're using the Azure CLI in a local console, first sign in to Azure using az login. Use an account that is associated with the Azure subscription that contains the VM.

```
az login
```

2. Use az vm identity assign with the `identity assign` command enable the system-assigned identity to an existing VM:

```
az vm identity assign -g myResourceGroup -n myVm
```

Disable system-assigned identity from an Azure VM

To disable system-assigned managed identity on a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

If you have a Virtual Machine that no longer needs the system-assigned identity, but still needs user-assigned identities, use the following command:

```
az vm update -n myVM -g myResourceGroup --set identity.type='UserAssigned'
```

If you have a virtual machine that no longer needs system-assigned identity and it has no user-assigned identities, use the following command:

NOTE

The value `none` is case sensitive. It must be lowercase.

```
az vm update -n myVM -g myResourceGroup --set identity.type="none"
```

NOTE

If you have provisioned the managed identity for Azure resources VM extension (to be deprecated), you need to remove it using az vm extension delete. For more information, see Migrate from VM extension to Azure IMDS for authentication.

User-assigned managed identity

In this section, you will learn how to add and remove a user-assigned managed identity from an Azure VM using Azure CLI.

Assign a user-assigned managed identity during the creation of an Azure VM

To assign a user-assigned identity to a VM during its creation, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

1. You can skip this step if you already have a resource group you would like to use. Create a resource group for containment and deployment of your user-assigned managed identity, using az group create. Be sure to replace the `<RESOURCE GROUP>` and `<LOCATION>` parameter values with your own values. :

   ```
   az group create --name <RESOURCE GROUP> --location <LOCATION>
   ```

2. Create a user-assigned managed identity using az identity create. The `-g` parameter specifies the resource group where the user-assigned managed identity is created, and the `-n` parameter specifies its name.

   IMPORTANT

   When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

   ```
   az identity create -g myResourceGroup -n myUserAssignedIdentity
   ```

   The response contains details for the user-assigned managed identity created, similar to the following. The resource ID value assigned to the user-assigned managed identity is used in the following step.

```
{
    "clientId": "73444643-8088-4d70-9532-c3a0fdc190fz",
    "clientSecretUrl": "https://control-westcentralus.identity.azure.net/subscriptions/<SUBSCRIPTON
ID>/resourcegroups/<RESOURCE
GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<myUserAssignedIdentity>/credentials?
tid=5678&oid=9012&aid=73444643-8088-4d70-9532-c3a0fdc190fz",
    "id": "/subscriptions/<SUBSCRIPTON ID>/resourcegroups/<RESOURCE
GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>",
    "location": "westcentralus",
    "name": "<USER ASSIGNED IDENTITY NAME>",
    "principalId": "e5fdfdc1-ed84-4d48-8551-fe9fb9dedfll",
    "resourceGroup": "<RESOURCE GROUP>",
    "tags": {},
    "tenantId": "733a8f0e-ec41-4e69-8ad8-971fc4b533bl",
    "type": "Microsoft.ManagedIdentity/userAssignedIdentities"
}
```

3. Create a VM using az vm create. The following example creates a VM associated with the new user-assigned identity, as specified by the `--assign-identity` parameter. Be sure to replace the `<RESOURCE GROUP>`, `<VM NAME>`, `<USER NAME>`, `<PASSWORD>`, and `<USER ASSIGNED IDENTITY NAME>` parameter values with your own values.

```
az vm create --resource-group <RESOURCE GROUP> --name <VM NAME> --image UbuntuLTS --admin-username
<USER NAME> --admin-password <PASSWORD> --assign-identity <USER ASSIGNED IDENTITY NAME>
```

Assign a user-assigned managed identity to an existing Azure VM

To assign a user-assigned identity to a VM, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

1. Create a user-assigned identity using az identity create. The `-g` parameter specifies the resource group where the user-assigned identity is created, and the `-n` parameter specifies its name. Be sure to replace the `<RESOURCE GROUP>` and `<USER ASSIGNED IDENTITY NAME>` parameter values with your own values:

> **IMPORTANT**
>
> Creating user-assigned managed identities with special characters (i.e. underscore) in the name is not currently supported. Please use alphanumeric characters. Check back for updates. For more information see FAQs and known issues

```
az identity create -g <RESOURCE GROUP> -n <USER ASSIGNED IDENTITY NAME>
```

The response contains details for the user-assigned managed identity created, similar to the following.

```
{
     "clientId": "73444643-8088-4d70-9532-c3a0fdc190fz",
     "clientSecretUrl": "https://control-westcentralus.identity.azure.net/subscriptions/<SUBSCRIPTON ID>/resourcegroups/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>/credentials?tid=5678&oid=9012&aid=73444643-8088-4d70-9532-c3a0fdc190fz",
     "id": "/subscriptions/<SUBSCRIPTON ID>/resourcegroups/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>",
     "location": "westcentralus",
     "name": "<USER ASSIGNED IDENTITY NAME>",
     "principalId": "e5fdfdc1-ed84-4d48-8551-fe9fb9dedfll",
     "resourceGroup": "<RESOURCE GROUP>",
     "tags": {},
     "tenantId": "733a8f0e-ec41-4e69-8ad8-971fc4b533bl",
     "type": "Microsoft.ManagedIdentity/userAssignedIdentities"
}
```

2. Assign the user-assigned identity to your VM using az vm identity assign. Be sure to replace the `<RESOURCE GROUP>` and `<VM NAME>` parameter values with your own values. The `<USER ASSIGNED IDENTITY NAME>` is the user-assigned managed identity's resource `name` property, as created in the previous step:

```
az vm identity assign -g <RESOURCE GROUP> -n <VM NAME> --identities <USER ASSIGNED IDENTITY>
```

Remove a user-assigned managed identity from an Azure VM

To remove a user-assigned identity to a VM, your account needs the Virtual Machine Contributor role assignment.

If this is the only user-assigned managed identity assigned to the virtual machine, `UserAssigned` will be removed from the identity type value. Be sure to replace the `<RESOURCE GROUP>` and `<VM NAME>` parameter values with your own values. The `<USER ASSIGNED IDENTITY>` will be the user-assigned identity's `name` property, which can be found in the identity section of the virtual machine using `az vm identity show`:

```
az vm identity remove -g <RESOURCE GROUP> -n <VM NAME> --identities <USER ASSIGNED IDENTITY>
```

If your VM does not have a system-assigned managed identity and you want to remove all user-assigned identities from it, use the following command:

NOTE

The value `none` is case sensitive. It must be lowercase.

```
az vm update -n myVM -g myResourceGroup --set identity.type="none" identity.userAssignedIdentities=null
```

If your VM has both system-assigned and user-assigned identities, you can remove all the user-assigned identities by switching to use only system-assigned. Use the following command:

```
az vm update -n myVM -g myResourceGroup --set identity.type='SystemAssigned'
identity.userAssignedIdentities=null
```

Next steps

- Managed identities for Azure resources overview
- For the full Azure VM creation Quickstarts, see:

- Create a Windows virtual machine with CLI
- Create a Linux virtual machine with CLI

Configure managed identities for Azure resources on an Azure VM using PowerShell

4/2/2019 • 8 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, using PowerShell, you learn how to perform the following managed identities for Azure resources operations on an Azure VM.

NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- Install the latest version of Azure PowerShell if you haven't already.

System-assigned managed identity

In this section, you will learn how to enable and disable the system-assigned managed identity using Azure PowerShell.

Enable system-assigned managed identity during creation of an Azure VM

To create an Azure VM with the system-assigned managed identity enabled,your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Refer to one of the following Azure VM Quickstarts, completing only the necessary sections ("Sign in to Azure", "Create resource group", "Create networking group", "Create the VM").

   When you get to the "Create the VM" section, make a slight modification to the New-AzVMConfig cmdlet syntax. Be sure to add a `-AssignIdentity:$SystemAssigned` parameter to provision the VM with the system-assigned identity enabled, for example:

   ```
   $vmConfig = New-AzVMConfig -VMName myVM -AssignIdentity:$SystemAssigned ...
   ```

- Create a Windows virtual machine using PowerShell
- Create a Linux virtual machine using PowerShell

> **NOTE**
>
> You may optionally provision the managed identities for Azure resources VM extension, but it will soon be deprecated. We recommend using the Azure Instance Metadata identity endpoint for authentication. For more information, see Migrate from the VM extension to Azure IMDS endpoint for authentication.

Enable system-assigned managed identity on an existing Azure VM

To enable system-assigned managed identity on a VM that was originally provisioned without it, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Sign in to Azure using `Connect-AzAccount`. Use an account that is associated with the Azure subscription that contains the VM.

   ```
   Connect-AzAccount
   ```

2. First retrieve the VM properties using the `Get-AzVM` cmdlet. Then to enable a system-assigned managed identity, use the `-AssignIdentity` switch on the Update-AzVM cmdlet:

   ```
   $vm = Get-AzVM -ResourceGroupName myResourceGroup -Name myVM
   Update-AzVM -ResourceGroupName myResourceGroup -VM $vm -AssignIdentity:$SystemAssigned
   ```

> **NOTE**
>
> You may optionally provision the managed identities for Azure resources VM extension, but it will soon be deprecated. We recommend using the Azure Instance Metadata identity endpoint for authentication. For more information, see Migrate from the VM extension to Azure IMDS endpoint for authentication.

Add VM system assigned identity to a group

After you have enabled system assigned identity on a VM, you can add it to a group. The following procedure adds a VM's system assigned identity to a group.

1. Sign in to Azure using `Connect-AzAccount`. Use an account that is associated with the Azure subscription that contains the VM.

   ```
   Connect-AzAccount
   ```

2. Retrieve and note the `ObjectID` (as specified in the `Id` field of the returned values) of the VM's service principal:

   ```
   Get-AzADServicePrincipal -displayname "myVM"
   ```

3. Retrieve and note the `ObjectID` (as specified in the `Id` field of the returned values) of the group:

   ```
   Get-AzADGroup -searchstring "myGroup"
   ```

4. Add the VM's service principal to the group:

```
Add-AzureADGroupMember -ObjectId "<objectID of group>" -RefObjectId "<object id of VM service
principal>"
```

Disable system-assigned managed identity from an Azure VM

To disable system-assigned managed identity on a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

If you have a Virtual Machine that no longer needs the system-assigned managed identity but still needs user-assigned managed identities, use the following cmdlet:

1. Sign in to Azure using `Connect-AzAccount`. Use an account that is associated with the Azure subscription that contains the VM.

   ```
   Connect-AzAccount
   ```

2. Retrieve the VM properties using the `Get-AzVM` cmdlet and set the `-IdentityType` parameter to `UserAssigned`:

   ```
   $vm = Get-AzVM -ResourceGroupName myResourceGroup -Name myVM
   Update-AzVm -ResourceGroupName myResourceGroup -VM $vm -IdentityType "UserAssigned"
   ```

If you have a virtual machine that no longer needs system-assigned managed identity and it has no user-assigned managed identities, use the following commands:

```
$vm = Get-AzVM -ResourceGroupName myResourceGroup -Name myVM
Update-AzVm -ResourceGroupName myResourceGroup -VM $vm -IdentityType None
```

NOTE

If you have provisioned the managed identity for Azure resources VM extension (to be deprecated), you need to remove it using the Remove-AzVMExtension. For more information, see Migrate from VM extension to Azure IMDS for authentication.

User-assigned managed identity

In this section, you learn how to add and remove a user-assigned managed identity from a VM using Azure PowerShell.

Assign a user-assigned managed identity to a VM during creation

To assign a user-assigned identity to a VM, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

1. Refer to one of the following Azure VM Quickstarts, completing only the necessary sections ("Sign in to Azure", "Create resource group", "Create networking group", "Create the VM").

   When you get to the "Create the VM" section, make a slight modification to the `New-AzVMConfig` cmdlet syntax. Add the `-IdentityType UserAssigned` and `-IdentityID` parameters to provision the VM with a user-assigned identity. Replace `<VM NAME>`, `<SUBSCRIPTION ID>`, `<RESROURCE GROUP>`, and `<USER ASSIGNED IDENTITY NAME>` with your own values. For example:

```
$vmConfig = New-AzVMConfig -VMName <VM NAME> -IdentityType UserAssigned -IdentityID
"/subscriptions/<SUBSCRIPTION ID>/resourcegroups/<RESROURCE
GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>..."
```

- Create a Windows virtual machine using PowerShell
- Create a Linux virtual machine using PowerShell

NOTE

You may optionally provision the managed identities for Azure resources VM extension, but it will soon be deprecated. We recommend using the Azure Instance Metadata identity endpoint for authentication. For more information, see Migrate from the VM extension to Azure IMDS endpoint for authentication.

Assign a user-assigned managed identity to an existing Azure VM

To assign a user-assigned identity to a VM, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

1. Sign in to Azure using `Connect-AzAccount`. Use an account that is associated with the Azure subscription that contains the VM.

```
Connect-AzAccount
```

2. Create a user-assigned managed identity using the New-AzUserAssignedIdentity cmdlet. Note the `Id` in the output because you will need this in the next step.

IMPORTANT

Creating user-assigned managed identities only supports alphanumeric, underscore and hyphen (0-9 or a-z or A-Z, _ or -) characters. Additionally, name should be limited from 3 to 128 character length for the assignment to VM/VMSS to work properly. For more information see FAQs and known issues

```
New-AzUserAssignedIdentity -ResourceGroupName <RESOURCEGROUP> -Name <USER ASSIGNED IDENTITY NAME>
```

3. Retrieve the VM properties using the `Get-AzVM` cmdlet. Then to assign a user-assigned managed identity to the Azure VM, use the `-IdentityType` and `-IdentityID` switch on the Update-AzVM cmdlet. The value for the `-IdentityId` parameter is the `Id` you noted in the previous step. Replace `<VM NAME>`, `<SUBSCRIPTION ID>`, `<RESROURCE GROUP>`, and `<USER ASSIGNED IDENTITY NAME>` with your own values.

WARNING

To retain any previously user-assigned managed identities assigned to the VM, query the `Identity` property of the VM object (for example, `$vm.Identity`). If any user assigned managed identities are returned, include them in the following command along with the new user assigned managed identity you would like to assign to the VM.

```
$vm = Get-AzVM -ResourceGroupName <RESOURCE GROUP> -Name <VM NAME>
Update-AzVM -ResourceGroupName <RESOURCE GROUP> -VM $vm -IdentityType UserAssigned -IdentityID
"/subscriptions/<SUBSCRIPTION ID>/resourcegroups/<RESROURCE
GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>"
```

NOTE

You may optionally provision the managed identities for Azure resources VM extension, but it will soon be deprecated. We recommend using the Azure Instance Metadata identity endpoint for authentication. For more information, see Migrate from the VM extension to Azure IMDS endpoint for authentication.

Remove a user-assigned managed identity from an Azure VM

To remove a user-assigned identity to a VM, your account needs the Virtual Machine Contributor role assignment.

If your VM has multiple user-assigned managed identities, you can remove all but the last one using the following commands. Be sure to replace the `<RESOURCE GROUP>` and `<VM NAME>` parameter values with your own values. The `<USER ASSIGNED IDENTITY NAME>` is the user-assigned managed identity's name property, which should remain on the VM. This information can be found by querying the `Identity` property of the VM object. For example, `$vm.Identity`:

```
$vm = Get-AzVm -ResourceGroupName myResourceGroup -Name myVm
Update-AzVm -ResourceGroupName myResourceGroup -VirtualMachine $vm -IdentityType UserAssigned -IdentityID
<USER ASSIGNED IDENTITY NAME>
```

If your VM does not have a system-assigned managed identity and you want to remove all user-assigned managed identities from it, use the following command:

```
$vm = Get-AzVm -ResourceGroupName myResourceGroup -Name myVm
Update-AzVm -ResourceGroupName myResourceGroup -VM $vm -IdentityType None
```

If your VM has both system-assigned and user-assigned managed identities, you can remove all the user-assigned managed identities by switching to use only system-assigned managed identities.

```
$vm = Get-AzVm -ResourceGroupName myResourceGroup -Name myVm
Update-AzVm -ResourceGroupName myResourceGroup -VirtualMachine $vm -IdentityType "SystemAssigned"
```

Next steps

- Managed identities for Azure resources overview
- For the full Azure VM creation Quickstarts, see:
  - Create a Windows virtual machine with PowerShell
  - Create a Linux virtual machine with PowerShell

Configure managed identities for Azure resources on an Azure VM using a templates

3/26/2019 • 7 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, using the Azure Resource Manager deployment template, you learn how to perform the following managed identities for Azure resources operations on an Azure VM:

Prerequisites

- If you're unfamiliar with using Azure Resource Manager deployment template, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.

Azure Resource Manager templates

As with the Azure portal and scripting, Azure Resource Manager templates provide the ability to deploy new or modified resources defined by an Azure resource group. Several options are available for template editing and deployment, both local and portal-based, including:

- Using a custom template from the Azure Marketplace, which allows you to create a template from scratch, or base it on an existing common or quickstart template.
- Deriving from an existing resource group, by exporting a template from either the original deployment, or from the current state of the deployment.
- Using a local JSON editor (such as VS Code), and then uploading and deploying by using PowerShell or CLI.
- Using the Visual Studio Azure Resource Group project to both create and deploy a template.

Regardless of the option you choose, template syntax is the same during initial deployment and redeployment. Enabling a system or user-assigned managed identity on a new or existing VM is done in the same manner. Also, by default, Azure Resource Manager does an incremental update to deployments.

System-assigned managed identity

In this section, you will enable and disable a system-assigned managed identity using an Azure Resource Manager template.

Enable system-assigned managed identity during creation of an Azure VM or on an existing VM

To enable system-assigned managed identity on a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Whether you sign in to Azure locally or via the Azure portal, use an account that is associated with the Azure subscription that contains the VM.

2. To enable system-assigned managed identity, load the template into an editor, locate the `Microsoft.Compute/virtualMachines` resource of interest within the `resources` section and add the `"identity"` property at the same level as the `"type": "Microsoft.Compute/virtualMachines"` property. Use the following syntax:

```
"identity": {
    "type": "SystemAssigned"
},
```

> **NOTE**
>
> You may optionally provision the managed identities for Azure resources VM extension by specifying it as a `resources` element in the template. This step is optional as you can use the Azure Instance Metadata Service (IMDS) identity endpoint, to retrieve tokens as well. For more information, see Migrate from VM extension to Azure IMDS for authentication.

3. When you're done, the following sections should added to the `resource` section of your template and it should resemble the following:

```
"resources": [
     {
         //other resource provider properties...
         "apiVersion": "2018-06-01",
         "type": "Microsoft.Compute/virtualMachines",
         "name": "[variables('vmName')]",
         "location": "[resourceGroup().location]",
         "identity": {
             "type": "SystemAssigned",
             },
         },

         //The following appears only if you provisioned the optional VM extension (to be deprecated)
         {
         "type": "Microsoft.Compute/virtualMachines/extensions",
         "name": "[concat(variables('vmName'),'/ManagedIdentityExtensionForWindows')]",
         "apiVersion": "2018-06-01",
         "location": "[resourceGroup().location]",
         "dependsOn": [
             "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
         ],
         "properties": {
             "publisher": "Microsoft.ManagedIdentity",
             "type": "ManagedIdentityExtensionForWindows",
             "typeHandlerVersion": "1.0",
             "autoUpgradeMinorVersion": true,
             "settings": {
                 "port": 50342
             }
         }
     }
 ]
```

Assign a role the VM's system-assigned managed identity

After you have enabled system-assigned managed identity on your VM, you may want to grant it a role such as **Reader** access to the resource group in which it was created.

To assign a role to your VM's system-assigned identity, your account needs the User Access Administrator role

assignment.

1. Whether you sign in to Azure locally or via the Azure portal, use an account that is associated with the Azure subscription that contains the VM.

2. Load the template into an editor and add the following information to give your VM **Reader** access to the resource group in which it was created. Your template structure may vary depending on the editor and the deployment model you choose.

   Under the `parameters` section add the following:

   ```
   "builtInRoleType": {
         "type": "string",
         "defaultValue": "Reader"
       },
       "rbacGuid": {
         "type": "string"
       }
   ```

   Under the `variables` section add the following:

   ```
   "Reader": "[concat('/subscriptions/', subscription().subscriptionId,
   '/providers/Microsoft.Authorization/roleDefinitions/', 'acdd72a7-3385-48ef-bd42-f606fba81ae7')]"
   ```

   Under the `resources` section add the following:

   ```
   {
       "apiVersion": "2017-09-01",
        "type": "Microsoft.Authorization/roleAssignments",
        "name": "[parameters('rbacGuid')]",
        "properties": {
               "roleDefinitionId": "[variables(parameters('builtInRoleType'))]",
               "principalId": "[reference(variables('vmResourceId'), '2017-12-01',
   'Full').identity.principalId]",
               "scope": "[resourceGroup().id]"
         },
         "dependsOn": [
               "[concat('Microsoft.Compute/virtualMachines/', parameters('vmName'))]"
           ]
   }
   ```

Disable a system-assigned managed identity from an Azure VM

To remove system-assigned managed identity from a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Whether you sign in to Azure locally or via the Azure portal, use an account that is associated with the Azure subscription that contains the VM.

2. Load the template into an editor and locate the `Microsoft.Compute/virtualMachines` resource of interest within the `resources` section. If you have a VM that only has system-assigned managed identity, you can disable it by changing the identity type to `None`.

   Microsoft.Compute/virtualMachines API version 2018-06-01

   If your VM has both system and user-assigned managed identities, remove `SystemAssigned` from the identity type and keep `UserAssigned` along with the `userAssignedIdentities` dictionary values.

   Microsoft.Compute/virtualMachines API version 2018-06-01

If your `apiVersion` is `2017-12-01` and your VM has both system and user-assigned managed identities, remove `SystemAssigned` from the identity type and keep `UserAssigned` along with the `identityIds` array of the user-assigned managed identities.

The following example shows you how remove a system-assigned managed identity from a VM with no user-assigned managed identities:

```
{
    "apiVersion": "2018-06-01",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[parameters('vmName')]",
    "location": "[resourceGroup().location]",
    "identity": {
        "type": "None"
}
```

User-assigned managed identity

In this section, you assign a user-assigned managed identity to an Azure VM using Azure Resource Manager template.

> **NOTE**
>
> To create a user-assigned managed identity using an Azure Resource Manager Template, see Create a user-assigned managed identity.

Assign a user-assigned managed identity to an Azure VM

To assign a user-assigned identity to a VM, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

1. Under the `resources` element, add the following entry to assign a user-assigned managed identity to your VM. Be sure to replace `<USERASSIGNEDIDENTITY>` with the name of the user-assigned managed identity you created.

   Microsoft.Compute/virtualMachines API version 2018-06-01

   If your `apiVersion` is `2018-06-01`, your user-assigned managed identities are stored in the `userAssignedIdentities` dictionary format and the `<USERASSIGNEDIDENTITYNAME>` value must be stored in a variable defined in the `variables` section of your template.

   ```
   {
       "apiVersion": "2018-06-01",
       "type": "Microsoft.Compute/virtualMachines",
       "name": "[variables('vmName')]",
       "location": "[resourceGroup().location]",
       "identity": {
           "type": "userAssigned",
           "userAssignedIdentities": {
               "
   [resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',variables('<USERASSIGNEDIDENTITYNAME>')
   )]": {}
           }
        }
   }
   ```

   Microsoft.Compute/virtualMachines API version 2017-12-01

If your `apiVersion` is `2017-12-01`, your user-assigned managed identities are stored in the `identityIds` array and the `<USERASSIGNEDIDENTITYNAME>` value must be stored in a variable defined in the `variables` section of your template.

```
{
    "apiVersion": "2017-12-01",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[variables('vmName')]",
    "location": "[resourceGroup().location]",
    "identity": {
        "type": "userAssigned",
        "identityIds": [
            "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',variables('<USERASSIGNEDIDENTITYNAME>')
)]"
        ]
    }
}
```

2. When you're done, the following sections should added to the `resource` section of your template and it should resemble the following:

   Microsoft.Compute/virtualMachines API version 2018-06-01

```
"resources": [
     {
         //other resource provider properties...
         "apiVersion": "2018-06-01",
         "type": "Microsoft.Compute/virtualMachines",
         "name": "[variables('vmName')]",
         "location": "[resourceGroup().location]",
         "identity": {
             "type": "userAssigned",
             "userAssignedIdentities": {
                "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',variables('<USERASSIGNEDIDENTITYNAME>')
)]": {}
             }
         }
     },
     //The following appears only if you provisioned the optional VM extension (to be deprecated)
     {
         "type": "Microsoft.Compute/virtualMachines/extensions",
         "name": "[concat(variables('vmName'),'/ManagedIdentityExtensionForWindows')]",
         "apiVersion": "2018-06-01-preview",
         "location": "[resourceGroup().location]",
         "dependsOn": [
             "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
         ],
         "properties": {
             "publisher": "Microsoft.ManagedIdentity",
             "type": "ManagedIdentityExtensionForWindows",
             "typeHandlerVersion": "1.0",
             "autoUpgradeMinorVersion": true,
             "settings": {
                 "port": 50342
         }
     }
    }
 ]
```

   Microsoft.Compute/virtualMachines API version 2017-12-01

```
"resources": [
     {
         //other resource provider properties...
         "apiVersion": "2017-12-01",
         "type": "Microsoft.Compute/virtualMachines",
         "name": "[variables('vmName')]",
         "location": "[resourceGroup().location]",
         "identity": {
             "type": "userAssigned",
             "identityIds": [
                "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',variables('<USERASSIGNEDIDENTITYNAME>')
)]"
             ]
         }
     },

     //The following appears only if you provisioned the optional VM extension (to be deprecated)
     {
         "type": "Microsoft.Compute/virtualMachines/extensions",
         "name": "[concat(variables('vmName'),'/ManagedIdentityExtensionForWindows')]",
         "apiVersion": "2015-05-01-preview",
         "location": "[resourceGroup().location]",
         "dependsOn": [
             "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
         ],
         "properties": {
             "publisher": "Microsoft.ManagedIdentity",
             "type": "ManagedIdentityExtensionForWindows",
             "typeHandlerVersion": "1.0",
             "autoUpgradeMinorVersion": true,
             "settings": {
                 "port": 50342
         }
     }
    }
 ]
```

Remove a user-assigned managed identity from an Azure VM

To remove a user-assigned identity from a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Whether you sign in to Azure locally or via the Azure portal, use an account that is associated with the Azure subscription that contains the VM.

2. Load the template into an editor and locate the `Microsoft.Compute/virtualMachines` resource of interest within the `resources` section. If you have a VM that only has user-assigned managed identity, you can disable it by changing the identity type to `None`.

   The following example shows you how remove all user-assigned managed identities from a VM with no system-assigned managed identities:

   ```
    {
      "apiVersion": "2018-06-01",
      "type": "Microsoft.Compute/virtualMachines",
      "name": "[parameters('vmName')]",
      "location": "[resourceGroup().location]",
      "identity": {
          "type": "None"
    }
   ```

Microsoft.Compute/virtualMachines API version 2018-06-01

To remove a single user-assigned managed identity from a VM, remove it from the `useraAssignedIdentities` dictionary.

If you have a system-assigned managed identity, keep it in the in the `type` value under the `identity` value.

Microsoft.Compute/virtualMachines API version 2017-12-01

To remove a single user-assigned managed identity from a VM, remove it from the `identityIds` array.

If you have a system-assigned managed identity, keep it in the in the `type` value under the `identity` value.

Next steps

- Managed identities for Azure resources overview.

Configure Managed identities for Azure resources on an Azure VM using REST API calls

3/26/2019 • 15 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed system identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, using CURL to make calls to the Azure Resource Manager REST endpoint, you learn how to perform the following managed identities for Azure resources operations on an Azure VM:

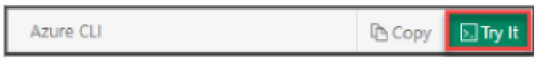
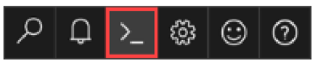
- Enable and disable the system-assigned managed identity on an Azure VM
- Add and remove a user-assigned managed identity on an Azure VM

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- If you are using Windows, install the Windows Subsystem for Linux or use the Azure Cloud Shell in the Azure portal.
- Install the Azure CLI local console, if you use the Windows Subsystem for Linux or a Linux distribution OS.
- If you are using Azure CLI local console, sign in to Azure using `az login` with an account that is associated with the Azure subscription you would like to manage system or user-assigned managed identities.

Open Azure Cloud Shell

Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Select **Copy** to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.	
Open Cloud Shell in your browser.	
Select the **Cloud Shell** button on the menu in the upper-right corner of the Azure portal.	

System-assigned managed identity

In this section, you learn how to enable and disable system-assigned managed identity on an Azure VM using CURL to make calls to the Azure Resource Manager REST endpoint.

Enable system-assigned managed identity during creation of an Azure VM

To create an Azure VM with the system-assigned managed identity enabled,your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Create a resource group for containment and deployment of your VM and its related resources, using az group create. You can skip this step if you already have resource group you would like to use instead:

```
az group create --name myResourceGroup --location westus
```

2. Create a network interface for your VM:

```
 az network nic create -g myResourceGroup --vnet-name myVnet --subnet mySubnet -n myNic
```

3. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your VM with a system-assigned managed identity.

```
az account get-access-token
```

4. Create a VM using CURL to call the Azure Resource Manager REST endpoint. The following example creates a VM named *myVM* with a system-assigned managed identity, as identified in the request body by the value `"identity":{"type":"SystemAssigned"}`. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01' -X PUT -d '{"location":"westus","name":"myVM","identity":{"type":"SystemAssigned"},"properties":
{"hardwareProfile":{"vmSize":"Standard_D2_v2"},"storageProfile":{"imageReference":{"sku":"2016-
Datacenter","publisher":"MicrosoftWindowsServer","version":"latest","offer":"WindowsServer"},"osDisk":
{"caching":"ReadWrite","managedDisk":
{"storageAccountType":"Standard_LRS"},"name":"myVM3osdisk","createOption":"FromImage"},"dataDisks":
[{"diskSizeGB":1023,"createOption":"Empty","lun":0},
{"diskSizeGB":1023,"createOption":"Empty","lun":1}]},"osProfile":
{"adminUsername":"azureuser","computerName":"myVM","adminPassword":"<SECURE PASSWORD
STRING>"},"networkProfile":{"networkInterfaces":[{"id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/networkInterfaces/myNic","properties":
{"primary":true}}]}}}' -H "Content-Type: application/json" -H "Authorization: Bearer <ACCESS TOKEN>"
```

```
PUT https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.

REQUEST HEADER	DESCRIPTION
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "location":"westus",
   "name":"myVM",
   "identity":{
      "type":"SystemAssigned"
   },
   "properties":{
      "hardwareProfile":{
         "vmSize":"Standard_D2_v2"
      },
      "storageProfile":{
         "imageReference":{
            "sku":"2016-Datacenter",
            "publisher":"MicrosoftWindowsServer",
            "version":"latest",
            "offer":"WindowsServer"
         },
         "osDisk":{
            "caching":"ReadWrite",
            "managedDisk":{
               "storageAccountType":"Standard_LRS"
            },
            "name":"myVM3osdisk",
            "createOption":"FromImage"
         },
         "dataDisks":[
            {
               "diskSizeGB":1023,
               "createOption":"Empty",
               "lun":0
            },
            {
               "diskSizeGB":1023,
               "createOption":"Empty",
               "lun":1
            }
         ]
      },
      "osProfile":{
         "adminUsername":"azureuser",
         "computerName":"myVM",
         "adminPassword":"myPassword12"
      },
      "networkProfile":{
         "networkInterfaces":[
            {
               "id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/networkInterfaces/myNic",
               "properties":{
                  "primary":true
               }
            }
         ]
      }
   }
}
```

Enable system-assigned identity on an existing Azure VM

To enable system-assigned managed identity on a VM that was originally provisioned without it, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your VM with a system-assigned managed identity.

   ```
   az account get-access-token
   ```

2. Use the following CURL command to call the Azure Resource Manager REST endpoint to enable system-assigned managed identity on your VM as identified in the request body by the value `{"identity":{"type":"SystemAssigned"}` for a VM named *myVM*. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

   IMPORTANT

   To ensure you don't delete any existing user-assigned managed identities that are assigned to the VM, you need to list the user-assigned managed identities by using this CURL command:
   ```
   curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
   GROUP>/providers/Microsoft.Compute/virtualMachines/<VM NAME>?api-version=2018-06-01' -H
   "Authorization: Bearer <ACCESS TOKEN>"
   ```
   . If you have any user-assigned managed identities assigned to the VM as identified in the `identity` value in the response, skip to step 3 that shows you how to retain user-assigned managed identities while enabling system-assigned managed identity on your VM.

   ```
   curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
   ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
   01' -X PATCH -d '{"identity":{"type":"SystemAssigned"}}' -H "Content-Type: application/json" -H
   Authorization:"Bearer <ACCESS TOKEN>"
   ```

   ```
   PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
   ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
   01 HTTP/1.1
   ```

   Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

   Request body

   ```
    {
       "identity":{
          "type":"SystemAssigned"
       }
    }
   ```

3. To enable system-assigned managed identity on a VM with existing user-assigned managed identities, you need to add `SystemAssigned` to the `type` value.

For example, if your VM has the user-assigned managed identities `ID1` and `ID2` assigned to it, and you would like to add system-assigned managed identity to the VM, use the following CURL call. Replace `<ACCESS TOKEN>` and `<SUBSCRIPTION ID>` with values appropriate to your environment.

API version `2018-06-01` stores user-assigned managed identities in the `userAssignedIdentities` value in a dictionary format as opposed to the `identityIds` value in an array format used in API version `2017-12-01`.

API VERSION 2018-06-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01' -X PATCH -d '{"identity":{"type":"SystemAssigned, UserAssigned", "userAssignedIdentities":
{"/subscriptions/<<SUBSCRIPTION
ID>>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":
{},"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":
{}}}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned, UserAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<<SUBSCRIPTION
ID>>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":{

          },
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":{

          }
       }
    }
 }
```

API VERSION 2017-12-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01' -X PATCH -d '{"identity":{"type":"SystemAssigned, UserAssigned", "identityIds":
["/subscriptions/<<SUBSCRIPTION
ID>>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1","/su
bscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2"]}}' -
H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned, UserAssigned",
       "identityIds":[
          "/subscriptions/<<SUBSCRIPTION
ID>>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1",
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2"
       ]
    }
 }
```

Disable system-assigned managed identity from an Azure VM

To disable system-assigned managed identity on a VM, your account needs the Virtual Machine Contributor role assignment. No additional Azure AD directory role assignments are required.

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your VM with a system-assigned managed identity.

```
az account get-access-token
```

2. Update the VM using CURL to call the Azure Resource Manager REST endpoint to disable system-assigned managed identity. The following example disables system-assigned managed identity as identified in the request body by the value `{"identity":{"type":"None"}}` from a VM named *myVM*. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

IMPORTANT

To ensure you don't delete any existing user-assigned managed identities that are assigned to the VM, you need to list the user-assigned managed identities by using this CURL command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Compute/virtualMachines/<VM NAME>?api-version=2018-06-01' -H
"Authorization: Bearer <ACCESS TOKEN>"
```

. If you have any user-assigned managed identities assigned to the VM as identified in the `identity` value in the response, skip to step 3 that shows you how to retain user-assigned managed identities while disabling system-assigned managed identity on your VM.

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01' -X PATCH -d '{"identity":{"type":"None"}}' -H "Content-Type: application/json" -H
Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "identity":{
      "type":"None"
   }
}
```

To remove system-assigned managed identity from a virtual machine that has user-assigned managed identities, remove `SystemAssigned` from the `{"identity":{"type:" "}}` value while keeping the `UserAssigned` value and the `userAssignedIdentities` dictionary values if you are using **API version 2018-06-01**. If you are using **API version 2017-12-01** or earlier, keep the `identityIds` array.

User-assigned managed identity

In this section, you learn how to add and remove user-assigned managed identity on an Azure VM using CURL to make calls to the Azure Resource Manager REST endpoint.

Assign a user-assigned managed identity during the creation of an Azure VM

To assign a user-assigned identity to a VM, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your VM with a system-assigned managed identity.

```
az account get-access-token
```

2. Create a network interface for your VM:

```
 az network nic create -g myResourceGroup --vnet-name myVnet --subnet mySubnet -n myNic
```

3. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your VM with a system-assigned managed identity.

```
az account get-access-token
```

4. Create a user-assigned managed identity using the instructions found here: Create a user-assigned managed identity.

5. Create a VM using CURL to call the Azure Resource Manager REST endpoint. The following example creates a VM named *myVM* in the resource group *myResourceGroup* with a user-assigned managed identity `ID1`, as identified in the request body by the value `"identity":{"type":"UserAssigned"}`. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

API VERSION 2018-06-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01' -X PUT -d '{"location":"westus","name":"myVM","identity":{"type":"UserAssigned","identityIds":
["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"]},"pr
operties":{"hardwareProfile":{"vmSize":"Standard_D2_v2"},"storageProfile":{"imageReference":
{"sku":"2016-
Datacenter","publisher":"MicrosoftWindowsServer","version":"latest","offer":"WindowsServer"},"osDisk":
{"caching":"ReadWrite","managedDisk":
{"storageAccountType":"Standard_LRS"},"name":"myVM3osdisk","createOption":"FromImage"},"dataDisks":
[{"diskSizeGB":1023,"createOption":"Empty","lun":0},
{"diskSizeGB":1023,"createOption":"Empty","lun":1}]},"osProfile":
{"adminUsername":"azureuser","computerName":"myVM","adminPassword":"myPassword12"},"networkProfile":
{"networkInterfaces":[{"id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/networkInterfaces/myNic","properties":
{"primary":true}}]}}}' -H "Content-Type: application/json" -H "Authorization: Bearer <ACCESS TOKEN>"
```

```
PUT https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "location":"westus",
    "name":"myVM",
    "identity":{
       "type":"UserAssigned",
       "identityIds":[
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"
       ]
    },
    "properties":{
       "hardwareProfile":{
          "vmSize":"Standard_D2_v2"
       },
       "storageProfile":{
          "imageReference":{
             "sku":"2016-Datacenter",
             "publisher":"MicrosoftWindowsServer",
             "version":"latest",
             "offer":"WindowsServer"
          },
          "osDisk":{
             "caching":"ReadWrite",
             "managedDisk":{
                "storageAccountType":"Standard_LRS"
             },
             "name":"myVM3osdisk",
             "createOption":"FromImage"
          },
          "dataDisks":[
             {
                "diskSizeGB":1023,
                "createOption":"Empty",
                "lun":0
             },
             {
                "diskSizeGB":1023,
                "createOption":"Empty",
                "lun":1
             }
          ]
       },
       "osProfile":{
          "adminUsername":"azureuser",
          "computerName":"myVM",
          "adminPassword":"myPassword12"
       },
       "networkProfile":{
          "networkInterfaces":[
             {
                "id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/networkInterfaces/myNic",
                "properties":{
                   "primary":true
                }
             }
          ]
       }
    }
 }
```

API VERSION 2017-12-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01' -X PUT -d '{"location":"westus","name":"myVM","identity":{"type":"UserAssigned","identityIds":
["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"]},"pr
operties":{"hardwareProfile":{"vmSize":"Standard_D2_v2"},"storageProfile":{"imageReference":
{"sku":"2016-
Datacenter","publisher":"MicrosoftWindowsServer","version":"latest","offer":"WindowsServer"},"osDisk":
{"caching":"ReadWrite","managedDisk":
{"storageAccountType":"Standard_LRS"},"name":"myVM3osdisk","createOption":"FromImage"},"dataDisks":
[{"diskSizeGB":1023,"createOption":"Empty","lun":0},
{"diskSizeGB":1023,"createOption":"Empty","lun":1}]},"osProfile":
{"adminUsername":"azureuser","computerName":"myVM","adminPassword":"myPassword12"},"networkProfile":
{"networkInterfaces":[{"id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/networkInterfaces/myNic","properties":
{"primary":true}}]}}}' -H "Content-Type: application/json" -H "Authorization: Bearer <ACCESS TOKEN>"
```

```
PUT https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "location":"westus",
   "name":"myVM",
   "identity":{
      "type":"UserAssigned",
      "identityIds":[
         "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"
      ]
   },
   "properties":{
      "hardwareProfile":{
         "vmSize":"Standard_D2_v2"
      },
      "storageProfile":{
         "imageReference":{
            "sku":"2016-Datacenter",
            "publisher":"MicrosoftWindowsServer",
            "version":"latest",
            "offer":"WindowsServer"
         },
         "osDisk":{
            "caching":"ReadWrite",
            "managedDisk":{
               "storageAccountType":"Standard_LRS"
            },
            "name":"myVM3osdisk",
            "createOption":"FromImage"
         },
         "dataDisks":[
            {
               "diskSizeGB":1023,
               "createOption":"Empty",
               "lun":0
            },
            {
               "diskSizeGB":1023,
               "createOption":"Empty",
               "lun":1
            }
         ]
      },
      "osProfile":{
         "adminUsername":"azureuser",
         "computerName":"myVM",
         "adminPassword":"myPassword12"
      },
      "networkProfile":{
         "networkInterfaces":[
            {
               "id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/networkInterfaces/myNic",
               "properties":{
                  "primary":true
               }
            }
         ]
      }
   }
}
```

Assign a user-assigned managed identity to an existing Azure VM

To assign a user-assigned identity to a VM, your account needs the Virtual Machine Contributor and Managed Identity Operator role assignments. No additional Azure AD directory role assignments are required.

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your VM with a system-assigned managed identity.

   ```
   az account get-access-token
   ```

2. Create a user-assigned managed identity using the instructions found here, Create a user-assigned managed identity.

3. To ensure you don't delete existing user or system-assigned managed identities that are assigned to the VM, you need to list the identity types assigned to the VM by using the following CURL command. If you have managed identities assigned to the virtual machine scale set, they are listed under in the `identity` value.

   ```
   curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
   GROUP>/providers/Microsoft.Compute/virtualMachines/<VM NAME>?api-version=2018-06-01' -H "Authorization:
   Bearer <ACCESS TOKEN>"
   ```

   ```
   GET https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
   GROUP>/providers/Microsoft.Compute/virtualMachines/<VM NAME>?api-version=2018-06-01 HTTP/1.1
   ```

   Request headers

REQUEST HEADER	DESCRIPTION
Authorization	Required. Set to a valid `Bearer` access token.

   If you have any user or system-assigned managed identities assigned to the VM as identified in the `identity` value in the response, skip to step 5 that shows you how to retain the system-assigned managed identity while adding a user-assigned managed identity on your VM.

4. If you don't have any user-assigned managed identities assigned to your VM, use the following CURL command to call the Azure Resource Manager REST endpoint to assign the first user-assigned managed identity to the VM.

   The following examples assigns a user-assigned managed identity, `ID1` to a VM named *myVM* in the resource group *myResourceGroup*. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

   API VERSION 2018-06-01

   ```
   curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
   ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
   01' -X PATCH -d '{"identity":{"type":"UserAssigned", "userAssignedIdentities":
   {"/subscriptions/<SUBSCRIPTION
   ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":
   {}}}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
   ```

   ```
   PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
   ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
   01 HTTP/1.1
   ```

   Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"UserAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":{

          }
       }
    }
 }
```

API VERSION 2017-12-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01' -X PATCH -d '{"identity":{"type":"userAssigned", "identityIds":["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"]}}' -
H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"userAssigned",
       "identityIds":[
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"
       ]
    }
 }
```

5. If you have an existing user-assigned or system-assigned managed identity assigned to your VM:

API VERSION 2018-06-01

Add the user-assigned managed identity to the `userAssignedIdentities` dictionary value.

For example, if you have system-assigned managed identity and the user-assigned managed identity `ID1` currently assigned to your VM and would like to add the user-assigned managed identity `ID2` to it:

```
curl  'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01' -X PATCH -d '{"identity":{"type":"SystemAssigned, UserAssigned", "userAssignedIdentities":
{"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":
{},"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":
{}}}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned, UserAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":{

          },
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":{

          }
       }
    }
 }
```

API VERSION 2017-12-01

Retain the user-assigned managed identities you would like to keep in the `identityIds` array value while adding the new user-assigned managed identity.

For example, if you have system-assigned managed identity and the user-assigned managed identity `ID1` currently assigned to your VM and would like to add the user-assigned managed identity `ID2` to it:

```
curl  'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01' -X PATCH -d '{"identity":{"type":"SystemAssigned,UserAssigned", "identityIds":
["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1","/sub
scriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2"]}}' -
H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned,UserAssigned",
       "identityIds":[
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1",
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2"
       ]
    }
 }
```

Remove a user-assigned managed identity from an Azure VM

To remove a user-assigned identity to a VM, your account needs the Virtual Machine Contributor role assignment.

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your VM with a system-assigned managed identity.

```
az account get-access-token
```

2. To ensure you don't delete any existing user-assigned managed identities that you would like to keep assigned to the VM or remove the system-assigned managed identity, you need to list the managed identities by using the following CURL command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Compute/virtualMachines/<VM NAME>?api-version=2018-06-01' -H "Authorization:
Bearer <ACCESS TOKEN>"
```

```
GET https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Compute/virtualMachines/<VM NAME>?api-version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

If you have managed identities assigned to the VM, they are listed in the response in the `identity` value.

For example, if you have user-assigned managed identities `ID1` and `ID2` assigned to your VM, and you only want to keep `ID1` assigned and retain the system-assigned identity:

API VERSION 2018-06-01

Add `null` to the user-assigned managed identity you would like to remove:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01' -X PATCH -d '{"identity":{"type":"SystemAssigned, UserAssigned", "userAssignedIdentities":
{"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":null}
}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned, UserAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":null
       }
    }
 }
```

API VERSION 2017-12-01

Retain only the user-assigned managed identity(s) you would like to keep in the `identityIds` array:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01' -X PATCH -d '{"identity":{"type":"SystemAssigned, UserAssigned", "identityIds":
["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"]}}' -
H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2017-12-
01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned, UserAssigned",
       "identityIds":[
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"
       ]
    }
 }
```

If your VM has both system-assigned and user-assigned managed identities, you can remove all the user-assigned managed identities by switching to use only system-assigned managed identity using the following command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-01' -X
PATCH -d '{"identity":{"type":"SystemAssigned"}}' -H "Content-Type: application/json" -H Authorization:"Bearer
<ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-01
HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "identity":{
      "type":"SystemAssigned"
   }
}
```

If your VM has only user-assigned managed identities and you would like to remove them all, use the following command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-01' -X
PATCH -d '{"identity":{"type":"None"}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS
TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachines/myVM?api-version=2018-06-01
HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "identity":{
      "type":"None"
   }
}
```

Next steps

For information on how to create, list, or delete user-assigned managed identities using REST see:

- Create, list or delete a user-assigned managed identities using REST API calls

Configure a VM with managed identities for Azure resources using an Azure SDK

3/26/2019 • 2 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory (AD). You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, you learn how to enable and remove managed identities for Azure resources for an Azure VM, using an Azure SDK.

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.

Azure SDKs with managed identities for Azure resources support

Azure supports multiple programming platforms through a series of Azure SDKs. Several of them have been updated to support managed identities for Azure resources, and provide corresponding samples to demonstrate usage. This list is updated as additional support is added:

SDK	SAMPLE
.NET	Manage resource from a VM enabled with managed identities for Azure resources enabled
Java	Manage storage from a VM enabled with managed identities for Azure resources
Node.js	Create a VM with system-assigned managed identity enabled
Python	Create a VM with system-assigned managed identity enabled
Ruby	Create Azure VM with an system-assigned identity enabled

Next steps

- See related articles under **Configure Identity for an Azure VM**, to learn how you can also use the Azure portal, PowerShell, CLI, and resource templates.

Configure managed identities for Azure resources on a virtual machine scale set using the Azure portal

3/26/2019 • 3 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, using PowerShell, you learn how to perform the following managed identities for Azure resources operations on a virtual machine scale set:

- If you're unfamiliar with managed identities for Azure resources, check out the overview section.
- If you don't already have an Azure account, sign up for a free account before continuing.
- To perform the management operations in this article, your account needs the following Azure role based access control assignments:

  NOTE

  No additional Azure AD directory role assignments required.

  - Virtual Machine Contributor to enable and remove system-assigned managed identity from a virtual machine scale set.

System-assigned managed identity

In this section, you will learn how to enable and disable the system-assigned managed identity using the Azure portal.

Enable system-assigned managed identity during creation of a virtual machine scale set

Currently, the Azure portal does not support enabling system-assigned managed identity during the creation of a virtual machine scale set. Instead, refer to the following virtual machine scale set creation Quickstart article to first create a virtual machine scale set, and then proceed to the next section for details on enabling system-assigned managed identity on a virtual machine scale set:

- Create a Virtual Machine Scale Set in the Azure portal

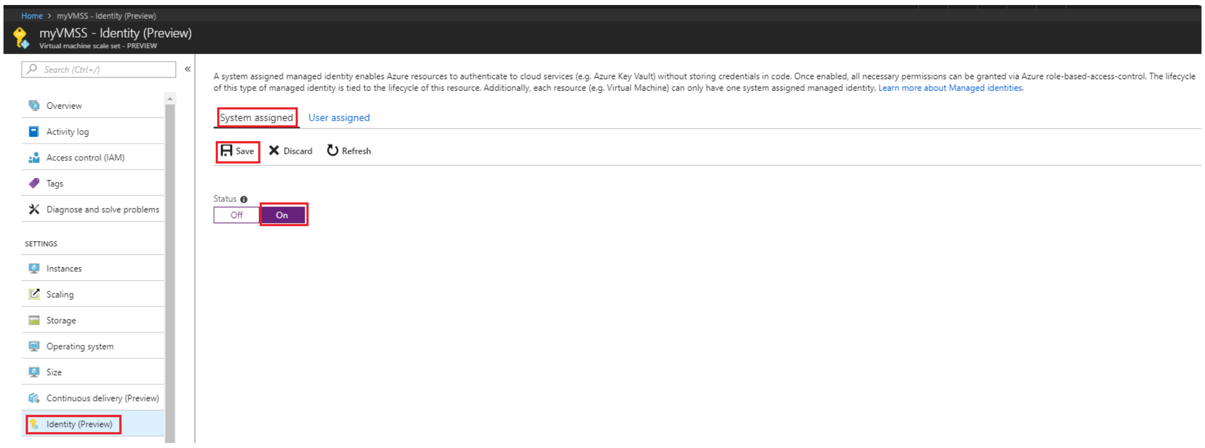
Enable system-assigned managed identity on an existing virtual machine scale set

To enable the system-assigned managed identity on a virtual machine scale set that was originally provisioned without it:

1. Sign in to the Azure portal using an account associated with the Azure subscription that contains the virtual machine scale set.

2. Navigate to the desired virtual machine scale set.

3. Under **System assigned**, **Status**, select **On** and then click **Save**:


A system assigned managed identity enables Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. Once enabled, all necessary permissions can be granted via Azure role-based-access-control. The lifecycle of this type of managed identity is tied to the lifecycle of this resource. Additionally, each resource (e.g. Virtual Machine) can only have one system assigned managed identity. Learn more about Managed identities.

Status: Off [✓ On]

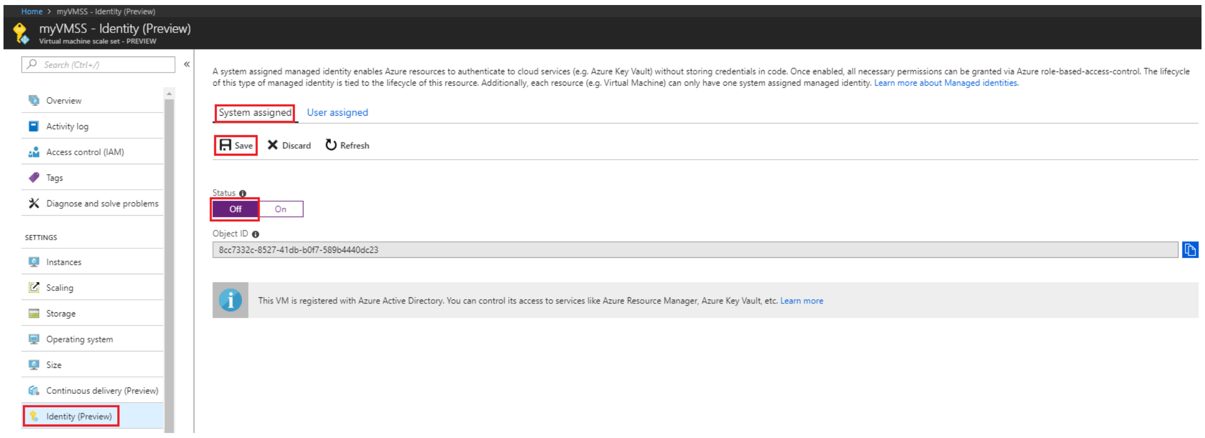
Remove system-assigned managed identity from a virtual machine scale set

If you have a virtual machine scale set that no longer needs a system-assigned managed identity:

1. Sign in to the Azure portal using an account associated with the Azure subscription that contains the virtual machine scale set. Also make sure your account belongs to a role that gives you write permissions on the virtual machine scale set.

2. Navigate to the desired virtual machine scale set.

3. Under **System assigned**, **Status**, select **Off** and then click **Save**:


A system assigned managed identity enables Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. Once enabled, all necessary permissions can be granted via Azure role-based-access-control. The lifecycle of this type of managed identity is tied to the lifecycle of this resource. Additionally, each resource (e.g. Virtual Machine) can only have one system assigned managed identity. Learn more about Managed identities.

Status: [✓ Off] On

Object ID: 8cc7332c-8527-41db-b0f7-589b4440dc23

This VM is registered with Azure Active Directory. You can control its access to services like Azure Resource Manager, Azure Key Vault, etc. Learn more

User-assigned managed identity

In this section, you learn how to add and remove a user-assigned managed identity from a virtual machine scale set using the Azure portal.

Assign a user-assigned managed identity during the creation of a virtual machine scale set

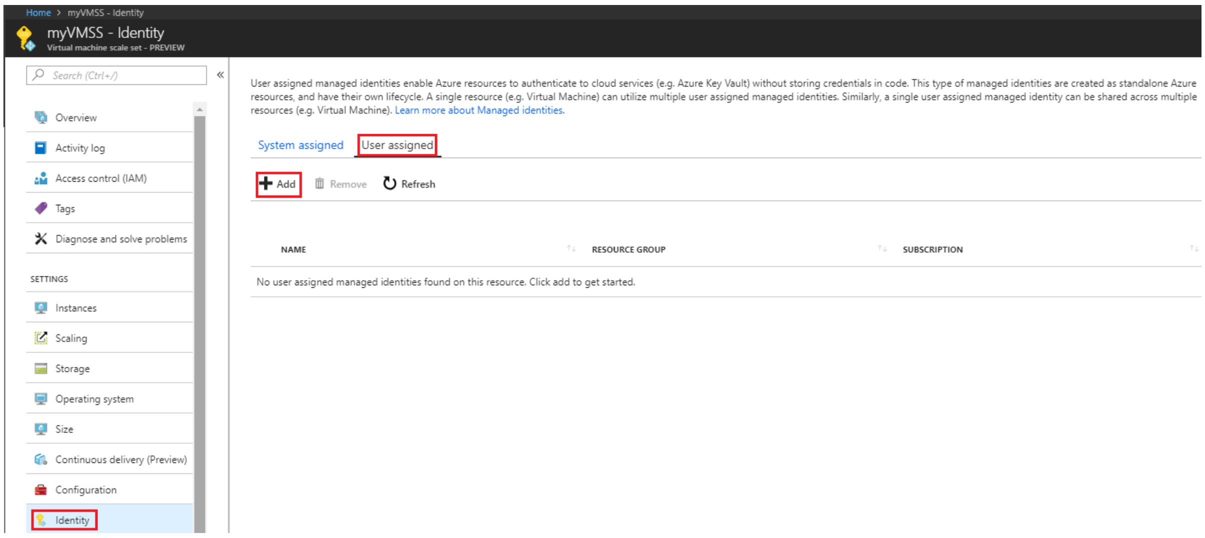
Currently, the Azure portal does not support assigning a user-assigned managed identity during the creation of a virtual machine scale set. Instead, refer to the following virtual machine scale set creation Quickstart article to first create a virtual machine scale set, and then proceed to the next section for details on assigning a user-assigned managed identity to it:

- Create a Virtual Machine Scale Set in the Azure portal

Assign a user-assigned managed identity to an existing virtual machine scale set

1. Sign in to the Azure portal using an account associated with the Azure subscription that contains the virtual machine scale set.

2. Navigate to the desired virtual machine scale set and click **Identity**, **User assigned** and then **+Add**.

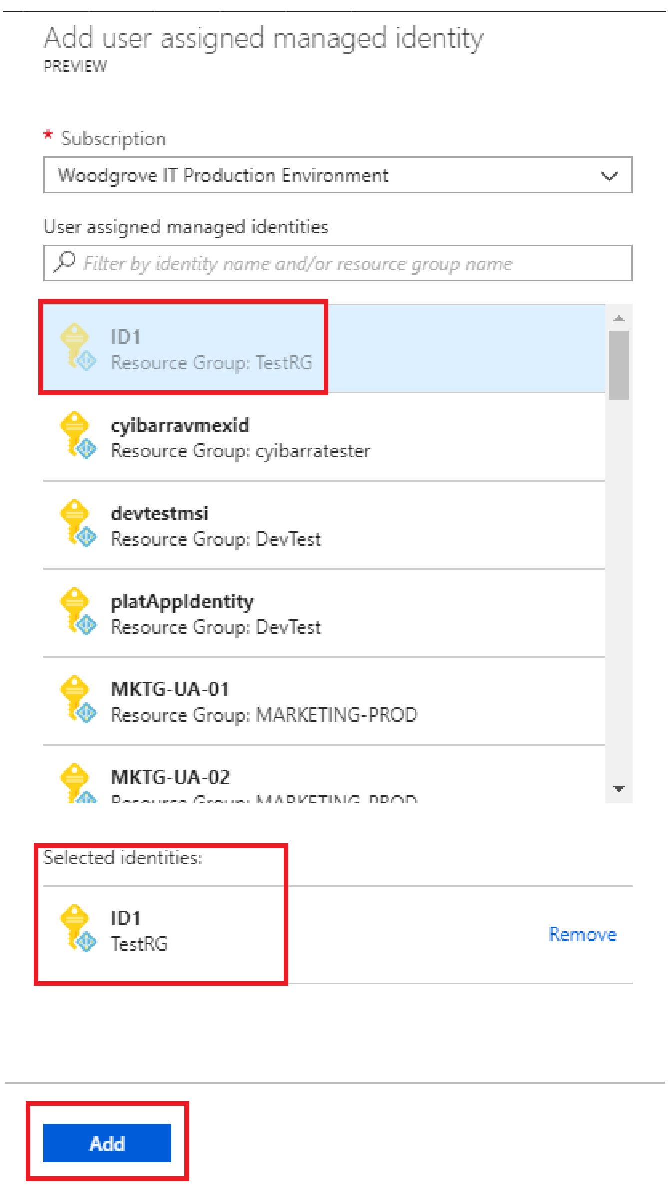


User assigned managed identities enable Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. This type of managed identities are created as standalone Azure resources, and have their own lifecycle. A single resource (e.g. Virtual Machine) can utilize multiple user assigned managed identities. Similarly, a single user assigned managed identity can be shared across multiple resources (e.g. Virtual Machine). Learn more about Managed identities.

NAME	RESOURCE GROUP	SUBSCRIPTION

No user assigned managed identities found on this resource. Click add to get started.

3. Click the user-assigned identity you want to add to the virtual machine scale set and then click **Add**.

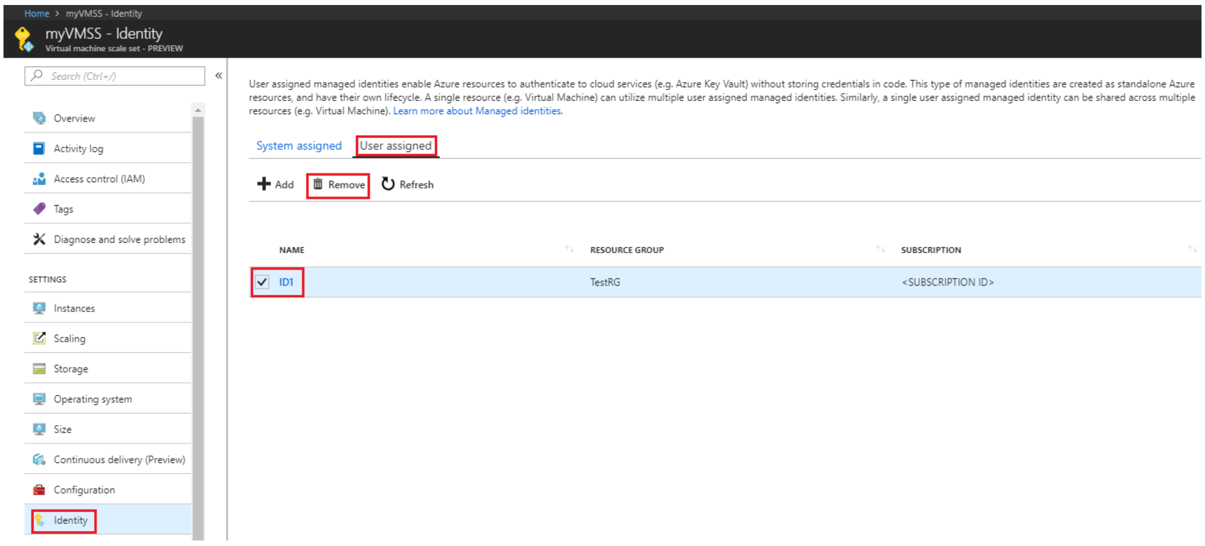

Add user assigned managed identity
PREVIEW
* Subscription
Woodgrove IT Production Environment
User assigned managed identities
Filter by identity name and/or resource group name
ID1
Resource Group: TestRG
cyibarravmexid
Resource Group: cyibarratester
devtestmsi
Resource Group: DevTest
platAppIdentity
Resource Group: DevTest
MKTG-UA-01
Resource Group: MARKETING-PROD
MKTG-UA-02
Resource Group: MARKETING-PROD
Selected identities:
ID1
TestRG
Remove
Add

Remove a user-assigned managed identity from a virtual machine scale set

1. Sign in to the Azure portal using an account associated with the Azure subscription that contains the VM.

2. Navigate to the desired virtual machine scale set and click **Identity**, **User assigned**, the name of the user-

assigned managed identity you want to delete and then click **Remove** (click **Yes** in the confirmation pane).



User assigned managed identities enable Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. This type of managed identities are created as standalone Azure resources, and have their own lifecycle. A single resource (e.g. Virtual Machine) can utilize multiple user assigned managed identities. Similarly, a single user assigned managed identity can be shared across multiple resources (e.g. Virtual Machine). Learn more about Managed identities.

NAME	RESOURCE GROUP	SUBSCRIPTION
✓ ID1	TestRG	<SUBSCRIPTION ID>

Next steps

- Using the Azure portal, give an Azure virtual machine scale set managed identity access to another Azure resource.

Configure managed identities for Azure resources on a virtual machine scale set using Azure CLI

3/26/2019 • 8 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, you learn how to perform the following managed identities for Azure resources operations on an Azure virtual machine scale set, using the Azure CLI:

- Enable and disable the system-assigned managed identity on an Azure virtual machine scale set
- Add and remove a user-assigned managed identity on an Azure virtual machine scale set

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- To perform the management operations in this article, your account needs the following Azure role based access control assignments:

  > **NOTE**
  >
  > No additional Azure AD directory role assignments required.

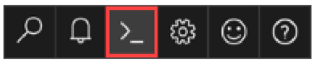
  - Virtual Machine Contributor to create a virtual machine scale set and enable and remove system and/or user-assigned managed identity from a virtual machine scale set.
  - Managed Identity Contributor role to create a user-assigned managed identity.
  - Managed Identity Operator role to assign and remove a user-assigned managed identity from and to a virtual machine scale set.
- To run the CLI script examples, you have three options:
  - Use Azure Cloud Shell from the Azure portal (see next section).
  - Use the embedded Azure Cloud Shell via the "Try It" button, located in the top right corner of each code block.
  - Install the latest version of the Azure CLI (2.0.13 or later) if you prefer to use a local CLI console.

NOTE

The commands have been updated to reflect the latest release of the Azure CLI.

Open Azure Cloud Shell

Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Select **Copy** to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.	
Open Cloud Shell in your browser.	
Select the **Cloud Shell** button on the menu in the upper-right corner of the Azure portal.	

System-assigned managed identity

In this section, you learn how to enable and disable the system-assigned managed identity for an Azure virtual machine scale set using Azure CLI.

Enable system-assigned managed identity during creation of an Azure virtual machine scale set

To create a virtual machine scale set with the system-assigned managed identity enabled:

1. If you're using the Azure CLI in a local console, first sign in to Azure using az login. Use an account that is associated with the Azure subscription under which you would like to deploy the virtual machine scale set:

   ```
   az login
   ```

2. Create a resource group for containment and deployment of your virtual machine scale set and its related resources, using az group create. You can skip this step if you already have a resource group you would like to use instead:

   ```
   az group create --name myResourceGroup --location westus
   ```

3. Create a virtual machine scale set using az vmss create . The following example creates a virtual machine scale set named *myVMSS* with a system-assigned managed identity, as requested by the `--assign-identity` parameter. The `--admin-username` and `--admin-password` parameters specify the administrative user name and password account for virtual machine sign-in. Update these values as appropriate for your environment:

   ```
   az vmss create --resource-group myResourceGroup --name myVMSS --image win2016datacenter --upgrade-
   policy-mode automatic --custom-data cloud-init.txt --admin-username azureuser --admin-password
   myPassword12 --assign-identity --generate-ssh-keys
   ```

Enable system-assigned managed identity on an existing Azure virtual machine scale set

If you need to enable the system-assigned managed identity on an existing Azure virtual machine scale set:

1. If you're using the Azure CLI in a local console, first sign in to Azure using az login. Use an account that is associated with the Azure subscription that contains the virtual machine scale set.

   ```
   az login
   ```

2. Use az vmss identity assign command to enable a system-assigned managed identity to an existing VM:

   ```
   az vmss identity assign -g myResourceGroup -n myVMSS
   ```

Disable system-assigned managed identity from an Azure virtual machine scale set

If you have a virtual machine scale set that no longer needs the system-assigned managed identity, but still needs user-assigned managed identities, use the following command:

```
az vmss update -n myVM -g myResourceGroup --set identity.type='UserAssigned'
```

If you have a virtual machine that no longer needs system-assigned managed identity and it has no user-assigned managed identities, use the following command:

NOTE

The value `none` is case sensitive. It must be lowercase.

```
az vmss update -n myVM -g myResourceGroup --set identity.type="none"
```

NOTE

If you have provisioned the managed identity for Azure resources VM extension (to be deprecated), you need to remove it using az vmss extension delete. For more information, see Migrate from VM extension to Azure IMDS for authentication.

user-assigned managed identity

In this section, you learn how to enable and remove a user-assigned managed identity using Azure CLI.

Assign a user-assigned managed identity during the creation of a virtual machine scale set

This section walks you through creation of a virtual machine scale set and assignment of a user-assigned managed identity to the virtual machine scale set. If you already have a virtual machine scale set you want to use, skip this section and proceed to the next.

1. You can skip this step if you already have a resource group you would like to use. Create a resource group for containment and deployment of your user-assigned managed identity, using az group create. Be sure to replace the `<RESOURCE GROUP>` and `<LOCATION>` parameter values with your own values. :

   ```
   az group create --name <RESOURCE GROUP> --location <LOCATION>
   ```

2. Create a user-assigned managed identity using az identity create. The `-g` parameter specifies the resource group where the user-assigned managed identity is created, and the `-n` parameter specifies its name. Be sure to replace the `<RESOURCE GROUP>` and `<USER ASSIGNED IDENTITY NAME>` parameter values with your own

values:

> **IMPORTANT**
>
> When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

```
az identity create -g <RESOURCE GROUP> -n <USER ASSIGNED IDENTITY NAME>
```

The response contains details for the user-assigned managed identity created, similar to the following. The resource `id` value assigned to the user-assigned managed identity is used in the following step.

```
{
     "clientId": "73444643-8088-4d70-9532-c3a0fdc190fz",
     "clientSecretUrl": "https://control-westcentralus.identity.azure.net/subscriptions/<SUBSCRIPTON
ID>/resourcegroups/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER
ASSIGNED IDENTITY NAME>/credentials?tid=5678&oid=9012&aid=73444643-8088-4d70-9532-c3a0fdc190fz",
     "id": "/subscriptions/<SUBSCRIPTON ID>/resourcegroups/<RESOURCE
GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>",
     "location": "westcentralus",
     "name": "<USER ASSIGNED IDENTITY NAME>",
     "principalId": "e5fdfdc1-ed84-4d48-8551-fe9fb9dedfll",
     "resourceGroup": "<RESOURCE GROUP>",
     "tags": {},
     "tenantId": "733a8f0e-ec41-4e69-8ad8-971fc4b533bl",
     "type": "Microsoft.ManagedIdentity/userAssignedIdentities"
}
```

3. Create a virtual machine scale set using az vmss create. The following example creates a virtual machine scale set associated with the new user-assigned managed identity, as specified by the `--assign-identity` parameter. Be sure to replace the `<RESOURCE GROUP>`, `<VMSS NAME>`, `<USER NAME>`, `<PASSWORD>`, and `<USER ASSIGNED IDENTITY>` parameter values with your own values.

```
az vmss create --resource-group <RESOURCE GROUP> --name <VMSS NAME> --image UbuntuLTS --admin-username
<USER NAME> --admin-password <PASSWORD> --assign-identity <USER ASSIGNED IDENTITY>
```

Assign a user-assigned managed identity to an existing virtual machine scale set

1. Create a user-assigned managed identity using az identity create. The `-g` parameter specifies the resource group where the user-assigned managed identity is created, and the `-n` parameter specifies its name. Be sure to replace the `<RESOURCE GROUP>` and `<USER ASSIGNED IDENTITY NAME>` parameter values with your own values:

```
az identity create -g <RESOURCE GROUP> -n <USER ASSIGNED IDENTITY NAME>
```

The response contains details for the user-assigned managed identity created, similar to the following.

```
{
      "clientId": "73444643-8088-4d70-9532-c3a0fdc190fz",
      "clientSecretUrl": "https://control-westcentralus.identity.azure.net/subscriptions/<SUBSCRIPTON ID>/resourcegroups/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY >/credentials?tid=5678&oid=9012&aid=73444643-8088-4d70-9532-c3a0fdc190fz",
      "id": "/subscriptions/<SUBSCRIPTON ID>/resourcegroups/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY>",
      "location": "westcentralus",
      "name": "<USER ASSIGNED IDENTITY>",
      "principalId": "e5fdfdc1-ed84-4d48-8551-fe9fb9dedfll",
      "resourceGroup": "<RESOURCE GROUP>",
      "tags": {},
      "tenantId": "733a8f0e-ec41-4e69-8ad8-971fc4b533bl",
      "type": "Microsoft.ManagedIdentity/userAssignedIdentities"
}
```

2. Assign the user-assigned managed identity to your virtual machine scale set using az vmss identity assign. Be sure to replace the `<RESOURCE GROUP>` and `<VIRTUAL MACHINE SCALE SET NAME>` parameter values with your own values. The `<USER ASSIGNED IDENTITY>` is the user-assigned identity's resource `name` property, as created in the previous step:

```
az vmss identity assign -g <RESOURCE GROUP> -n <VIRTUAL MACHINE SCALE SET NAME> --identities <USER ASSIGNED IDENTITY>
```

Remove a user-assigned managed identity from an Azure virtual machine scale set

To remove a user-assigned managed identity from a virtual machine scale set use az vmss identity remove. If this is the only user-assigned managed identity assigned to the virtual machine scale set, `UserAssigned` will be removed from the identity type value. Be sure to replace the `<RESOURCE GROUP>` and `<VIRTUAL MACHINE SCALE SET NAME>` parameter values with your own values. The `<USER ASSIGNED IDENTITY>` will be the user-assigned managed identity's `name` property, which can be found in the identity section of the virtual machine scale set using `az vmss identity show`:

```
az vmss identity remove -g <RESOURCE GROUP> -n <VIRTUAL MACHINE SCALE SET NAME> --identities <USER ASSIGNED IDENTITY>
```

If your virtual machine scale set does not have a system-assigned managed identity and you want to remove all user-assigned managed identities from it, use the following command:

NOTE

The value `none` is case sensitive. It must be lowercase.

```
az vmss update -n myVMSS -g myResourceGroup --set identity.type="none" identity.userAssignedIdentities=null
```

If your virtual machine scale set has both system-assigned and user-assigned managed identities, you can remove all the user-assigned identities by switching to use only system-assigned managed identity. Use the following command:

```
az vmss update -n myVMSS -g myResourceGroup --set identity.type='SystemAssigned' identity.userAssignedIdentities=null
```

Next steps

- Managed identities for Azure resources overview
- For the full Azure virtual machine scale set creation Quickstart, see:
    - Create a Virtual Machine Scale Set with CLI

Configure managed identities for Azure resources on virtual machine scale sets using PowerShell

3/26/2019 • 6 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, using PowerShell, you learn how to perform the managed identities for Azure resources operations on a virtual machine scale set:

- Enable and disable the system-assigned managed identity on a virtual machine scale set
- Add and remove a user-assigned managed identity on a virtual machine scale set

NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user managed assigned identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- To perform the management operations in this article, your account needs the following Azure role based access control assignments:

  NOTE

  No additional Azure AD directory role assignments required.

  - Virtual Machine Contributor to create a virtual machine scale set and enable and remove system-assigned managed and/or user-assigned managed identity from a virtual machine scale set.
  - Managed Identity Contributor role to create a user-assigned managed identity.
  - Managed Identity Operator role to assign and remove a user-assigned managed identity from and to a virtual machine scale set.
- Install the latest version of Azure PowerShell if you haven't already.

System-assigned managed identity

In this section, you learn how to enable and remove a system-assigned managed identity using Azure PowerShell.

Enable system-assigned managed identity during the creation of an Azure virtual machine scale set

To create a virtual machine scale set with the system-assigned managed identity enabled:

1. Refer to *Example 1* in the New-AzVmssConfig cmdlet reference article to create a virtual machine scale set with a system-assigned managed identity. Add the parameter `-IdentityType SystemAssigned` to the `New-AzVmssConfig` cmdlet:

   ```
   $VMSS = New-AzVmssConfig -Location $Loc -SkuCapacity 2 -SkuName "Standard_A0" -UpgradePolicyMode
   "Automatic" -NetworkInterfaceConfiguration $NetCfg -IdentityType SystemAssigned`
   ```

NOTE

You may optionally provision the managed identities for Azure resources virtual machine scale set extension, but it will soon be deprecated. We recommend using the Azure Instance Metadata identity endpoint for authentication. For more information, see Stop using the VM extension and start using the Azure IMDS endpoint for authentication.

Enable system-assigned managed identity on an existing Azure virtual machine scale set

If you need to enable a system-assigned managed identity on an existing Azure virtual machine scale set:

1. Sign in to Azure using `Connect-AzAccount`. Use an account that is associated with the Azure subscription that contains the virtual machine scale set. Also make sure your account belongs to a role that gives you write permissions on the virtual machine scale set, such as “Virtual Machine Contributor”:

   ```
   Connect-AzAccount
   ```

2. First retrieve the virtual machine scale set properties using the `Get-AzVmss` cmdlet. Then to enable a system-assigned managed identity, use the `-IdentityType` switch on the Update-AzVmss cmdlet:

   ```
   Update-AzVmss -ResourceGroupName myResourceGroup -Name -myVmss -IdentityType "SystemAssigned"
   ```

NOTE

You may optionally provision the managed identities for Azure resources virtual machine scale set extension, but it will soon be deprecated. We recommend using the Azure Instance Metadata identity endpoint for authentication. For more information, see Migrate from the VM extension to Azure IMDS endpoint for authentication.

Disable the system-assigned managed identity from an Azure virtual machine scale set

If you have a virtual machine scale set that no longer needs the system-assigned managed identity but still needs user-assigned managed identities, use the following cmdlet:

1. Sign in to Azure using `Connect-AzAccount`. Use an account that is associated with the Azure subscription that contains the VM. Also make sure your account belongs to a role that gives you write permissions on the virtual machine scale set, such as “Virtual Machine Contributor”:

2. Run the following cmdlet:

```
Update-AzVmss -ResourceGroupName myResourceGroup -Name myVmss -IdentityType "UserAssigned"
```

If you have a virtual machine scale set that no longer needs system-assigned managed identity and it has no user-assigned managed identities, use the following commands:

```
Update-AzVmss -ResourceGroupName myResourceGroup -Name myVmss -IdentityType None
```

User-assigned managed identity

In this section, you learn how to add and remove a user-assigned managed identity from a virtual machine scale set using Azure PowerShell.

Assign a user-assigned managed identity during creation of an Azure virtual machine scale set

Creating a new virtual machine scale set with a user-assigned managed identity isn't currently supported via PowerShell. See the next section on how to add a user-assigned managed identity to an existing virtual machine scale set. Check back for updates.

Assign a user-assigned managed identity to an existing Azure virtual machine scale set

To assign a user-assigned managed identity to an existing Azure virtual machine scale set:

1. Sign in to Azure using `Connect-AzAccount`. Use an account that is associated with the Azure subscription that contains the virtual machine scale set. Also make sure your account belongs to a role that gives you write permissions on the virtual machine scale set, such as “Virtual Machine Contributor”:

   ```
   Connect-AzAccount
   ```

2. First retrieve the virtual machine scale set properties using the `Get-AzVM` cmdlet. Then to assign a user-assigned managed identity to the virtual machine scale set, use the `-IdentityType` and `-IdentityID` switch on the Update-AzVmss cmdlet. Replace `<VM NAME>`, `<SUBSCRIPTION ID>`, `<RESROURCE GROUP>`, `<USER ASSIGNED ID1>`, `USER ASSIGNED ID2` with your own values.

   IMPORTANT

   When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

   ```
   Update-AzVmss -ResourceGroupName <RESOURCE GROUP> -Name <VMSS NAME> -IdentityType UserAssigned -
   IdentityID "<USER ASSIGNED ID1>","<USER ASSIGNED ID2>"
   ```

Remove a user-assigned managed identity from an Azure virtual machine scale set

If your virtual machine scale set has multiple user-assigned managed identities, you can remove all but the last one using the following commands. Be sure to replace the `<RESOURCE GROUP>` and `<VIRTUAL MACHINE SCALE SET NAME>` parameter values with your own values. The `<USER ASSIGNED IDENTITY NAME>` is the user-assigned managed identity's name property, which should remain on the virtual machine scale set. This information can be found in the identity section of the virtual machine scale set using `az vmss show`:

```
Update-AzVmss -ResourceGroupName myResourceGroup -Name myVmss -IdentityType UserAssigned -IdentityID "<USER
ASSIGNED IDENTITY NAME>"
```

If your virtual machine scale set does not have a system-assigned managed identity and you want to remove all user-assigned managed identities from it, use the following command:

```
Update-AzVmss -ResourceGroupName myResourceGroup -Name myVmss -IdentityType None
```

If your virtual machine scale set has both system-assigned and user-assigned managed identities, you can remove all the user-assigned managed identities by switching to use only system-assigned managed identity.

```
Update-AzVmss -ResourceGroupName myResourceGroup -Name myVmss -IdentityType "SystemAssigned"
```

Next steps

- Managed identities for Azure resources overview
- For the full Azure VM creation Quickstarts, see:
  - Create a Windows virtual machine with PowerShell
  - Create a Linux virtual machine with PowerShell

Configure managed identities for Azure resources on an Azure virtual machine scale using a template

4/4/2019 • 7 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, you learn how to perform the following managed identities for Azure resources operations on an Azure virtual machine scale set, using Azure Resource Manager deployment template:

- Enable and disable the system-assigned managed identity on an Azure virtual machine scale set
- Add and remove a user-assigned managed identity on an Azure virtual machine scale set

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- To perform the management operations in this article, your account needs the following Azure role based access control assignments:

  NOTE

  No additional Azure AD directory role assignments required.

  - Virtual Machine Contributor to create a virtual machine scale set and enable and remove system and/or user-assigned managed identity from a virtual machine scale set.
  - Managed Identity Contributor role to create a user-assigned managed identity.
  - Managed Identity Operator role to assign and remove a user-assigned managed identity from and to a virtual machine scale set.

Azure Resource Manager templates

As with the Azure portal and scripting, Azure Resource Manager templates provide the ability to deploy new or modified resources defined by an Azure resource group. Several options are available for template editing and deployment, both local and portal-based, including:

- Using a custom template from the Azure Marketplace, which allows you to create a template from scratch, or base it on an existing common or quickstart template.
- Deriving from an existing resource group, by exporting a template from either the original deployment, or from

the current state of the deployment.

- Using a local JSON editor (such as VS Code), and then uploading and deploying by using PowerShell or CLI.
- Using the Visual Studio Azure Resource Group project to both create and deploy a template.

Regardless of the option you choose, template syntax is the same during initial deployment and redeployment. Enabling managed identities for Azure resources on a new or existing VM is done in the same manner. Also, by default, Azure Resource Manager does an incremental update to deployments.

System-assigned managed identity

In this section, you will enable and disable the system-assigned managed identity using an Azure Resource Manager template.

Enable system-assigned managed identity during creation the creation of a virtual machines scale set or an existing virtual machine scale set

1. Whether you sign in to Azure locally or via the Azure portal, use an account that is associated with the Azure subscription that contains the virtual machine scale set.

2. To enable the system-assigned managed identity, load the template into an editor, locate the `Microsoft.Compute/virtualMachinesScaleSets` resource of interest within the resources section and add the `identity` property at the same level as the `"type": "Microsoft.Compute/virtualMachinesScaleSets"` property. Use the following syntax:

```
"identity": {
    "type": "SystemAssigned"
}
```

> **NOTE**
>
> You may optionally provision the managed identities for Azure resources virtual machine scale set extension by specifying it in the `extensionProfile` element of the template. This step is optional as you can use the Azure Instance Metadata Service (IMDS) identity endpoint, to retrieve tokens as well. For more information, see Migrate from VM extension to Azure IMDS for authentication.

4. When you're done, the following sections should added to the resource section of your template and should resemble the following:

```
"resources": [
    {
        //other resource provider properties...
        "apiVersion": "2018-06-01",
        "type": "Microsoft.Compute/virtualMachineScaleSets",
        "name": "[variables('vmssName')]",
        "location": "[resourceGroup().location]",
        "identity": {
            "type": "SystemAssigned",
        },
       "properties": {
            //other resource provider properties...
            "virtualMachineProfile": {
                //other virtual machine profile properties...
                //The following appears only if you provisioned the optional virtual machine scale set
extension (to be deprecated)
                "extensionProfile": {
                    "extensions": [
                        {
                            "name": "ManagedIdentityWindowsExtension",
                            "properties": {
                              "publisher": "Microsoft.ManagedIdentity",
                              "type": "ManagedIdentityExtensionForWindows",
                              "typeHandlerVersion": "1.0",
                              "autoUpgradeMinorVersion": true,
                              "settings": {
                                  "port": 50342
                              }
                            }
                        }
                    ]
                }
            }
        }
    }
]
```

Disable a system-assigned managed identity from an Azure virtual machine scale set

If you have a virtual machine scale set that no longer needs a system-assigned managed identity:

1. Whether you sign in to Azure locally or via the Azure portal, use an account that is associated with the Azure subscription that contains the virtual machine scale set.

2. Load the template into an editor and locate the `Microsoft.Compute/virtualMachineScaleSets` resource of interest within the `resources` section. If you have a VM that only has system-assigned managed identity, you can disable it by changing the identity type to `None`.

   Microsoft.Compute/virtualMachineScaleSets API version 2018-06-01

   If your apiVersion is `2018-06-01` and your VM has both system and user-assigned managed identities, remove `SystemAssigned` from the identity type and keep `UserAssigned` along with the userAssignedIdentities dictionary values.

   Microsoft.Compute/virtualMachineScaleSets API version 2018-06-01

   If your apiVersion is `2017-12-01` and your virtual machine scale set has both system and user-assigned managed identities, remove `SystemAssigned` from the identity type and keep `UserAssigned` along with the `identityIds` array of the user-assigned managed identities.

   The following example shows you how remove a system-assigned managed identity from a virtual machine scale set with no user-assigned managed identities:

```
{
    "name": "[variables('vmssName')]",
    "apiVersion": "2018-06-01",
    "location": "[parameters(Location')]",
    "identity": {
        "type": "None"
     }

}
```

User-assigned managed identity

In this section, you assign a user-assigned managed identity to a virtual machine scale set using Azure Resource Manager template.

> **NOTE**
>
> To create a user-assigned managed identity using an Azure Resource Manager Template, see Create a user-assigned managed identity.

Assign a user-assigned managed identity to a virtual machine scale set

1. Under the `resources` element, add the following entry to assign a user-assigned managed identity to your virtual machine scale set. Be sure to replace `<USERASSIGNEDIDENTITY>` with the name of the user-assigned managed identity you created.

   Microsoft.Compute/virtualMachineScaleSets API version 2018-06-01

   If your apiVersion is `2018-06-01`, your user-assigned managed identities are stored in the `userAssignedIdentities` dictionary format and the `<USERASSIGNEDIDENTITYNAME>` value must be stored in a variable defined in the `variables` section of your template.

   ```
   {
       "name": "[variables('vmssName')]",
       "apiVersion": "2018-06-01",
       "location": "[parameters(Location')]",
       "identity": {
           "type": "userAssigned",
           "userAssignedIdentities": {
               "
   [resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',variables('<USERASSIGNEDIDENTITYNAME>'))
   ]": {}
           }
       }

   }
   ```

   Microsoft.Compute/virtualMachineScaleSets API version 2017-12-01

   If your `apiVersion` is `2017-12-01` or earlier, your user-assigned managed identities are stored in the `identityIds` array and the `<USERASSIGNEDIDENTITYNAME>` value must be stored in a variable defined in the variables section of your template.

```
{
    "name": "[variables('vmssName')]",
    "apiVersion": "2017-03-30",
    "location": "[parameters(Location')]",
    "identity": {
        "type": "userAssigned",
        "identityIds": [
            "
[resourceID('Micrososft.ManagedIdentity/userAssignedIdentities/',variables('<USERASSIGNEDIDENTITY>'))]"
        ]
    }

}
```

NOTE

You may optionally provision the managed identities for Azure resources virtual machine scale set extension by specifying it in the `extensionProfile` element of the template. This step is optional as you can use the Azure Instance Metadata Service (IMDS) identity endpoint, to retrieve tokens as well. For more information, see Migrate from VM extension to Azure IMDS for authentication.

3. When you are done, your template should look similar to the following:

   Microsoft.Compute/virtualMachineScaleSets API version 2018-06-01

```
"resources": [
     {
         //other resource provider properties...
         "apiVersion": "2018-06-01",
         "type": "Microsoft.Compute/virtualMachineScaleSets",
         "name": "[variables('vmssName')]",
         "location": "[resourceGroup().location]",
         "identity": {
             "type": "UserAssigned",
             "userAssignedIdentities": {
                 "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',variables('<USERASSIGNEDIDENTITYNAME>'))
]": {}
             }
         },
        "properties": {
             //other virtual machine properties...
             "virtualMachineProfile": {
                 //other virtual machine profile properties...
                 //The following appears only if you provisioned the optional virtual machine scale set
extension (to be deprecated)
                 "extensionProfile": {
                     "extensions": [
                         {
                             "name": "ManagedIdentityWindowsExtension",
                             "properties": {
                               "publisher": "Microsoft.ManagedIdentity",
                               "type": "ManagedIdentityExtensionForWindows",
                               "typeHandlerVersion": "1.0",
                               "autoUpgradeMinorVersion": true,
                               "settings": {
                                   "port": 50342
                               }
                             }
                         }
                     ]
                 }
             }
         }
     }
 ]
```

Microsoft.Compute/virtualMachines API version 2017-12-01

```
"resources": [
     {
         //other resource provider properties...
         "apiVersion": "2017-12-01",
         "type": "Microsoft.Compute/virtualMachineScaleSets",
         "name": "[variables('vmssName')]",
         "location": "[resourceGroup().location]",
         "identity": {
             "type": "UserAssigned",
             "identityIds": [
                 "
[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',variables('<USERASSIGNEDIDENTITYNAME>'))
]"
             ]
         },
        "properties": {
             //other virtual machine properties...
             "virtualMachineProfile": {
                 //other virtual machine profile properties...
                 //The following appears only if you provisioned the optional virtual machine scale set
extension (to be deprecated)
                 "extensionProfile": {
                     "extensions": [
                         {
                             "name": "ManagedIdentityWindowsExtension",
                             "properties": {
                               "publisher": "Microsoft.ManagedIdentity",
                               "type": "ManagedIdentityExtensionForWindows",
                               "typeHandlerVersion": "1.0",
                               "autoUpgradeMinorVersion": true,
                               "settings": {
                                   "port": 50342
                               }
                             }
                         }
                     ]
                 }
             }
         }
     }
 ]
```

Remove user-assigned managed identity from an Azure virtual machine scale set

If you have a virtual machine scale set that no longer needs a user-assigned managed identity:

1. Whether you sign in to Azure locally or via the Azure portal, use an account that is associated with the Azure subscription that contains the virtual machine scale set.

2. Load the template into an editor and locate the `Microsoft.Compute/virtualMachineScaleSets` resource of interest within the `resources` section. If you have a virtual machine scale set that only has user-assigned managed identity, you can disable it by changing the identity type to `None`.

   The following example shows you how remove all user-assigned managed identities from a VM with no system-assigned managed identities:

```
{
    "name": "[variables('vmssName')]",
    "apiVersion": "2018-06-01",
    "location": "[parameters(Location')]",
    "identity": {
        "type": "None"
     }
}
```

Microsoft.Compute/virtualMachineScaleSets API version 2018-06-01

To remove a single user-assigned managed identity from a virtual machine scale set, remove it from the `userAssignedIdentities` dictionary.

If you have a system-assigned identity, keep it in the in the `type` value under the `identity` value.

Microsoft.Compute/virtualMachineScaleSets API version 2017-12-01

To remove a single user-assigned managed identity from a virtual machine scale set, remove it from the `identityIds` array.

If you have a system-assigned managed identity, keep it in the in the `type` value under the `identity` value.

Next steps

- Managed identities for Azure resources overview.

Configure managed identities for Azure resources on a virtual machine scale set using REST API calls

4/4/2019 • 15 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed system identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, using CURL to make calls to the Azure Resource Manager REST endpoint, you learn how to perform the following managed identities for Azure resources operations on an virtual machine scale set:

- Enable and disable the system-assigned managed identity on an Azure virtual machine scale set
- Add and remove a user-assigned managed identity on an Azure virtual machine scale set

Prerequisites

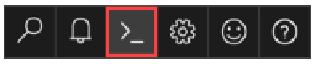
- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- To perform the management operations in this article, your account needs the following Azure role based access control assignments:

  > **NOTE**
  >
  > No additional Azure AD directory role assignments required.

  - Virtual Machine Contributor to create a virtual machine scale set and enable and remove system and/or user-assigned managed identity from an virtual machine scale set.
  - Managed Identity Contributor role to create a user-assigned managed identity.
  - Managed Identity Operator role to assign and remove a user-assigned identity from and to a virtual machine scale set.
- If you are using Windows, install the Windows Subsystem for Linux or use the Azure Cloud Shell in the Azure portal.
- Install the Azure CLI local console, if you use the Windows Subsystem for Linux or a Linux distribution OS.
- If you are using Azure CLI local console, sign in to Azure using `az login` with an account that is associated with the Azure subscription you would like to manage system or user-assigned managed identities.

Open Azure Cloud Shell

Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Select **Copy** to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.	
Open Cloud Shell in your browser.	
Select the **Cloud Shell** button on the menu in the upper-right corner of the Azure portal.	

System-assigned managed identity

In this section, you learn how to enable and disable system-assigned managed identity on a virtual machine scale set using CURL to make calls to the Azure Resource Manager REST endpoint.

Enable system-assigned managed identity during creation of a virtual machine scale set

To create a virtual machine scale set with system-assigned managed identity enabled, you need create a virtual machine scale set and retrieve an access token to use CURL to call the Resource Manager endpoint with the system-assigned managed identity type value.

1. Create a resource group for containment and deployment of your virtual machine scale set and its related resources, using az group create. You can skip this step if you already have resource group you would like to use instead:

   ```
   az group create --name myResourceGroup --location westus
   ```

2. Create a network interface for your virtual machine scale set:

   ```
    az network nic create -g myResourceGroup --vnet-name myVnet --subnet mySubnet -n myNic
   ```

3. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your virtual machine scale set with a system-assigned managed identity.

   ```
   az account get-access-token
   ```

4. Create a virtual machine scale set using CURL to call the Azure Resource Manager REST endpoint. The following example creates a virtual machine scale set named *myVMSS* in the *myResourceGroup* with a system-assigned managed identity, as identified in the request body by the value `"identity":{"type":"SystemAssigned"}`. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01' -X PUT -d '{"sku":
{"tier":"Standard","capacity":3,"name":"Standard_D1_v2"},"location":"eastus","identity":
{"type":"SystemAssigned"},"properties":{"overprovision":true,"virtualMachineProfile":{"storageProfile":
{"imageReference":{"sku":"2016-
Datacenter","publisher":"MicrosoftWindowsServer","version":"latest","offer":"WindowsServer"},"osDisk":
{"caching":"ReadWrite","managedDisk":
{"storageAccountType":"Standard_LRS"},"createOption":"FromImage"}},"osProfile":
{"computerNamePrefix":"myVMSS","adminUsername":"azureuser","adminPassword":"myPassword12"},"networkProfi
le":{"networkInterfaceConfigurations":[{"name":"myVMSS","properties":
{"primary":true,"enableIPForwarding":true,"ipConfigurations":[{"name":"myVMSS","properties":{"subnet":
{"id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/virtualNetworks/myVnet/subnets/mySubnet"}
}}]}}]}},"upgradePolicy":{"mode":"Manual"}}}' -H "Content-Type: application/json" -H "Authorization:
Bearer <ACCESS TOKEN>"
```

```
PUT https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "sku":{
      "tier":"Standard",
      "capacity":3,
      "name":"Standard_D1_v2"
   },
   "location":"eastus",
   "identity":{
      "type":"SystemAssigned"
   },
   "properties":{
      "overprovision":true,
      "virtualMachineProfile":{
         "storageProfile":{
            "imageReference":{
               "sku":"2016-Datacenter",
               "publisher":"MicrosoftWindowsServer",
               "version":"latest",
               "offer":"WindowsServer"
            },
            "osDisk":{
               "caching":"ReadWrite",
               "managedDisk":{
                  "storageAccountType":"Standard_LRS"
               },
               "createOption":"FromImage"
            }
         },
         "osProfile":{
            "computerNamePrefix":"myVMSS",
            "adminUsername":"azureuser",
            "adminPassword":"myPassword12"
         },
         "networkProfile":{
            "networkInterfaceConfigurations":[
               {
                  "name":"myVMSS",
                  "properties":{
                     "primary":true,
                     "enableIPForwarding":true,
                     "ipConfigurations":[
                        {
                           "name":"myVMSS",
                           "properties":{
                              "subnet":{
                                 "id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/virtualNetworks/myVnet/subnets/mySubnet"
                              }
                           }
                        }
                     ]
                  }
               }
            ]
         }
      },
      "upgradePolicy":{
         "mode":"Manual"
      }
   }
}
```

Enable system-assigned managed identity on an existing virtual machine scale set

To enable system-assigned managed identity on an existing virtual machine scale set, you need to acquire an access token and then use CURL to call the Resource Manager REST endpoint to update the identity type.

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your virtual machine scale set with a system-assigned managed identity.

```
az account get-access-token
```

2. Use the following CURL command to call the Azure Resource Manager REST endpoint to enable system-assigned managed identity on your virtual machine scale set as identified in the request body by the value `{"identity":{"type":"SystemAssigned"}` for a virtual machine scale set named *myVMSS*. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

IMPORTANT

To ensure you don't delete any existing user-assigned managed identities that are assigned to the virtual machine scale set, you need to list the user-assigned managed identities by using this CURL command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE GROUP>/providers/Microsoft.Compute/virtualMachineScaleSets/<VMSS NAME>?api-version=2018-06-01' -H "Authorization: Bearer <ACCESS TOKEN>"
```

. If you have any user-assigned managed identities assigned to the virtual machine scale set as identified in the `identity` value in the response, skip to step 3 that shows you how to retain user-assigned managed identities while enabling system-assigned managed identity on your virtual machine scale set.

```
 curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01' -X PATCH -d '{"identity":{"type":"SystemAssigned"}}' -H "Content-Type:
application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned"
    }
 }
```

3. To enable system-assigned managed identity on a virtual machine scale set with existing user-assigned managed identities, you need to add `SystemAssigned` to the `type` value.

   For example, if your virtual machine scale set has the user-assigned managed identities `ID1` and `ID2` assigned to it, and you would like to add system-assigned managed identity to the virtual machine scale set, use the following CURL call. Replace `<ACCESS TOKEN>` and `<SUBSCRIPTION ID>` with values appropriate to your environment.

API version `2018-06-01` stores user-assigned managed identities in the `userAssignedIdentities` value in a dictionary format as opposed to the `identityIds` value in an array format used in API version `2017-12-01`.

API VERSION 2018-06-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01' -X PATCH -d '{"identity":{"type":"SystemAssigned,UserAssigned",
"userAssignedIdentities":{"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":
{},"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":
{}}}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned,UserAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":{
          },
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":{

          }
       }
    }
 }
```

API VERSION 2017-12-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01' -X PATCH -d '{"identity":{"type":"SystemAssigned,UserAssigned", "identityIds":
["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1","/sub
scriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2"]}}' -
H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "identity":{
      "type":"SystemAssigned,UserAssigned",
      "identityIds":[
         "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1",
         "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2"
      ]
   }
}
```

Disable system-assigned managed identity from a virtual machine scale set

To disable a system-assigned identity on an existing virtual machine scale set, you need to acquire an access token and then use CURL to call the Resource Manager REST endpoint to update the identity type to `None`.

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your virtual machine scale set with a system-assigned managed identity.

```
az account get-access-token
```

2. Update the virtual machine scale set using CURL to call the Azure Resource Manager REST endpoint to disable system-assigned managed identity. The following example disables system-assigned managed identity as identified in the request body by the value `{"identity":{"type":"None"}}` from a virtual machine scale set named *myVMSS*. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

IMPORTANT

To ensure you don't delete any existing user-assigned managed identities that are assigned to the virtual machine scale set, you need to list the user-assigned managed identities by using this CURL command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Compute/virtualMachineScaleSets/<VMSS NAME>?api-version=2018-06-01' -H
"Authorization: Bearer <ACCESS TOKEN>"
```

. If you have any user-assigned managed identity assigned to the virtual machine scale set, skip to step 3 that shows you how retain the user-assigned managed identities while removing the system-assigned managed identity from your virtual machine scale set.

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01' -X PATCH -d '{"identity":{"type":"None"}}' -H "Content-Type: application/json" -H
Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "identity":{
      "type":"None"
   }
}
```

To remove system-assigned managed identity from a virtual machine scale set that has user-assigned managed identities, remove `SystemAssigned` from the `{"identity":{"type:" "}}` value while keeping the `UserAssigned` value and the `userAssignedIdentities` dictionary values if you are using **API version 2018-06-01**. If you are using **API version 2017-12-01** or earlier, keep the `identityIds` array.

User-assigned managed identity

In this section, you learn how to add and remove user-assigned managed identity on a virtual machine scale set using CURL to make calls to the Azure Resource Manager REST endpoint.

Assign a user-assigned managed identity during the creation of a virtual machine scale set

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your virtual machine scale set with a system-assigned managed identity.

```
az account get-access-token
```

2. Create a network interface for your virtual machine scale set:

```
 az network nic create -g myResourceGroup --vnet-name myVnet --subnet mySubnet -n myNic
```

3. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your virtual machine scale set with a system-assigned managed identity.

```
az account get-access-token
```

4. Create a user-assigned managed identity using the instructions found here: Create a user-assigned

managed identity.

5. Create a virtual machine scale set using CURL to call the Azure Resource Manager REST endpoint. The following example creates a virtual machine scale set named *myVMSS* in the resource group *myResourceGroup* with a user-assigned managed identity `ID1`, as identified in the request body by the value `"identity":{"type":"UserAssigned"}`. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

API VERSION 2018-06-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01' -X PUT -d '{"sku":
{"tier":"Standard","capacity":3,"name":"Standard_D1_v2"},"location":"eastus","identity":
{"type":"UserAssigned","userAssignedIdentities":{"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":
{}}},"properties":{"overprovision":true,"virtualMachineProfile":{"storageProfile":{"imageReference":
{"sku":"2016-
Datacenter","publisher":"MicrosoftWindowsServer","version":"latest","offer":"WindowsServer"},"osDisk":
{"caching":"ReadWrite","managedDisk":
{"storageAccountType":"Standard_LRS"},"createOption":"FromImage"}},"osProfile":
{"computerNamePrefix":"myVMSS","adminUsername":"azureuser","adminPassword":"myPassword12"},"networkProfi
le":{"networkInterfaceConfigurations":[{"name":"myVMSS","properties":
{"primary":true,"enableIPForwarding":true,"ipConfigurations":[{"name":"myVMSS","properties":{"subnet":
{"id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/virtualNetworks/myVnet/subnets/mySubnet"}
}}]}}]}},"upgradePolicy":{"mode":"Manual"}}}' -H "Content-Type: application/json" -H "Authorization:
Bearer <ACCESS TOKEN>"
```

```
PUT https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "sku":{
       "tier":"Standard",
       "capacity":3,
       "name":"Standard_D1_v2"
    },
    "location":"eastus",
    "identity":{
       "type":"UserAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":{

          }
       }
    },
    "properties":{
       "overprovision":true,
       "virtualMachineProfile":{
          "storageProfile":{
             "imageReference":{
                "sku":"2016-Datacenter",
                "publisher":"MicrosoftWindowsServer",
                "version":"latest",
                "offer":"WindowsServer"
             },
             "osDisk":{
                "caching":"ReadWrite",
                "managedDisk":{
                   "storageAccountType":"Standard_LRS"
                },
                "createOption":"FromImage"
             }
          },
          "osProfile":{
             "computerNamePrefix":"myVMSS",
             "adminUsername":"azureuser",
             "adminPassword":"myPassword12"
          },
          "networkProfile":{
             "networkInterfaceConfigurations":[
                {
                   "name":"myVMSS",
                   "properties":{
                      "primary":true,
                      "enableIPForwarding":true,
                      "ipConfigurations":[
                         {
                            "name":"myVMSS",
                            "properties":{
                               "subnet":{
                                  "id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/virtualNetworks/myVnet/subnets/mySubnet"
                               }
                            }
                         }
                      ]
                   }
                }
             ]
          }
       },
       "upgradePolicy":{
          "mode":"Manual"
       }
    }
 }
```

API VERSION 2017-12-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01' -X PUT -d '{"sku":
{"tier":"Standard","capacity":3,"name":"Standard_D1_v2"},"location":"eastus","identity":
{"type":"UserAssigned","identityIds":["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"]},"pr
operties":{"overprovision":true,"virtualMachineProfile":{"storageProfile":{"imageReference":
{"sku":"2016-
Datacenter","publisher":"MicrosoftWindowsServer","version":"latest","offer":"WindowsServer"},"osDisk":
{"caching":"ReadWrite","managedDisk":
{"storageAccountType":"Standard_LRS"},"createOption":"FromImage"}},"osProfile":
{"computerNamePrefix":"myVMSS","adminUsername":"azureuser","adminPassword":"myPassword12"},"networkProfi
le":{"networkInterfaceConfigurations":[{"name":"myVMSS","properties":
{"primary":true,"enableIPForwarding":true,"ipConfigurations":[{"name":"myVMSS","properties":{"subnet":
{"id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/virtualNetworks/myVnet/subnets/mySubnet"}
}}]}}]}},"upgradePolicy":{"mode":"Manual"}}}' -H "Content-Type: application/json" -H "Authorization:
Bearer <ACCESS TOKEN>"
```

```
PUT https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "sku":{
      "tier":"Standard",
      "capacity":3,
      "name":"Standard_D1_v2"
   },
   "location":"eastus",
   "identity":{
      "type":"UserAssigned",
      "identityIds":[
         "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"
      ]
   },
   "properties":{
      "overprovision":true,
      "virtualMachineProfile":{
         "storageProfile":{
            "imageReference":{
               "sku":"2016-Datacenter",
               "publisher":"MicrosoftWindowsServer",
               "version":"latest",
               "offer":"WindowsServer"
            },
            "osDisk":{
               "caching":"ReadWrite",
               "managedDisk":{
                  "storageAccountType":"Standard_LRS"
               },
               "createOption":"FromImage"
            }
         },
         "osProfile":{
            "computerNamePrefix":"myVMSS",
            "adminUsername":"azureuser",
            "adminPassword":"myPassword12"
         },
         "networkProfile":{
            "networkInterfaceConfigurations":[
               {
                  "name":"myVMSS",
                  "properties":{
                     "primary":true,
                     "enableIPForwarding":true,
                     "ipConfigurations":[
                        {
                           "name":"myVMSS",
                           "properties":{
                              "subnet":{
                                 "id":"/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Network/virtualNetworks/myVnet/subnets/mySubnet"
                              }
                           }
                        }
                     ]
                  }
               }
            ]
         }
      },
      "upgradePolicy":{
         "mode":"Manual"
      }
   }
}
```

Assign a user-assigned managed identity to an existing Azure virtual machine scale set

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your virtual machine scale set with a system-assigned managed identity.

   ```
   az account get-access-token
   ```

2. Create a user-assigned managed identity using the instructions found here, Create a user-assigned managed identity.

3. To ensure you don't delete existing user or system-assigned managed identities that are assigned to the virtual machine scale set, you need to list the identity types assigned to the virtual machine scale set by using the following CURL command. If you have managed identities assigned to the virtual machine scale set, they are listed in the `identity` value.

   ```
   curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
   GROUP>/providers/Microsoft.Compute/virtualMachineScaleSets/<VMSS NAME>?api-version=2018-06-01' -H
   "Authorization: Bearer <ACCESS TOKEN>"
   ```

   ```
   GET https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
   GROUP>/providers/Microsoft.Compute/virtualMachineScaleSets/<VMSS NAME>?api-version=2018-06-01 HTTP/1.1
   ```

   Request headers

REQUEST HEADER	DESCRIPTION
Authorization	Required. Set to a valid `Bearer` access token.

4. If you don't have any user or system-assigned managed identities assigned to your virtual machine scale set, use the following CURL command to call the Azure Resource Manager REST endpoint to assign the first user-assigned managed identity to the virtual machine scale set. If you have a user or system-assigned managed identity(s) assigned to the virtual machine scale set, skip to step 5 that shows you how to add multiple user-assigned managed identities to a virtual machine scale set while also maintaining the system-assigned managed identity.

   The following example assigns a user-assigned managed identity, `ID1` to a virtual machine scale set named *myVMSS* in the resource group *myResourceGroup*. Replace `<ACCESS TOKEN>` with the value you received in the previous step when you requested a Bearer access token and the `<SUBSCRIPTION ID>` value as appropriate for your environment.

   API VERSION 2018-06-01

   ```
   curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
   ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
   version=2018-12-01' -X PATCH -d '{"identity":{"type":"userAssigned", "userAssignedIdentities":
   {"/subscriptions/<SUBSCRIPTION
   ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":
   {}}}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
   ```

   ```
   PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
   ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
   version=2018-12-01 HTTP/1.1
   ```

   Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"userAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":{

          }
       }
    }
 }
```

API VERSION 2017-12-01

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01' -X PATCH -d '{"identity":{"type":"userAssigned", "identityIds":
["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"]}}' -
H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"userAssigned",
       "identityIds":[
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"
       ]
    }
 }
```

5. If you have an existing user-assigned or system-assigned managed identity assigned to your virtual machine scale set:

API VERSION 2018-06-01

Add the user-assigned managed identity to the `userAssignedIdentities` dictionary value.

For example, if you have system-assigned managed identity and the user-assigned managed identity `ID1` currently assigned to your virtual machine scale and would like to add the user-assigned managed identity `ID2` to it:

```
curl  'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01' -X PATCH -d '{"identity":{"type":"SystemAssigned, UserAssigned",
"userAssignedIdentities":{"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":
{},"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":
{}}}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned, UserAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1":{

          },
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":{

          }
       }
    }
 }
```

API VERSION 2017-12-01

Retain the user-assigned managed identities you would like to keep in the `identityIds` array value while adding the new user-assigned managed identity.

For example, if you have system-assigned identity and the user-assigned managed identity `ID1` currently assigned to your virtual machine scale set and would like to add the user-assigned managed identity `ID2` to it:

```
curl  'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01' -X PATCH -d '{"identity":{"type":"SystemAssigned, UserAssigned", "identityIds":
["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1","/sub
scriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2"]}}' -
H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned, UserAssigned",
       "identityIds":[
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1",
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2"
       ]
    }
 }
```

Remove a user-assigned managed identity from a virtual machine scale set

1. Retrieve a Bearer access token, which you will use in the next step in the Authorization header to create your virtual machine scale set with a system-assigned managed identity.

```
az account get-access-token
```

2. To ensure you don't delete any existing user-assigned managed identities that you would like to keep assigned to the virtual machine scale set or remove the system-assigned managed identity, you need to list the managed identities by using the following CURL command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Compute/virtualMachineScaleSets/<VMSS NAME>?api-version=2018-06-01' -H
"Authorization: Bearer <ACCESS TOKEN>"
```

```
GET https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.Compute/virtualMachineScaleSets/<VMSS NAME>?api-version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Authorization	Required. Set to a valid `Bearer` access token.

If you have managed identities assigned to the VM, they are listed in the response in the `identity` value.

For example, if you have user-assigned managed identities `ID1` and `ID2` assigned to your virtual machine scale set, and you only want to keep `ID1` assigned and retain the system-assigned managed identity:

API VERSION 2018-06-01

Add `null` to the user-assigned managed identity you would like to remove:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01' -X PATCH -d '{"identity":{"type":"SystemAssigned, UserAssigned",
"userAssignedIdentities":{"/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":null}
}}' -H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2018-06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned, UserAssigned",
       "userAssignedIdentities":{
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID2":null
       }
    }
 }
```

API VERSION 2017-12-01

Retain only the user-assigned managed identity(s) you would like to keep in the `identityIds` array:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01' -X PATCH -d '{"identity":{"type":"SystemAssigned,UserAssigned", "identityIds":
["/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"]}}' -
H "Content-Type: application/json" -H Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-
version=2017-12-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
 {
    "identity":{
       "type":"SystemAssigned,UserAssigned",
       "identityIds":[
          "/subscriptions/<SUBSCRIPTION
ID>/resourcegroups/myResourceGroup/providers/Microsoft.ManagedIdentity/userAssignedIdentities/ID1"
       ]
    }
 }
```

If your virtual machine scale set has both system-assigned and user-assigned managed identities, you can remove all the user-assigned managed identities by switching to use only system-assigned using the following command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-version=2018-
06-01' -X PATCH -d '{"identity":{"type":"SystemAssigned"}}' -H "Content-Type: application/json" -H
Authorization:"Bearer <ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-version=2018-
06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "identity":{
      "type":"SystemAssigned"
   }
}
```

If your virtual machine scale set has only user-assigned managed identities and you would like to remove them all, use the following command:

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-version=2018-
06-01' -X PATCH -d '{"identity":{"type":"None"}}' -H "Content-Type: application/json" -H Authorization:"Bearer
<ACCESS TOKEN>"
```

```
PATCH https://management.azure.com/subscriptions/<SUBSCRIPTION
ID>/resourceGroups/myResourceGroup/providers/Microsoft.Compute/virtualMachineScaleSets/myVMSS?api-version=2018-
06-01 HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

```
{
   "identity":{
      "type":"None"
   }
}
```

Next steps

For information on how to create, list, or delete user-assigned managed identities using REST see:

- Create, list or delete a user-assigned managed identity using REST API calls

How to use managed identities for Azure resources on an Azure VM to acquire an access token

4/5/2019 • 14 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

This article provides various code and script examples for token acquisition, as well as guidance on important topics such as handling token expiration and HTTP errors.

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.

If you plan to use the Azure PowerShell examples in this article, be sure to install the latest version of Azure PowerShell.

IMPORTANT

- All sample code/script in this article assumes the client is running on a virtual machine with managed identities for Azure resources. Use the virtual machine "Connect" feature in the Azure portal, to remotely connect to your VM. For details on enabling managed identities for Azure resources on a VM, see Configure managed identities for Azure resources on a VM using the Azure portal, or one of the variant articles (using PowerShell, CLI, a template, or an Azure SDK).

IMPORTANT

- The security boundary of managed identities for Azure resources, is the resource it's being used on. All code/scripts running on a virtual machine can request and retrieve tokens for any managed identities available on it.

Overview

A client application can request managed identities for Azure resources app-only access token for accessing a given resource. The token is based on the managed identities for Azure resources service principal. As such, there is no need for the client to register itself to obtain an access token under its own service principal. The token is suitable for use as a bearer token in service-to-service calls requiring client credentials.

Get a token using HTTP	Protocol details for managed identities for Azure resources token endpoint

Get a token using the Microsoft.Azure.Services.AppAuthentication library for .NET	Example of using the Microsoft.Azure.Services.AppAuthentication library from a .NET client
Get a token using C#	Example of using managed identities for Azure resources REST endpoint from a C# client
Get a token using Java	Example of using managed identities for Azure resources REST endpoint from a Java client
Get a token using Go	Example of using managed identities for Azure resources REST endpoint from a Go client
Get a token using Azure PowerShell	Example of using managed identities for Azure resources REST endpoint from a PowerShell client
Get a token using CURL	Example of using managed identities for Azure resources REST endpoint from a Bash/CURL client
Handling token caching	Guidance for handling expired access tokens
Error handling	Guidance for handling HTTP errors returned from the managed identities for Azure resources token endpoint
Resource IDs for Azure services	Where to get resource IDs for supported Azure services

Get a token using HTTP

The fundamental interface for acquiring an access token is based on REST, making it accessible to any client application running on the VM that can make HTTP REST calls. This is similar to the Azure AD programming model, except the client uses an endpoint on the virtual machine (vs an Azure AD endpoint).

Sample request using the Azure Instance Metadata Service (IMDS) endpoint *(recommended)*:

```
GET 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
01&resource=https://management.azure.com/' HTTP/1.1 Metadata: true
```

ELEMENT	DESCRIPTION
`GET`	The HTTP verb, indicating you want to retrieve data from the endpoint. In this case, an OAuth access token.
`http://169.254.169.254/metadata/identity/oauth2/token`	The managed identities for Azure resources endpoint for the Instance Metadata Service.
`api-version`	A query string parameter, indicating the API version for the IMDS endpoint. Please use API version `2018-02-01` or greater.
`resource`	A query string parameter, indicating the App ID URI of the target resource. It also appears in the `aud` (audience) claim of the issued token. This example requests a token to access Azure Resource Manager, which has an App ID URI of https://management.azure.com/.

ELEMENT	DESCRIPTION
`Metadata`	An HTTP request header field, required by managed identities for Azure resources as a mitigation against Server Side Request Forgery (SSRF) attack. This value must be set to "true", in all lower case.
`object_id`	(Optional) A query string parameter, indicating the object_id of the managed identity you would like the token for. Required, if your VM has multiple user-assigned managed identities.
`client_id`	(Optional) A query string parameter, indicating the client_id of the managed identity you would like the token for. Required, if your VM has multiple user-assigned managed identities.
`mi_res_id`	(Optional) A query string parameter, indicating the mi_res_id (Azure Resource ID) of the managed identity you would like the token for. Required, if your VM has multiple user-assigned managed identities.

Sample request using the managed identities for Azure resources VM Extension Endpoint *(planned for deprecation in January 2019)*:

```
GET http://localhost:50342/oauth2/token?resource=https%3A%2F%2Fmanagement.azure.com%2F HTTP/1.1
Metadata: true
```

ELEMENT	DESCRIPTION
`GET`	The HTTP verb, indicating you want to retrieve data from the endpoint. In this case, an OAuth access token.
`http://localhost:50342/oauth2/token`	The managed identities for Azure resources endpoint, where 50342 is the default port and is configurable.
`resource`	A query string parameter, indicating the App ID URI of the target resource. It also appears in the `aud` (audience) claim of the issued token. This example requests a token to access Azure Resource Manager, which has an App ID URI of https://management.azure.com/.
`Metadata`	An HTTP request header field, required by managed identities for Azure resources as a mitigation against Server Side Request Forgery (SSRF) attack. This value must be set to "true", in all lower case.
`object_id`	(Optional) A query string parameter, indicating the object_id of the managed identity you would like the token for. Required, if your VM has multiple user-assigned managed identities.
`client_id`	(Optional) A query string parameter, indicating the client_id of the managed identity you would like the token for. Required, if your VM has multiple user-assigned managed identities.

Sample response:

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "access_token": "eyJ0eXAi...",
  "refresh_token": "",
  "expires_in": "3599",
  "expires_on": "1506484173",
  "not_before": "1506480273",
  "resource": "https://management.azure.com/",
  "token_type": "Bearer"
}
```

ELEMENT	DESCRIPTION
`access_token`	The requested access token. When calling a secured REST API, the token is embedded in the `Authorization` request header field as a "bearer" token, allowing the API to authenticate the caller.
`refresh_token`	Not used by managed identities for Azure resources.
`expires_in`	The number of seconds the access token continues to be valid, before expiring, from time of issuance. Time of issuance can be found in the token's `iat` claim.
`expires_on`	The timespan when the access token expires. The date is represented as the number of seconds from "1970-01-01T0:0:0Z UTC" (corresponds to the token's `exp` claim).
`not_before`	The timespan when the access token takes effect, and can be accepted. The date is represented as the number of seconds from "1970-01-01T0:0:0Z UTC" (corresponds to the token's `nbf` claim).
`resource`	The resource the access token was requested for, which matches the `resource` query string parameter of the request.
`token_type`	The type of token, which is a "Bearer" access token, which means the resource can give access to the bearer of this token.

Get a token using the Microsoft.Azure.Services.AppAuthentication library for .NET

For .NET applications and functions, the simplest way to work with managed identities for Azure resources is through the Microsoft.Azure.Services.AppAuthentication package. This library will also allow you to test your code locally on your development machine, using your user account from Visual Studio, the Azure CLI, or Active Directory Integrated Authentication. For more on local development options with this library, see the Microsoft.Azure.Services.AppAuthentication reference. This section shows you how to get started with the library in your code.

1. Add references to the Microsoft.Azure.Services.AppAuthentication and Microsoft.Azure.KeyVault NuGet packages to your application.

2. Add the following code to your application:

```
using Microsoft.Azure.Services.AppAuthentication;
using Microsoft.Azure.KeyVault;
// ...
var azureServiceTokenProvider = new AzureServiceTokenProvider();
string accessToken = await azureServiceTokenProvider.GetAccessTokenAsync("https://management.azure.com/");
// OR
var kv = new KeyVaultClient(new
KeyVaultClient.AuthenticationCallback(azureServiceTokenProvider.KeyVaultTokenCallback));
```

To learn more about Microsoft.Azure.Services.AppAuthentication and the operations it exposes, see the Microsoft.Azure.Services.AppAuthentication reference and the App Service and KeyVault with managed identities for Azure resources .NET sample.

Get a token using C#

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Net;
using System.Web.Script.Serialization;

// Build request to acquire managed identities for Azure resources token
HttpWebRequest request = (HttpWebRequest)WebRequest.Create("http://169.254.169.254/metadata/identity/oauth2/token?
api-version=2018-02-01&resource=https://management.azure.com/");
request.Headers["Metadata"] = "true";
request.Method = "GET";

try
{
    // Call /token endpoint
    HttpWebResponse response = (HttpWebResponse)request.GetResponse();

    // Pipe response Stream to a StreamReader, and extract access token
    StreamReader streamResponse = new StreamReader(response.GetResponseStream());
    string stringResponse = streamResponse.ReadToEnd();
    JavaScriptSerializer j = new JavaScriptSerializer();
    Dictionary<string, string> list = (Dictionary<string, string>) j.Deserialize(stringResponse,
typeof(Dictionary<string, string>));
    string accessToken = list["access_token"];
}
catch (Exception e)
{
    string errorText = String.Format("{0} \n\n{1}", e.Message, e.InnerException != null ? e.InnerException.Message
: "Acquire token failed");
}
```

Get a token using Java

Use this JSON library to retrieve a token using Java.

```
import java.io.*;
import java.net.*;
import com.fasterxml.jackson.core.*;

class GetMSIToken {
    public static void main(String[] args) throws Exception {

        URL msiEndpoint = new URL("http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
01&resource=https://management.azure.com/");
        HttpURLConnection con = (HttpURLConnection) msiEndpoint.openConnection();
        con.setRequestMethod("GET");
        con.setRequestProperty("Metadata", "true");

        if (con.getResponseCode()!=200) {
            throw new Exception("Error calling managed identity token endpoint.");
        }

        InputStream responseStream = con.getInputStream();

        JsonFactory factory = new JsonFactory();
        JsonParser parser = factory.createParser(responseStream);

        while(!parser.isClosed()){
            JsonToken jsonToken = parser.nextToken();

            if(JsonToken.FIELD_NAME.equals(jsonToken)){
                String fieldName = parser.getCurrentName();
                jsonToken = parser.nextToken();

                if("access_token".equals(fieldName)){
                    String accesstoken = parser.getValueAsString();
                    System.out.println("Access Token: " + accesstoken.substring(0,5)+ "..." +
accesstoken.substring(accesstoken.length()-5));
                    return;
                }
            }
        }
    }
}
```

Get a token using Go

```
package main

import (
  "fmt"
  "io/ioutil"
  "net/http"
  "net/url"
  "encoding/json"
)

type responseJson struct {
  AccessToken string `json:"access_token"`
  RefreshToken string `json:"refresh_token"`
  ExpiresIn string `json:"expires_in"`
  ExpiresOn string `json:"expires_on"`
  NotBefore string `json:"not_before"`
  Resource string `json:"resource"`
  TokenType string `json:"token_type"`
}

func main() {

    // Create HTTP request for a managed services for Azure resources token to access Azure Resource Manager
    var msi_endpoint *url.URL
    msi_endpoint, err := url.Parse("http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01")
    if err != nil {
      fmt.Println("Error creating URL: ", err)
      return
    }
    msi_parameters := url.Values{}
    msi_parameters.Add("resource", "https://management.azure.com/")
    msi_endpoint.RawQuery = msi_parameters.Encode()
    req, err := http.NewRequest("GET", msi_endpoint.String(), nil)
    if err != nil {
      fmt.Println("Error creating HTTP request: ", err)
      return
    }
    req.Header.Add("Metadata", "true")

    // Call managed services for Azure resources token endpoint
    client := &http.Client{}
    resp, err := client.Do(req)
    if err != nil{
      fmt.Println("Error calling token endpoint: ", err)
      return
    }

    // Pull out response body
    responseBytes,err := ioutil.ReadAll(resp.Body)
    defer resp.Body.Close()
    if err != nil {
      fmt.Println("Error reading response body : ", err)
      return
    }

    // Unmarshall response body into struct
    var r responseJson
    err = json.Unmarshal(responseBytes, &r)
    if err != nil {
      fmt.Println("Error unmarshalling the response:", err)
      return
    }

    // Print HTTP response and marshalled response body elements to console
    fmt.Println("Response status:", resp.Status)
    fmt.Println("access_token: ", r.AccessToken)
    fmt.Println("refresh_token: ", r.RefreshToken)
    fmt.Println("expires_in: ", r.ExpiresIn)
    fmt.Println("expires_on: ", r.ExpiresOn)
    fmt.Println("not_before: ", r.NotBefore)
    fmt.Println("resource: ", r.Resource)
    fmt.Println("token_type: ", r.TokenType)
}
```

Get a token using Azure PowerShell

The following example demonstrates how to use the managed identities for Azure resources REST endpoint from a PowerShell client to:

1. Acquire an access token.
2. Use the access token to call an Azure Resource Manager REST API and get information about the VM. Be sure to substitute your subscription ID, resource group name, and virtual machine name for `<SUBSCRIPTION-ID>`, `<RESOURCE-GROUP>`, and `<VM-NAME>`, respectively.

```
Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-
01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -Headers @{Metadata="true"}
```

Example on how to parse the access token from the response:

```
# Get an access token for managed identities for Azure resources
$response = Invoke-WebRequest -Uri 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https%3A%2F%2Fmanagement.azure.com%2F' `
                              -Headers @{Metadata="true"}
$content =$response.Content | ConvertFrom-Json
$access_token = $content.access_token
echo "The managed identities for Azure resources access token is $access_token"

# Use the access token to get resource information for the VM
$vmInfoRest = (Invoke-WebRequest -Uri 'https://management.azure.com/subscriptions/<SUBSCRIPTION-ID>/resourceGroups/<RESOURCE-GROUP>/providers/Microsoft.Compute/virtualMachines/<VM-NAME>?api-version=2017-12-01' -Method GET -ContentType "application/json" -Headers @{ Authorization ="Bearer $access_token"}).content
echo "JSON returned from call to get VM info:"
echo $vmInfoRest
```

Get a token using CURL

```
curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -H Metadata:true -s
```

Example on how to parse the access token from the response:

```
response=$(curl 'http://169.254.169.254/metadata/identity/oauth2/token?api-version=2018-02-01&resource=https%3A%2F%2Fmanagement.azure.com%2F' -H Metadata:true -s)
access_token=$(echo $response | python -c 'import sys, json; print (json.load(sys.stdin)["access_token"])')
echo The managed identities for Azure resources access token is $access_token
```

Token caching

While the managed identities for Azure resources subsystem being used (IMDS/managed identities for Azure resources VM Extension) does cache tokens, we also recommend to implement token caching in your code. As a result, you should prepare for scenarios where the resource indicates that the token is expired.

On-the-wire calls to Azure AD result only when:

- cache miss occurs due to no token in the managed identities for Azure resources subsystem cache
- the cached token is expired

Error handling

The managed identities for Azure resources endpoint signals errors via the status code field of the HTTP response message header, as either 4xx or 5xx errors:

STATUS CODE	ERROR REASON	HOW TO HANDLE
404 Not found.	IMDS endpoint is updating.	Retry with Expontential Backoff. See guidance below.
429 Too many requests.	IMDS Throttle limit reached.	Retry with Exponential Backoff. See guidance below.
4xx Error in request.	One or more of the request parameters was incorrect.	Do not retry. Examine the error details for more information. 4xx errors are design-time errors.

STATUS CODE	ERROR REASON	HOW TO HANDLE
5xx Transient error from service.	The managed identities for Azure resources sub-system or Azure Active Directory returned a transient error.	It is safe to retry after waiting for at least 1 second. If you retry too quickly or too often, IMDS and/or Azure AD may return a rate limit error (429).
timeout	IMDS endpoint is updating.	Retry with Expontential Backoff. See guidance below.

If an error occurs, the corresponding HTTP response body contains JSON with the error details:

ELEMENT	DESCRIPTION
error	Error identifier.
error_description	Verbose description of error. **Error descriptions can change at any time. Do not write code that branches based on values in the error description.**

HTTP response reference

This section documents the possible error responses. A "200 OK" status is a successful response, and the access token is contained in the response body JSON, in the access_token element.

STATUS CODE	ERROR	ERROR DESCRIPTION	SOLUTION
400 Bad Request	invalid_resource	AADSTS50001: The application named <*URI*> was not found in the tenant named <*TENANT-ID*>. This can happen if the application has not been installed by the administrator of the tenant or consented to by any user in the tenant. You might have sent your authentication request to the wrong tenant.\	(Linux only)
400 Bad Request	bad_request_102	Required metadata header not specified	Either the `Metadata` request header field is missing from your request, or is formatted incorrectly. The value must be specified as `true`, in all lower case. See the "Sample request" in the preceding REST section for an example.
401 Unauthorized	unknown_source	Unknown Source <*URI*>	Verify that your HTTP GET request URI is formatted correctly. The `scheme:host/resource-path` portion must be specified as `http://localhost:50342/oauth2/tok` . See the "Sample request" in the preceding REST section for an example.

STATUS CODE	ERROR	ERROR DESCRIPTION	SOLUTION
	invalid_request	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.	
	unauthorized_client	The client is not authorized to request an access token using this method.	Caused by a request that didn't use local loopback to call the extension, or on a VM that doesn't have managed identities for Azure resources configured correctly. See Configure managed identities for Azure resources on a VM using the Azure portal if you need assistance with VM configuration.
	access_denied	The resource owner or authorization server denied the request.	
	unsupported_response_type	The authorization server does not support obtaining an access token using this method.	
	invalid_scope	The requested scope is invalid, unknown, or malformed.	
500 Internal server error	unknown	Failed to retrieve token from the Active directory. For details see logs in *<file path>*	Verify that managed identities for Azure resources has been enabled on the VM. See Configure managed identities for Azure resources on a VM using the Azure portal if you need assistance with VM configuration. Also verify that your HTTP GET request URI is formatted correctly, particularly the resource URI specified in the query string. See the "Sample request" in the preceding REST section for an example, or Azure services that support Azure AD authentication for a list of services and their respective resource IDs.

Retry guidance

It is recommended to retry if you receive a 404, 429, or 5xx error code (see Error handling above).

Throttling limits apply to the number of calls made to the IMDS endpoint. When the throttling threshold is exceeded, IMDS endpoint limits any further requests while the throttle is in effect. During this period, the IMDS endpoint will return the HTTP status code 429 ("Too many requests"), and the requests fail.

For retry, we recommend the following strategy:

RETRY STRATEGY	SETTINGS	VALUES	HOW IT WORKS
ExponentialBackoff	Retry count Min back-off Max back-off Delta back-off First fast retry	5 0 sec 60 sec 2 sec false	Attempt 1 - delay 0 sec Attempt 2 - delay ~2 sec Attempt 3 - delay ~6 sec Attempt 4 - delay ~14 sec Attempt 5 - delay ~30 sec

Resource IDs for Azure services

See Azure services that support Azure AD authentication for a list of resources that support Azure AD and have been tested with managed identities for Azure resources, and their respective resource IDs.

Next steps

- To enable managed identities for Azure resources on an Azure VM, see Configure managed identities for Azure resources on a VM using the Azure portal.

How to use managed identities for Azure resources on an Azure VM for sign in

3/26/2019 • 4 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This article provides PowerShell and CLI script examples for sign-in using managed identities for Azure resources service principal, and guidance on important topics such as error handling.

NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.

If you plan to use the Azure PowerShell or Azure CLI examples in this article, be sure to install the latest version of Azure PowerShell or Azure CLI.

IMPORTANT

- All sample script in this article assumes the command-line client is running on a VM with managed identities for Azure resources enabled. Use the VM "Connect" feature in the Azure portal, to remotely connect to your VM. For details on enabling managed identities for Azure resources on a VM, see Configure managed identities for Azure resources on a VM using the Azure portal, or one of the variant articles (using PowerShell, CLI, a template, or an Azure SDK).
- To prevent errors during resource access, the VM's managed identity must be given at least "Reader" access at the appropriate scope (the VM or higher) to allow Azure Resource Manager operations on the VM. See Assign managed identities for Azure resources access to a resource using the Azure portal for details.

Overview

Managed identities for Azure resources provides a service principal object , which is created upon enabling managed identities for Azure resources on the VM. The service principal can be given access to Azure resources, and used as an identity by script/command-line clients for sign in and resource access. Traditionally, in order to access secured resources under its own identity, a script client would need to:

- be registered and consented with Azure AD as a confidential/web client application
- sign in under its service principal, using the app's credentials (which are likely embedded in the script)

With managed identities for Azure resources, your script client no longer needs to do either, as it can sign in under the managed identities for Azure resources service principal."

Azure CLI

The following script demonstrates how to:

1. Sign in to Azure AD under the VM's managed identity for Azure resources service principal
2. Call Azure Resource Manager and get the VM's service principal ID. CLI takes care of managing token acquisition/use for you automatically. Be sure to substitute your virtual machine name for `<VM-NAME>`.

```
az login --identity

spID=$(az resource list -n <VM-NAME> --query [*].identity.principalId --out tsv)
echo The managed identity for Azure resources service principal ID is $spID
```

Azure PowerShell

The following script demonstrates how to:

1. Sign in to Azure AD under the VM's managed identity for Azure resources service principal
2. Call an Azure Resource Manager cmdlet to get information about the VM. PowerShell takes care of managing token use for you automatically.

```
Add-AzAccount -identity

# Call Azure Resource Manager to get the service principal ID for the VM's managed identity for Azure
resources.
$vmInfoPs = Get-AzVM -ResourceGroupName <RESOURCE-GROUP> -Name <VM-NAME>
$spID = $vmInfoPs.Identity.PrincipalId
echo "The managed identity for Azure resources service principal ID is $spID"
```

Resource IDs for Azure services

See Azure services that support Azure AD authentication for a list of resources that support Azure AD and have been tested with managed identities for Azure resources, and their respective resource IDs.

Error handling guidance

Responses such as the following may indicate that the VM's managed identity for Azure resources has not been correctly configured:

- PowerShell: *Invoke-WebRequest : Unable to connect to the remote server*
- CLI: *MSI: Failed to retrieve a token from* `http://localhost:50342/oauth2/token` *with an error of 'HTTPConnectionPool(host='localhost', port=50342)*

If you receive one of these errors, return to the Azure VM in the Azure portal and:

- Go to the **Identity** page and ensure **System assigned** is set to "Yes."
- Go to the **Extensions** page and ensure the managed identities for Azure resources extension **(planned for deprecation in January 2019)** deployed successfully.

If either is incorrect, you may need to redeploy the managed identities for Azure resources on your resource again, or troubleshoot the deployment failure. See Configure Managed identities for Azure resources on a VM using the

Azure portal if you need assistance with VM configuration.

Next steps

- To enable managed identities for Azure resources on an Azure VM, see Configure managed identities for Azure resources on an Azure VM using PowerShell, or Configure managed identities for Azure resources on an Azure VM using Azure CLI

How to use managed identities for Azure resources on an Azure VM with Azure SDKs

3/26/2019 • 2 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

This article provides a list of SDK samples, which demonstrate use of their respective Azure SDK's support for managed identities for Azure resources.

Prerequisites

- If you're not familiar with the managed identities for Azure resources feature, see this overview. If you don't have an Azure account, sign up for a free account before you continue.

IMPORTANT

- All sample code/script in this article assumes the client is running on a VM with managed identities for Azure resources enabled. Use the VM "Connect" feature in the Azure portal, to remotely connect to your VM. For details on enabling managed identities for Azure resources on a VM, see Configure managed identities for Azure resources on a VM using the Azure portal, or one of the variant articles (using PowerShell, CLI, a template, or an Azure SDK).

SDK code samples

SDK	CODE SAMPLE
.NET	Deploy an Azure Resource Manager template from a Windows VM using managed identities for Azure resources
.NET Core	Call Azure services from a Linux VM using managed identities for Azure resources
Node.js	Manage resources using managed identities for Azure resources
Python	Use managed identities for Azure resources to authenticate simply from inside a VM
Ruby	Manage resources from a VM with managed identities for Azure resources enabled

Next steps

- See Azure SDKs for the full list of Azure SDK resources, including library downloads, documentation, and more.
- To enable managed identities for Azure resources on an Azure VM, see Configure managed identities for Azure

resources on a VM using the Azure portal.

How to stop using the virtual machine managed identities extension and start using the Azure Instance Metadata Service

5/7/2019 • 7 minutes to read • Edit Online

Virtual machine extension for managed identities

The virtual machine extension for managed identities is used to request tokens for a managed identity within the virtual machine. The workflow consists of the following steps:

1. First, the workload within the resource calls the local endpoint `http://localhost/oauth2/token` to request an access token.
2. The virtual machine extension then uses the credentials for the managed identity, to request an access token from Azure AD..
3. The access token is returned to the caller, and can be used to authenticate to services that support Azure AD authentication, like Azure Key Vault or Azure Storage.

Due to several limitations outlined in the next section, the managed identity VM extension has been deprecated in favor of using the equivalent endpoint in the Azure Instance Metadata Service (IMDS)

Provision the extension

When you configure a virtual machine or virtual machine scale set to have a managed identity, you may optionally choose to provision the managed identities for Azure resources VM extension using the `-Type` parameter on the Set-AzVMExtension cmdlet. You can pass either `ManagedIdentityExtensionForWindows` or `ManagedIdentityExtensionForLinux`, depending on the type of virtual machine, and name it using the `-Name` parameter. The `-Settings` parameter specifies the port used by the OAuth token endpoint for token acquisition:

```
   $settings = @{ "port" = 50342 }
   Set-AzVMExtension -ResourceGroupName myResourceGroup -Location WestUS -VMName myVM -Name
"ManagedIdentityExtensionForWindows" -Type "ManagedIdentityExtensionForWindows" -Publisher
"Microsoft.ManagedIdentity" -TypeHandlerVersion "1.0" -Settings $settings
```

You can also use the Azure Resource Manager deployment template to provision the VM extension, by adding the following JSON to the `resources` section to the template (use `ManagedIdentityExtensionForLinux` for the name and type elements for the Linux version).

~~~
```json
{
    "type": "Microsoft.Compute/virtualMachines/extensions",
    "name": "[concat(variables('vmName'),'/ManagedIdentityExtensionForWindows')]",
    "apiVersion": "2018-06-01",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'))]"
    ],
    "properties": {
        "publisher": "Microsoft.ManagedIdentity",
        "type": "ManagedIdentityExtensionForWindows",
        "typeHandlerVersion": "1.0",
        "autoUpgradeMinorVersion": true,
        "settings": {
            "port": 50342
        }
    }
}
```
~~~

If you're working with virtual machine scale sets, you can also provision the managed identities for Azure resources virtual machine scale set extension using the Add-AzVmssExtension cmdlet. You can pass either `ManagedIdentityExtensionForWindows` or `ManagedIdentityExtensionForLinux`, depending on the type of virtual machine scale set, and name it using the `-Name` parameter. The `-Settings` parameter specifies the port used by the OAuth token endpoint for token acquisition:

```
$setting = @{ "port" = 50342 }
$vmss = Get-AzVmss
Add-AzVmssExtension -VirtualMachineScaleSet $vmss -Name "ManagedIdentityExtensionForWindows" -Type
"ManagedIdentityExtensionForWindows" -Publisher "Microsoft.ManagedIdentity" -TypeHandlerVersion "1.0" -
Setting $settings
```

To provision the virtual machine scale set extension with the Azure Resource Manager deployment template, add the following JSON to the `extensionpProfile` section to the template (use `ManagedIdentityExtensionForLinux` for the name and type elements for the Linux version).

~~~
```json
"extensionProfile": {
    "extensions": [
        {
            "name": "ManagedIdentityWindowsExtension",
            "properties": {
                "publisher": "Microsoft.ManagedIdentity",
                "type": "ManagedIdentityExtensionForWindows",
                "typeHandlerVersion": "1.0",
                "autoUpgradeMinorVersion": true,
                "settings": {
                    "port": 50342
                },
                "protectedSettings": {}
            }
        }
```
~~~

Provisioning of the virtual machine extension might fail due to DNS lookup failures. If this happens, restart the virtual machine, and try again.

Remove the extension

To remove the extension, use `-n ManagedIdentityExtensionForWindows` or `-n ManagedIdentityExtensionForLinux` switch (depending on the type of virtual machine) with az vm extension delete, or az vmss extension delete for virtual machine scale sets using Azure CLI, or `Remove-AzVMExtension` for Powershell:

```
az vm identity --resource-group myResourceGroup --vm-name myVm -n ManagedIdentityExtensionForWindows
```

```
az vmss extension delete -n ManagedIdentityExtensionForWindows -g myResourceGroup -vmss-name myVMSS
```

```
Remove-AzVMExtension -ResourceGroupName myResourceGroup -Name "ManagedIdentityExtensionForWindows" -VMName
myVM
```

Acquire a token using the virtual machine extension

The following is a sample request using the managed identities for Azure resources VM Extension Endpoint:

```
GET http://localhost:50342/oauth2/token?resource=https%3A%2F%2Fmanagement.azure.com%2F HTTP/1.1
Metadata: true
```

ELEMENT	DESCRIPTION
`GET`	The HTTP verb, indicating you want to retrieve data from the endpoint. In this case, an OAuth access token.
`http://localhost:50342/oauth2/token`	The managed identities for Azure resources endpoint, where 50342 is the default port and is configurable.
`resource`	A query string parameter, indicating the App ID URI of the target resource. It also appears in the `aud` (audience) claim of the issued token. This example requests a token to access Azure Resource Manager, which has an App ID URI of https://management.azure.com/.
`Metadata`	An HTTP request header field, required by managed identities for Azure resources as a mitigation against Server Side Request Forgery (SSRF) attack. This value must be set to "true", in all lower case.
`object_id`	(Optional) A query string parameter, indicating the object_id of the managed identity you would like the token for. Required, if your VM has multiple user-assigned managed identities.
`client_id`	(Optional) A query string parameter, indicating the client_id of the managed identity you would like the token for. Required, if your VM has multiple user-assigned managed identities.

Sample response:

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "access_token": "eyJ0eXAi...",
  "refresh_token": "",
  "expires_in": "3599",
  "expires_on": "1506484173",
  "not_before": "1506480273",
  "resource": "https://management.azure.com/",
  "token_type": "Bearer"
}
```

ELEMENT	DESCRIPTION
`access_token`	The requested access token. When calling a secured REST API, the token is embedded in the `Authorization` request header field as a "bearer" token, allowing the API to authenticate the caller.
`refresh_token`	Not used by managed identities for Azure resources.
`expires_in`	The number of seconds the access token continues to be valid, before expiring, from time of issuance. Time of issuance can be found in the token's `iat` claim.
`expires_on`	The timespan when the access token expires. The date is represented as the number of seconds from "1970-01-01T0:0:0Z UTC" (corresponds to the token's `exp` claim).
`not_before`	The timespan when the access token takes effect, and can be accepted. The date is represented as the number of seconds from "1970-01-01T0:0:0Z UTC" (corresponds to the token's `nbf` claim).
`resource`	The resource the access token was requested for, which matches the `resource` query string parameter of the request.
`token_type`	The type of token, which is a "Bearer" access token, which means the resource can give access to the bearer of this token.

Troubleshoot the virtual machine extension

Restart the virtual machine extension after a failure

On Windows and certain versions of Linux, if the extension stops, the following cmdlet may be used to manually restart it:

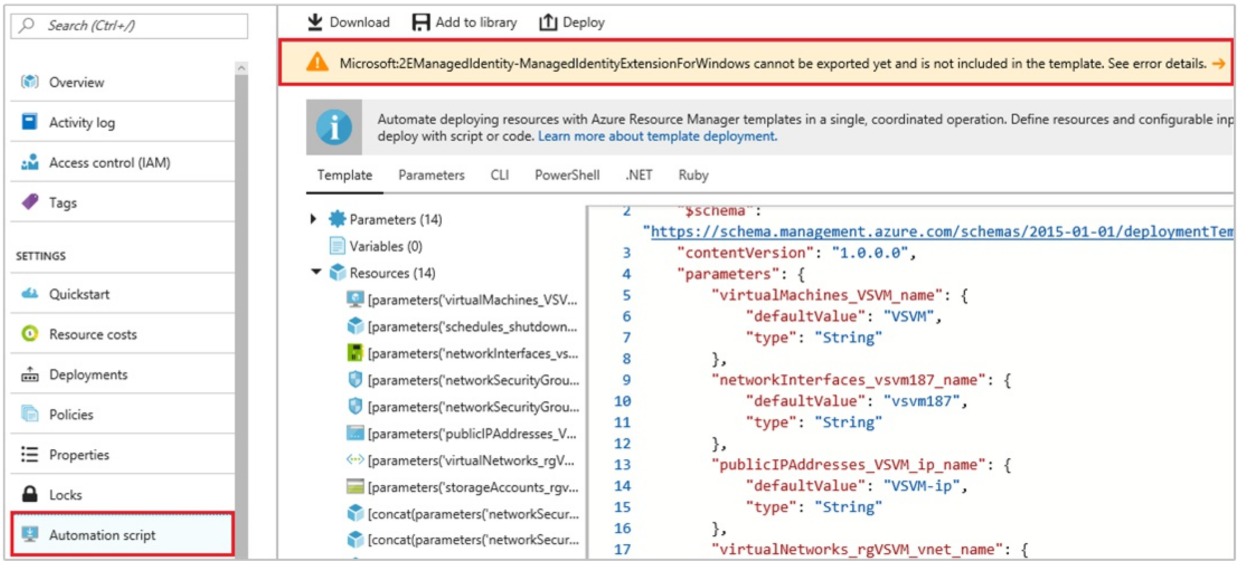
```
Set-AzVMExtension -Name <extension name>  -Type <extension Type>  -Location <location> -Publisher
Microsoft.ManagedIdentity -VMName <vm name> -ResourceGroupName <resource group name> -ForceRerun <Any
string different from any last value used>
```

Where:

- Extension name and type for Windows is: `ManagedIdentityExtensionForWindows`
- Extension name and type for Linux is: `ManagedIdentityExtensionForLinux`

"Automation script" fails when attempting schema export for managed identities for Azure resources extension

When managed identities for Azure resources is enabled on a virtual machine, the following error is shown when attempting to use the “Automation script” feature for the virtual machine, or its resource group:



Microsoft:2EManagedIdentity-ManagedIdentityExtensionForWindows cannot be exported yet and is not included in the template. See error details. →

Automate deploying resources with Azure Resource Manager templates in a single, coordinated operation. Define resources and configurable inp
deploy with script or code. Learn more about template deployment.

Parameters (14)
Variables (0)
Resources (14)
[parameters('virtualMachines_VSV...
[parameters('schedules_shutdown...
[parameters('networkInterfaces_vs...
[parameters('networkSecurityGrou...
[parameters('networkSecurityGrou...
[parameters('publicIPAddresses_V...
[parameters('virtualNetworks_rgV...
[parameters('storageAccounts_rgv...
[concat(parameters('networkSecur...
[concat(parameters('networkSecur...

```
    "$schema":
 "https://schema.management.azure.com/schemas/2015-01-01/deploymentTem
    "contentVersion": "1.0.0.0",
    "parameters": {
        "virtualMachines_VSVM_name": {
            "defaultValue": "VSVM",
            "type": "String"
        },
        "networkInterfaces_vsvm187_name": {
            "defaultValue": "vsvm187",
            "type": "String"
        },
        "publicIPAddresses_VSVM_ip_name": {
            "defaultValue": "VSVM-ip",
            "type": "String"
        },
        "virtualNetworks_rgVSVM_vnet_name": {
```

The managed identities for Azure resources virtual machine extension does not currently support the ability to export its schema to a resource group template. As a result, the generated template does not show configuration parameters to enable managed identities for Azure resources on the resource. These sections can be added manually by following the examples in Configure managed identities for Azure resources on an Azure virtual machine using a templates.

When the schema export functionality becomes available for the managed identities for Azure resources virtual machine extension (planned for deprecation in January 2019), it will be listed in Exporting Resource Groups that contain VM extensions.

Limitations of the virtual machine extension

There are several major limitations to using the virtual machine extension.

- The most serious limitation is the fact that the credentials used to request tokens are stored on the virtual machine. An attacker who successfully breaches the virtual machine can exfiltrate the credentials.
- Furthermore, the virtual machine extension is still unsupported by several Linux distributions, with a huge development cost to modify, build and test the extension on each of those distributions. Currently, only the following Linux distributions are supported:
    - CoreOS Stable
    - CentOS 7.1
    - Red Hat 7.2
    - Ubuntu 15.04
    - Ubuntu 16.04
- There is a performance impact to deploying virtual machines with managed identities, as the virtual machine extension also has to be provisioned.
- Finally, the virtual machine extension can only support having 32 user-assigned managed identities per virtual machine.

Azure Instance Metadata Service

The Azure Instance Metadata Service (IMDS) is a REST endpoint that provides information about running virtual machine instances that can be used to manage and configure your virtual machines. The endpoint is available at a well-known non-routable IP address (`169.254.169.254`) that can be accessed only from within the virtual machine.

There are several advantages to using Azure IMDS to request tokens.

1. The service is external to the virtual machine, therefore the credentials used by managed identities are no longer present on the virtual machine. Instead, they are hosted and secured on the host machine of the Azure virtual machine.
2. All Windows and Linux operating systems supported on Azure IaaS can use managed identities.
3. Deployment is faster and easier, since the VM extension no longer needs to be provisioned.
4. With the IMDS endpoint, up to 1000 user-assigned managed identities can be assigned to a single virtual machine.
5. There is no significant change to the requests using IMDS as opposed to those using the virtual machine extension, therefore it is fairly simple to port over existing deployments that currently use the virtual machine extension.

For these reasons, the Azure IMDS service will be the defacto way to request tokens, once the virtual machine extension is deprecated.

Next Steps

- How to use managed identities for Azure resources on an Azure virtual machine to acquire an access token
- Azure Instance Metadata Service

Assign a managed identity access to a resource by using the Azure portal

3/26/2019 • 2 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

After you've configured an Azure resource with a managed identity, you can give the managed identity access to another resource, just like any security principal. This article shows you how to give an Azure virtual machine or virtual machine scale set's managed identity access to an Azure storage account, by using the Azure portal.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.

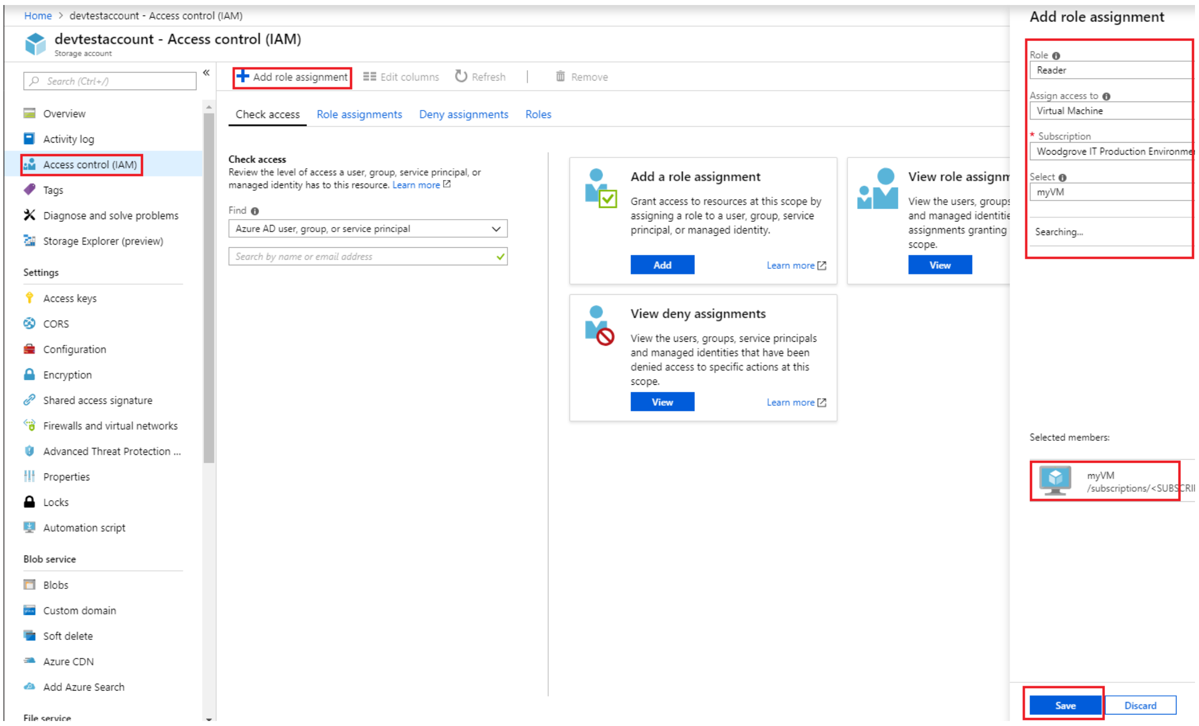
Use RBAC to assign a managed identity access to another resource

After you've enabled managed identity on an Azure resource, such as an Azure VM or Azure VMSS:

1. Sign in to the Azure portal using an account associated with the Azure subscription under which you have configured the managed identity.

2. Navigate to the desired resource on which you want to modify access control. In this example, we are giving an Azure virtual machine access to a storage account, so we navigate to the storage account.

3. Select the **Access control (IAM)** page of the resource, and select **+ Add role assignment**. Then specify the **Role**, **Assign access to**, and specify the corresponding **Subscription**. Under the search criteria area, you should see the resource. Select the resource, and select **Save**.


Home > devtestaccount - Access control (IAM)

devtestaccount - Access control (IAM)
Storage account

Search (Ctrl+/)

+ Add role assignment | Edit columns | Refresh | Remove

Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Storage Explorer (preview)

Settings
Access keys
CORS
Configuration
Encryption
Shared access signature
Firewalls and virtual networks
Advanced Threat Protection ...
Properties
Locks
Automation script

Blob service
Blobs
Custom domain
Soft delete
Azure CDN
Add Azure Search

File service

Check access | Role assignments | Deny assignments | Roles

Check access
Review the level of access a user, group, service principal, or managed identity has to this resource. Learn more

Find
Azure AD user, group, or service principal
Search by name or email address

Add a role assignment
Grant access to resources at this scope by assigning a role to a user, group, service principal, or managed identity.
Add | Learn more

View role assignm
View the users, groups and managed identitie assignments granting scope.
View

View deny assignments
View the users, groups, service principals and managed identities that have been denied access to specific actions at this scope.
View | Learn more

Add role assignment

Role
Reader

Assign access to
Virtual Machine

* Subscription
Woodgrove IT Production Environme

Select
myVM

Searching...

Selected members:
myVM
/subscriptions/<SUBSCRII

Save | Discard

Next steps

- Managed identity for Azure resources overview
- To enable managed identity on an Azure virtual machine, see Configure managed identities for Azure resources on a VM using the Azure portal.
- To enable managed identity on an Azure virtual machine scale set, see Configure managed identities for Azure resources on a virtual machine scale set using the Azure portal.

Assign a managed identity access to a resource using Azure CLI

3/26/2019 • 2 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

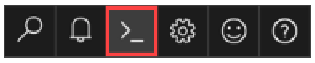
Once you've configured an Azure resource with a managed identity, you can give the managed identity access to another resource, just like any security principal. This example shows you how to give an Azure virtual machine or virtual machine scale set's managed identity access to an Azure storage account using Azure CLI.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- To run the CLI script examples, you have three options:
  - Use Azure Cloud Shell from the Azure portal (see next section).
  - Use the embedded Azure Cloud Shell via the "Try It" button, located in the top right corner of each code block.
  - Install the latest version of Azure CLI if you prefer to use a local CLI console.

Open Azure Cloud Shell

Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Select **Copy** to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.	
Open Cloud Shell in your browser.	
Select the **Cloud Shell** button on the menu in the upper-right corner of the Azure portal.	

Use RBAC to assign a managed identity access to another resource

After you've enabled managed identity on an Azure resource, such as an Azure virtual machine or Azure virtual machine scale set:

1. If you're using the Azure CLI in a local console, first sign in to Azure using az login. Use an account that is

associated with the Azure subscription under which you would like to deploy the VM or virtual machine scale set:

```
az login
```

2. In this example, we are giving an Azure virtual machine access to a storage account. First we use az resource list to get the service principal for the virtual machine named myVM:

   ```
   spID=$(az resource list -n myVM --query [*].identity.principalId --out tsv)
   ```

   For an Azure virtual machine scale set, the command is the same except here, you get the service principal for the virtual machine scale set named "DevTestVMSS":

   ```
   spID=$(az resource list -n DevTestVMSS --query [*].identity.principalId --out tsv)
   ```

3. Once you have the service principal ID, use az role assignment create to give the virtual machine or virtual machine scale set "Reader" access to a storage account called "myStorageAcct":

   ```
   az role assignment create --assignee $spID --role 'Reader' --scope
   /subscriptions/<mySubscriptionID>/resourceGroups/<myResourceGroup>/providers/Microsoft.Storage/storageAc
   counts/myStorageAcct
   ```

Next steps

- Managed identities for Azure resources overview
- To enable managed identity on an Azure virtual machine, see Configure managed identities for Azure resources on an Azure VM using Azure CLI.
- To enable managed identity on an Azure virtual machine scale set, see Configure managed identities for Azure resources on a virtual machine scale set using Azure CLI.

Assign a managed identity access to a resource using PowerShell

3/26/2019 • 2 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Once you've configured an Azure resource with a managed identity, you can give the managed identity access to another resource, just like any security principal. This example shows you how to give an Azure virtual machine's managed identity access to an Azure storage account using PowerShell.

NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- Install the latest version of Azure PowerShell if you haven't already.

Use RBAC to assign a managed identity access to another resource

After you've enabled managed identity on an Azure resource, such as an Azure VM:

1. Sign in to Azure using the `Connect-AzAccount` cmdlet. Use an account that is associated with the Azure subscription under which you have configured the managed identity:

   ```
   Connect-AzAccount
   ```

2. In this example, we are giving an Azure VM access to a storage account. First we use Get-AzVM to get the service principal for the VM named `myVM`, which was created when we enabled managed identity. Then, use New-AzRoleAssignment to give the VM **Reader** access to a storage account called `myStorageAcct`:

   ```
   $spID = (Get-AzVM -ResourceGroupName myRG -Name myVM).identity.principalid
   New-AzRoleAssignment -ObjectId $spID -RoleDefinitionName "Reader" -Scope
   "/subscriptions/<mySubscriptionID>/resourceGroups/<myResourceGroup>/providers/Microsoft.Storage/storageA
   ccounts/<myStorageAcct>"
   ```

Next steps

- Managed identity for Azure resources overview
- To enable managed identity on an Azure VM, see Configure managed identities for Azure resources on an Azure VM using PowerShell.

Create, list, delete or assign a role to a user-assigned managed identity using the Azure portal

3/26/2019 • 2 minutes to read • Edit Online

User assigned managed identities are a preview feature of Azure Active Directory. Make sure you review the known issues before you begin. For more information about previews, see Supplemental Terms of Use for Microsoft Azure Previews.

Managed identities for Azure resources provides Azure services with a managed identity in Azure Active Directory. You can use this identity to authenticate to services that support Azure AD authentication, without needing credentials in your code.

In this article, you learn how to create, list, delete or assign a role to a user-assigned managed identity using the Azure Portal.

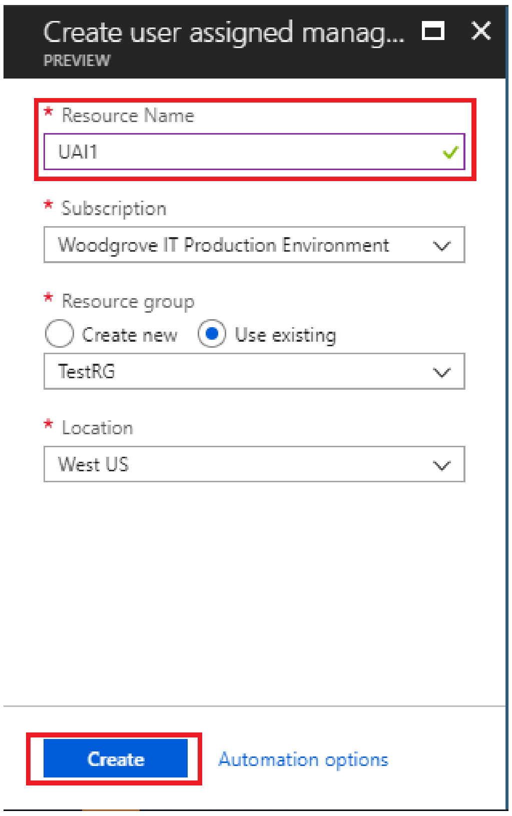
Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.

Create a user-assigned managed identity

To create a user-assigned managed identity, your account needs the Managed Identity Contributor role assignment.

1. Sign in to the Azure portal using an account associated with the Azure subscription to create the user-assigned managed identity.
2. In the search box, type *Managed Identities*, and under **Services**, click **Managed Identities**.
3. Click **Add** and enter values in the following fields under **Create user assigned managed** identity pane:
   - **Resource Name**: This is the name for your user-assigned managed identity, for example UAI1.
   - **Subscription**: Choose the subscription to create the user-assigned managed identity under
   - **Resource Group**: Create a new resource group to contain your user-assigned managed identity or choose **Use existing** to create the user-assigned managed identity in an existing resource group.
   - **Location**: Choose a location to deploy the user-assigned managed identity,for example **West US**.
4. Click **Create**.


Create user assigned manag...
PREVIEW

* Resource Name
UAI1

* Subscription
Woodgrove IT Production Environment

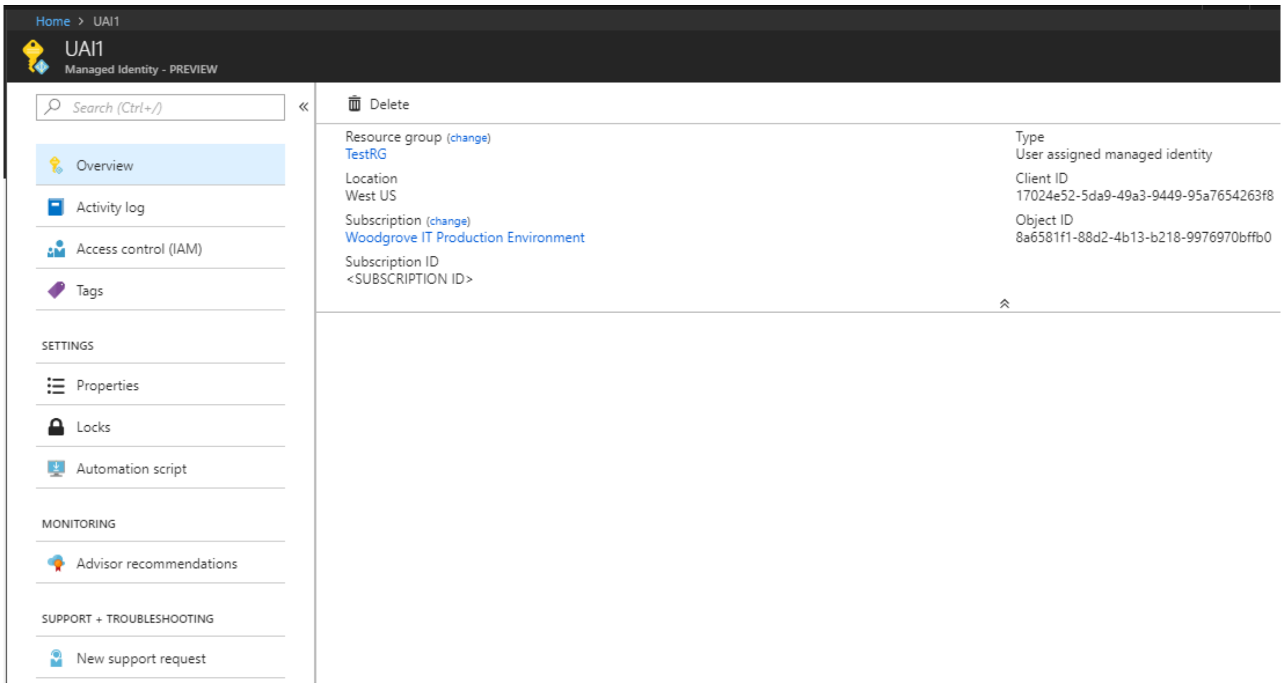
* Resource group
Create new [✓ Use existing]
TestRG

* Location
West US

List user-assigned managed identities

To list/read a user-assigned managed identity, your account needs the Managed Identity Operator or Managed Identity Contributor role assignment.

1. Sign in to the Azure portal using an account associated with the Azure subscription to list the user-assigned managed identities.
2. In the search box, type *Managed Identities*, and under Services, click **Managed Identities**.
3. A list of the user-assigned managed identities for your subscription is returned. To see the details of a user-assigned managed identity click its name.



Resource group (change)
TestRG
Location
West US
Subscription (change)
Woodgrove IT Production Environment
Subscription ID
<SUBSCRIPTION ID>

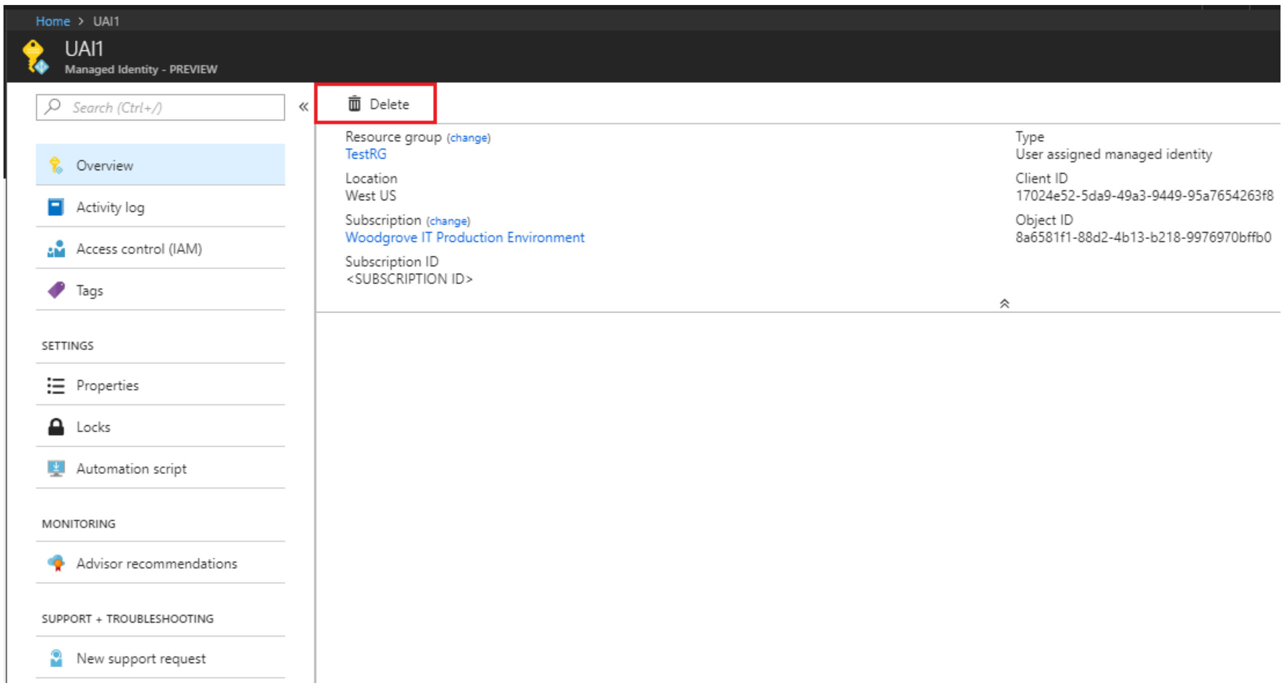
Type
User assigned managed identity
Client ID
17024e52-5da9-49a3-9449-95a7654263f8
Object ID
8a6581f1-88d2-4b13-b218-9976970bffb0

Delete a user-assigned managed identity

To delete a user-assigned managed identity, your account needs the Managed Identity Contributor role assignment.

Deleting a user assigned identity does not remove it from the VM or resource it was assigned to. To remove the user assigned identity from a VM see, Remove a user-assigned managed identity from a VM.

1. Sign in to the Azure portal using an account associated with the Azure subscription to delete a user-assigned managed identity.
2. Select the user-assigned managed identity and click **Delete**.
3. Under the confirmation box choose, **Yes**.



Resource group (change)
TestRG
Location
West US
Subscription (change)
Woodgrove IT Production Environment
Subscription ID
<SUBSCRIPTION ID>
Type
User assigned managed identity
Client ID
17024e52-5da9-49a3-9449-95a7654263f8
Object ID
8a6581f1-88d2-4b13-b218-9976970bffb0

Assign a role to a user-assigned managed identity

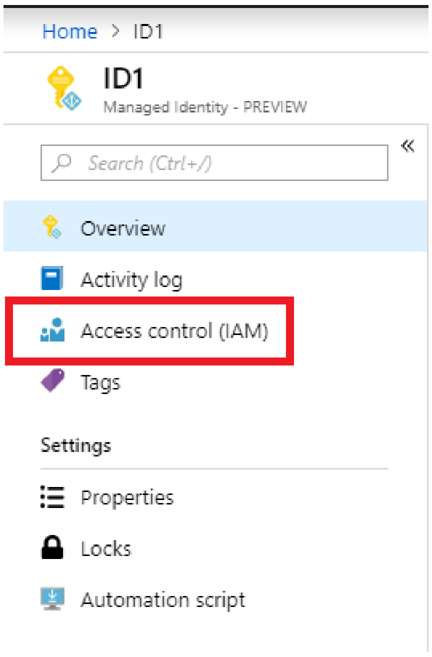
To assign a role to a user-assigned managed identity, your account needs the User Access Administrator role assignment.

1. Sign in to the Azure portal using an account associated with the Azure subscription to list the user-assigned managed identities.

2. In the search box, type *Managed Identities*, and under Services, click **Managed Identities**.

3. A list of the user-assigned managed identities for your subscription is returned. Select the user-assigned managed identity that you want to assign a role.

4. Select **Access control (IAM)** and then select **Add role assignment**.


Home > ID1

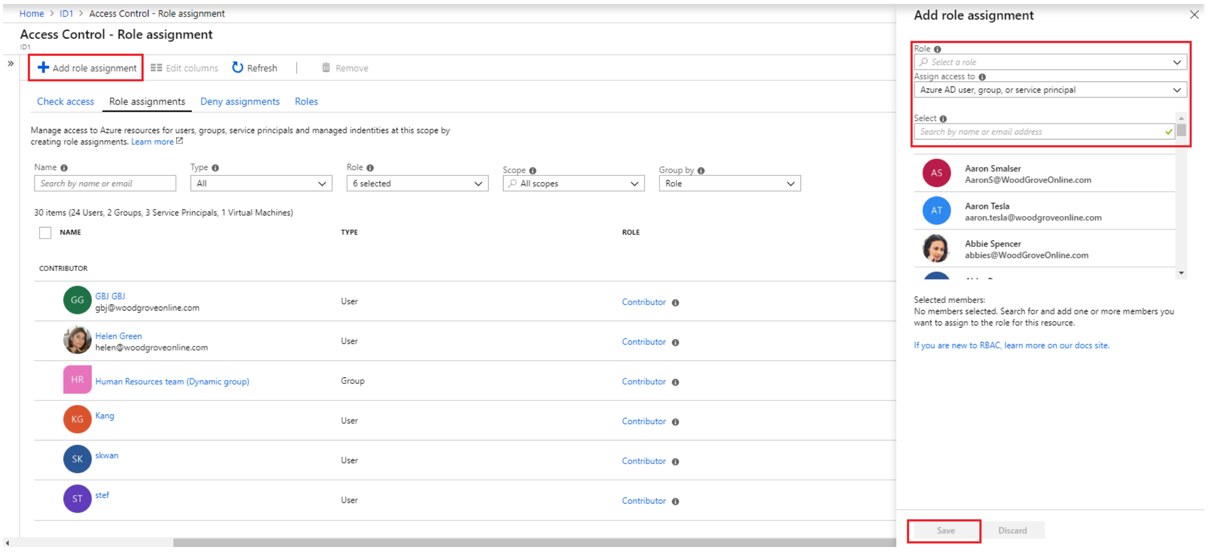
ID1

Managed Identity - PREVIEW

Search (Ctrl+/)

5. In the Add role assignment blade, configure the following values and then click **Save**:

   - **Role** - the role to assign
   - **Assign access to** - the resource to assign the user-assigned managed identity
   - **Select** - the member to assign access


Home > ID1 > Access Control - Role assignment

Access Control - Role assignment

ID1

Manage access to Azure resources for users, groups, service principals and managed indentities at this scope by creating role assignments. Learn more

Name: Search by name or email

Type: All

Role: 6 selected

Scope: All scopes

Group by: Role

30 items (24 Users, 2 Groups, 3 Service Principals, 1 Virtual Machines)

NAME	TYPE	ROLE
CONTRIBUTOR		
GBJ GBJ gbj@woodgroveonline.com	User	Contributor
Helen Green helen@woodgroveonline.com	User	Contributor
Human Resources team (Dynamic group)	Group	Contributor
Kang	User	Contributor
skwan	User	Contributor
stef	User	Contributor

Add role assignment

Role: Select a role

Assign access to: Azure AD user, group, or service principal

Select: Search by name or email address

Aaron Smalser AaronS@WoodGroveOnline.com

Aaron Tesla aaron.tesla@woodgroveonline.com

Abbie Spencer abbies@WoodGroveOnline.com

Selected members:
No members selected. Search for and add one or more members you want to assign to the role for this resource.

If you are new to RBAC, learn more on our docs site.

Create, list or delete a user-assigned managed identity using the Azure CLI

3/26/2019 • 3 minutes to read • Edit Online

User assigned managed identities are a preview feature of Azure Active Directory. Make sure you review the known issues before you begin. For more information about previews, see Supplemental Terms of Use for Microsoft Azure Previews.

Managed identities for Azure resources provides Azure services with a managed identity in Azure Active Directory. You can use this identity to authenticate to services that support Azure AD authentication, without needing credentials in your code.

In this article, you learn how to create, list and delete a user-assigned managed identity using Azure CLI.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- To run the CLI script examples, you have three options:
  - Use Azure Cloud Shell from the Azure portal (see next section).
  - Use the embedded Azure Cloud Shell via the "Try It" button, located in the top right corner of each code block.
  - Install the latest version of the Azure CLI (2.0.13 or later) if you prefer to use a local CLI console. Sign in to Azure using `az login`, using an account that is associated with the Azure subscription under which you would like to deploy the user-assigned managed identity.

Open Azure Cloud Shell

Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Select **Copy** to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.	
Open Cloud Shell in your browser.	
Select the **Cloud Shell** button on the menu in the upper-right corner of the Azure portal.	

Create a user-assigned managed identity

To create a user-assigned managed identity, your account needs the Managed Identity Contributor role

assignment.

Use the az identity create command to create a user-assigned managed identity. The `-g` parameter specifies the resource group where to create the user-assigned managed identity, and the `-n` parameter specifies its name. Replace the `<RESOURCE GROUP>` and `<USER ASSIGNED IDENTITY NAME>` parameter values with your own values:

IMPORTANT

When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

```
az identity create -g <RESOURCE GROUP> -n <USER ASSIGNED IDENTITY NAME>
```

List user-assigned managed identities

To list/read a user-assigned managed identity, your account needs the Managed Identity Operator or Managed Identity Contributor role assignment.

To list user-assigned managed identities, use the az identity list command. Replace the `<RESOURCE GROUP>` with your own value:

```
az identity list -g <RESOURCE GROUP>
```

In the json response, user-assigned managed identities have `"Microsoft.ManagedIdentity/userAssignedIdentities"` value returned for key, `type`.

`"type": "Microsoft.ManagedIdentity/userAssignedIdentities"`

Delete a user-assigned managed identity

To delete a user-assigned managed identity, your account needs the Managed Identity Contributor role assignment.

To delete a user-assigned managed identity, use the az identity delete command. The -n parameter specifies its name and the -g parameter specifies the resource group where the user-assigned managed identity was created. Replace the `<USER ASSIGNED IDENTITY NAME>` and `<RESOURCE GROUP>` parameters values with your own values:

```
az identity delete -n <USER ASSIGNED IDENTITY NAME> -g <RESOURCE GROUP>
```

NOTE

Deleting a user-assigned managed identity will not remove the reference, from any resource it was assigned to. Please remove those from VM/VMSS using the `az vm/vmss identity remove` command

Next steps

For a full list of Azure CLI identity commands, see az identity.

For information on how to assign a user-assigned managed identity to an Azure VM see, Configure managed identities for Azure resources on an Azure VM using Azure CLI

Create, list or delete a user-assigned managed identity using Azure PowerShell

3/26/2019 • 3 minutes to read • Edit Online

User assigned managed identities are a preview feature of Azure Active Directory. Make sure you review the known issues before you begin. For more information about previews, see Supplemental Terms of Use for Microsoft Azure Previews.

Managed identities for Azure resources provides Azure services with a managed identity in Azure Active Directory. You can use this identity to authenticate to services that support Azure AD authentication, without needing credentials in your code.

In this article, you learn how to create, list and delete a user-assigned managed identity using Azure PowerShell.

NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- Install the latest version of Azure PowerShell if you haven't already.
- If you are running PowerShell locally, you also need to:
  - Run `Connect-AzAccount` to create a connection with Azure.
  - Install the latest version of PowerShellGet.
  - Run `Install-Module -Name PowerShellGet -AllowPrerelease` to get the pre-release version of the `PowerShellGet` module (you may need to `Exit` out of the current PowerShell session after you run this command to install the `Az.ManagedServiceIdentity` module).
  - Run `Install-Module -Name Az.ManagedServiceIdentity -AllowPrerelease` to install the prerelease version of the `Az.ManagedServiceIdentity` module to perform the user-assigned managed identity operations in this article.

Create a user-assigned managed identity

To create a user-assigned managed identity, your account needs the Managed Identity Contributor role assignment.

To create a user-assigned managed identity, use the `New-AzUserAssignedIdentity` command. The `ResourceGroupName` parameter specifies the resource group where to create the user-assigned managed identity, and the `-Name` parameter specifies its name. Replace the `<RESOURCE GROUP>` and `<USER ASSIGNED IDENTITY NAME>` parameter values with your own values:

IMPORTANT

When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

```
New-AzUserAssignedIdentity -ResourceGroupName <RESOURCEGROUP> -Name <USER ASSIGNED IDENTITY NAME>
```

List user-assigned managed identities

To list/read a user-assigned managed identity, your account needs the Managed Identity Operator or Managed Identity Contributor role assignment.

To list user-assigned managed identities, use the [Get-AzUserAssigned] command. The `-ResourceGroupName` parameter specifies the resource group where the user-assigned managed identity was created. Replace the `<RESOURCE GROUP>` with your own value:

```
Get-AzUserAssignedIdentity -ResourceGroupName <RESOURCE GROUP>
```

In the response, user-assigned managed identities have `"Microsoft.ManagedIdentity/userAssignedIdentities"` value returned for key, `Type`.

`Type :Microsoft.ManagedIdentity/userAssignedIdentities`

Delete a user-assigned managed identity

To delete a user-assigned managed identity, your account needs the Managed Identity Contributor role assignment.

To delete a user-assigned managed identity, use the `Remove-AzUserAssignedIdentity` command. The `-ResourceGroupName` parameter specifies the resource group where the user-assigned identity was created and the `-Name` parameter specifies its name. Replace the `<RESOURCE GROUP>` and the `<USER ASSIGNED IDENTITY NAME>` parameters values with your own values:

```
Remove-AzUserAssignedIdentity -ResourceGroupName <RESOURCE GROUP> -Name <USER ASSIGNED IDENTITY NAME>
```

NOTE

Deleting a user-assigned managed identity will not remove the reference, from any resource it was assigned to. Identity assignments need to be removed separately.

Next steps

For a full list and more details of the Azure PowerShell managed identities for Azure resources commands, see Az.ManagedServiceIdentity.

Create, list and delete a user-assigned managed identity using Azure Resource Manager

3/26/2019 • 2 minutes to read • Edit Online

User assigned managed identities are a preview feature of Azure Active Directory. Make sure you review the known issues before you begin. For more information about previews, see Supplemental Terms of Use for Microsoft Azure Previews.

Managed identities for Azure resources provides Azure services with a managed identity in Azure Active Directory. You can use this identity to authenticate to services that support Azure AD authentication, without needing credentials in your code.

In this article, you create a user-assigned managed identity using an Azure Resource Manager.

It is not possible to list and delete a user-assigned managed identity using an Azure Resource Manager template. See the following articles to create and list a user-assigned managed identity:

- List user-assigned managed identity
- Delete user-assigned managed identity

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.

Template creation and editing

As with the Azure portal and scripting, Azure Resource Manager templates provide the ability to deploy new or modified resources defined by an Azure resource group. Several options are available for template editing and deployment, both local and portal-based, including:

- Using a custom template from the Azure Marketplace, which allows you to create a template from scratch, or base it on an existing common or QuickStart template.
- Deriving from an existing resource group, by exporting a template from either the original deployment, or from the current state of the deployment.
- Using a local JSON editor (such as VS Code), and then uploading and deploying by using PowerShell or CLI.
- Using the Visual Studio Azure Resource Group project to both create and deploy a template.

Create a user-assigned managed identity

To create a user-assigned managed identity, your account needs the Managed Identity Contributor role assignment.

To create a user-assigned managed identity, use the following template. Replace the `<USER ASSIGNED IDENTITY NAME>` value with your own values:

IMPORTANT

When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "resourceName": {
          "type": "string",
          "metadata": {
            "description": "<USER ASSIGNED IDENTITY NAME>"
          }
        }
  },
  "resources": [
    {
      "type": "Microsoft.ManagedIdentity/userAssignedIdentities",
      "name": "[parameters('resourceName')]",
      "apiVersion": "2018-11-30",
      "location": "[resourceGroup().location]"
    }
  ],
  "outputs": {
      "identityName": {
          "type": "string",
          "value": "[parameters('resourceName')]"
      }
  }
}
```

Next steps

For information on how to assign a user-assigned managed identity to an Azure VM using an Azure Resource Manager template see, Configure managed identities for Azure resources on an Azure VM using a templates.

Create, list or delete a user-assigned managed identity using REST API calls

3/26/2019 • 3 minutes to read • Edit Online

User assigned managed identities are a preview feature of Azure Active Directory. Make sure you review the known issues before you begin. For more information about previews, see Supplemental Terms of Use for Microsoft Azure Previews.

Managed identities for Azure resources provides Azure services the ability to authenticate to services that support Azure AD authentication, without needing credentials in your code.

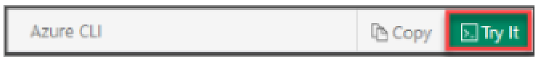
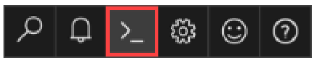
In this article, you learn how to create, list, and delete a user-assigned managed identity using CURL to make REST API calls.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section. **Be sure to review the difference between a system-assigned and user-assigned managed identity**.
- If you don't already have an Azure account, sign up for a free account before continuing.
- If you are using Windows, install the Windows Subsystem for Linux or use the Azure Cloud Shell in the Azure portal.
- If you use the Windows Subsystem for Linux or a Linux distribution OS, Install the Azure CLI local console.
- If you are using Azure CLI local console, sign in to Azure using `az login` with an account that is associated with the Azure subscription you would like to deploy or retrieve user-assigned managed identity information.
- Retrieve a Bearer access token using `az account get-access-token` to perform the following user-assigned managed identity operations.

Open Azure Cloud Shell

Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Select **Copy** to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.	
Open Cloud Shell in your browser.	
Select the **Cloud Shell** button on the menu in the upper-right corner of the Azure portal.	

Create a user-assigned managed identity

To create a user-assigned managed identity, your account needs the Managed Identity Contributor role

assignment.

IMPORTANT

When creating user assigned identities, only alphanumeric characters (0-9, a-z, A-Z), the underscore (_) and the hyphen (-) are supported. Additionally, the name should be atleast 3 characters and up to 128 characters in length for the assignment to VM/VMSS to work properly. Check back for updates. For more information, see FAQs and known issues.

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroup
s/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>?
api-version=2015-08-31-preview' -X PUT -d '{"loc
ation": "<LOCATION>"}' -H "Content-Type: application/json" -H "Authorization: Bearer <ACCESS TOKEN>"
```

```
PUT https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroup
s/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>?
api-version=2015-08-31-preview HTTP/1.1
```

Request headers

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Request body

NAME	DESCRIPTION
location	Required. Resource location.

List user-assigned managed identities

To list/read a user-assigned managed identity, your account needs the Managed Identity Operator or Managed Identity Contributor role assignment.

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities?api-version=2015-08-31-preview' -H
"Authorization: Bearer <ACCESS TOKEN>"
```

```
GET https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroups/<RESOURCE
GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities?api-version=2015-08-31-preview HTTP/1.1
```

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Delete a user-assigned managed identity

To delete a user-assigned managed identity, your account needs the Managed Identity Contributor role assignment.

> **NOTE**
> Deleting a user-assigned managed identity will not remove the reference from any resource it was assigned to. To remove a user-assigned managed identity from a VM using CURL see [Remove a user-assigned identity from an Azure VM](qs-configure-rest-vm.md#remove-a-user-assigned identity-from-an-azure-vm).

```
curl 'https://management.azure.com/subscriptions/<SUBSCRIPTION ID>/resourceGroup
s/<RESOURCE GROUP>/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED IDENTITY NAME>?
api-version=2015-08-31-preview' -X DELETE -H "Authorization: Bearer <ACCESS TOKEN>"
```

```
DELETE https://management.azure.com/subscriptions/80c696ff-5efa-4909-a64d-
f1b616f423ca/resourceGroups/TestRG/providers/Microsoft.ManagedIdentity/userAssignedIdentities/<USER ASSIGNED
IDENTITY NAME>?api-version=2015-08-31-preview HTTP/1.1
```

REQUEST HEADER	DESCRIPTION
Content-Type	Required. Set to `application/json`.
Authorization	Required. Set to a valid `Bearer` access token.

Next steps

For information on how to assign a user-assigned managed identity to an Azure VM/VMSS using CURL see, Configure managed identities for Azure resources on an Azure VM using REST API calls and Configure managed identities for Azure resources on a virtual machine scale set using REST API calls.

View the service principal of a managed identity in the Azure portal

3/26/2019 • 2 minutes to read • Edit Online

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, you learn how to view the service principal of a managed identity using the Azure portal.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section.
- If you don't already have an Azure account, sign up for a free account.
- Enable system assigned identity on a virtual machine or application.

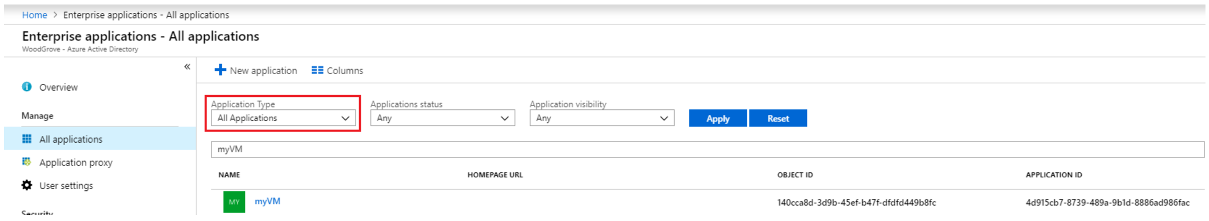
View the service principal

This procedure demonstrates how to view the service principal of a VM with system assigned identity enabled (the same steps apply for an application).

1. Click **Azure Active Directory** and then click **Enterprise applications**.

2. Under **Application Type**, choose **All Applications**.

3. In the search filter box, type the name of the VM or application that has managed identity enabled or choose it from the list presented.


Home > Enterprise applications - All applications
Enterprise applications - All applications
WoodGrove - Azure Active Directory
Overview
Manage
All applications
Application proxy
User settings
Security
New application
Columns
Apply
Reset

Application Type: All Applications
Applications status: Any
Application visibility: Any

myVM

NAME	HOMEPAGE URL	OBJECT ID	APPLICATION ID
MY myVM		140cca8d-3d9b-45ef-b47f-dfdfd449b8fc	4d915cb7-8739-489a-9b1d-8886ad986fac

Next steps

Managed identities for Azure resources

View the service principal of a managed identity using Azure CLI

3/26/2019 • 2 minutes to read • Edit Online

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication without having credentials in your code.

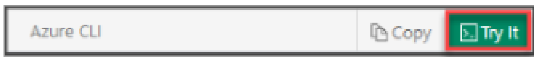
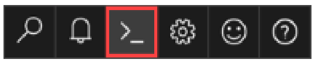
In this article, you learn how to view the service principal of a managed identity using Azure CLI.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section.
- If you don't already have an Azure account, sign up for a free account.
- Enable system assigned identity on a virtual machine or application.
- To run the CLI script examples, you have three options:
  - Use Azure Cloud Shell from the Azure portal (see next section).
  - Use the embedded Azure Cloud Shell via the "Try It" button, located in the top right corner of each code block.
  - Install the latest version of the Azure CLI if you prefer to use a local CLI console and sign in to Azure using `az login`

Open Azure Cloud Shell

Azure Cloud Shell is a free, interactive shell that you can use to run the steps in this article. Common Azure tools are preinstalled and configured in Cloud Shell for you to use with your account. Select **Copy** to copy the code, paste it in Cloud Shell, and then press Enter to run it. There are a few ways to open Cloud Shell:

Select **Try It** in the upper-right corner of a code block.	
Open Cloud Shell in your browser.	
Select the **Cloud Shell** button on the menu in the upper-right corner of the Azure portal.	

View the service principal

This following command demonstrates how to view the service principal of a VM or application with managed identity enabled. Replace `<VM or application name>` with your own values.

```
az ad sp list --display-name <VM or application name>
```

Next steps

For more information on managing Azure AD service principals using Azure CLI, see az ad sp.

View the service principal of a managed identity using PowerShell

3/26/2019 • 2 minutes to read • Edit Online

Managed identities for Azure resources provides Azure services with an automatically managed identity in Azure Active Directory. You can use this identity to authenticate to any service that supports Azure AD authentication, without having credentials in your code.

In this article, you learn how to view the service principal of a managed identity using PowerShell.

NOTE

This article has been updated to use the new Azure PowerShell Az module. You can still use the AzureRM module, which will continue to receive bug fixes until at least December 2020. To learn more about the new Az module and AzureRM compatibility, see Introducing the new Azure PowerShell Az module. For Az module installation instructions, see Install Azure PowerShell.

Prerequisites

- If you're unfamiliar with managed identities for Azure resources, check out the overview section.
- If you don't already have an Azure account, sign up for a free account.
- Enable system assigned identity on a virtual machine or application.
- Install the latest version of Azure PowerShell

View the service principal

This following command demonstrates how to view the service principal of a VM or application with system assigned identity enabled. Replace `<VM or application name>` with your own values.

```
Get-AzADServicePrincipal -DisplayName <VM or application name>
```

Next steps

For more information on viewing Azure AD service principals using PowerShell, see Get-AzADServicePrincipal.

FAQs and known issues with managed identities for Azure resources

3/26/2019 • 6 minutes to read • Edit Online

Managed identities for Azure resources is a feature of Azure Active Directory. Each of the Azure services that support managed identities for Azure resources are subject to their own timeline. Make sure you review the availability status of managed identities for your resource and known issues before you begin.

Frequently Asked Questions (FAQs)

NOTE

Managed identities for Azure resources is the new name for the service formerly known as Managed Service Identity (MSI).

Does managed identities for Azure resources work with Azure Cloud Services?

No, there are no plans to support managed identities for Azure resources in Azure Cloud Services.

Does managed identities for Azure resources work with the Active Directory Authentication Library (ADAL) or the Microsoft Authentication Library (MSAL)?

No, managed identities for Azure resources is not yet integrated with ADAL or MSAL. For details on acquiring a token for managed identities for Azure resources using the REST endpoint, see How to use managed identities for Azure resources on an Azure VM to acquire an access token.

What is the security boundary of managed identities for Azure resources?

The security boundary of the identity is the resource to which it is attached to. For example, the security boundary for a Virtual Machine with managed identities for Azure resources enabled, is the Virtual Machine. Any code running on that VM, is able to call the managed identities for Azure resources endpoint and request tokens. It is the similar experience with other resources that support managed identities for Azure resources.

What identity will IMDS default to if don't specify the identity in the request?

- If system assigned managed identity is enabled and no identity is specified in the request, IMDS will default to the system assigned managed identity.
- If system assigned managed identity is not enabled, and only one user assigned managed identity exists, IMDS will default to that single user assigned managed identity.
- If system assigned managed identity is not enabled, and multiple user assigned managed identities exist, then specifying a managed identity in the request is required.

Should I use the managed identities for Azure resources IMDS endpoint or the VM extension endpoint?

When using managed identities for Azure resources with VMs, we recommend using the IMDS endpoint. The Azure Instance Metadata Service is a REST Endpoint accessible to all IaaS VMs created via the Azure Resource Manager.

Some of the benefits of using managed identities for Azure resources over IMDS are:

- All Azure IaaS supported operating systems can use managed identities for Azure resources over IMDS.
- No longer need to install an extension on your VM to enable managed identities for Azure resources.

- The certificates used by managed identities for Azure resources are no longer present in the VM.
- The IMDS endpoint is a well-known non-routable IP address, only available from within the VM.
- 1000 user-assigned managed identities can be assigned to a single VM.

The managed identities for Azure resources VM extension is still available; however, we are no longer developing new functionality on it. We recommend switching to use the IMDS endpoint.

Some of the limitations of using the VM extension endpoint are:

- Limited support for Linux distributions: CoreOS Stable, CentOS 7.1, Red Hat 7.2, Ubuntu 15.04, Ubuntu 16.04
- Only 32 user-assigned managed identities can be assigned to the VM.

Note: The managed identities for Azure resources VM extension will be out of support in January 2019.

For more information on Azure Instance Metadata Service, see IMDS documentation

Will managed identities be recreated automatically if I move a subscription to another directory?

No. If you move a subscription to another directory, you will have to manually re-create them and grant Azure RBAC role assignments again.

- For system assigned managed identities: disable and re-enable.
- For user assigned managed identities: delete, re-create and attach them again to the necessary resources (e.g. virtual machines)

Can I use a managed identity to access a resource in a different directory/tenant?

No. Managed identities do not currently support cross-directory scenarios.

How do you restart the managed identities for Azure resources extension?

On Windows and certain versions of Linux, if the extension stops, the following cmdlet may be used to manually restart it:

```
Set-AzVMExtension -Name <extension name>  -Type <extension Type>  -Location <location> -Publisher
Microsoft.ManagedIdentity -VMName <vm name> -ResourceGroupName <resource group name> -ForceRerun <Any string
different from any last value used>
```

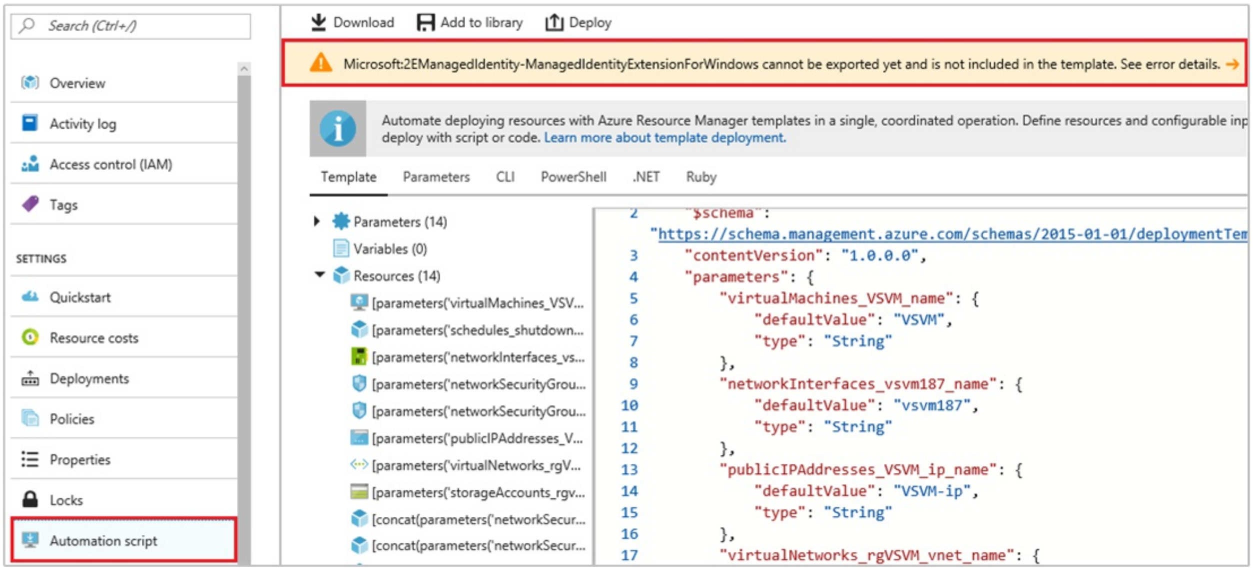
Where:

- Extension name and type for Windows is: ManagedIdentityExtensionForWindows
- Extension name and type for Linux is: ManagedIdentityExtensionForLinux

Known issues

"Automation script" fails when attempting schema export for managed identities for Azure resources extension

When managed identities for Azure resources is enabled on a VM, the following error is shown when attempting to use the “Automation script” feature for the VM, or its resource group:


Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
SETTINGS
Quickstart
Resource costs
Deployments
Policies
Properties
Locks
Automation script
Download
Add to library
Deploy
Template
Parameters
CLI
PowerShell
.NET
Ruby
Parameters (14)
Variables (0)
Resources (14)
[parameters('virtualMachines_VSV...
[parameters('schedules_shutdown...
[parameters('networkInterfaces_vs...
[parameters('networkSecurityGrou...
[parameters('networkSecurityGrou...
[parameters('publicIPAddresses_V...
[parameters('virtualNetworks_rgV...
[parameters('storageAccounts_rgv...
[concat(parameters('networkSecur...
[concat(parameters('networkSecur...

Microsoft:2EManagedIdentity-ManagedIdentityExtensionForWindows cannot be exported yet and is not included in the template. See error details. →

Automate deploying resources with Azure Resource Manager templates in a single, coordinated operation. Define resources and configurable inp
deploy with script or code. Learn more about template deployment.

```
    "$schema":
  "https://schema.management.azure.com/schemas/2015-01-01/deploymentTem
    "contentVersion": "1.0.0.0",
    "parameters": {
        "virtualMachines_VSVM_name": {
            "defaultValue": "VSVM",
            "type": "String"
        },
        "networkInterfaces_vsvm187_name": {
            "defaultValue": "vsvm187",
            "type": "String"
        },
        "publicIPAddresses_VSVM_ip_name": {
            "defaultValue": "VSVM-ip",
            "type": "String"
        },
        "virtualNetworks_rgVSVM_vnet_name": {
```

The managed identities for Azure resources VM extension (planned for deprecation in January 2019) does not currently support the ability to export its schema to a resource group template. As a result, the generated template does not show configuration parameters to enable managed identities for Azure resources on the resource. These sections can be added manually by following the examples in Configure managed identities for Azure resources on an Azure VM using a templates.

When the schema export functionality becomes available for the managed identities for Azure resources VM extension (planned for deprecation in January 2019), it will be listed in Exporting Resource Groups that contain VM extensions.

VM fails to start after being moved from resource group or subscription

If you move a VM in the running state, it continues to run during the move. However, after the move, if the VM is stopped and restarted, it will fail to start. This issue happens because the VM is not updating the reference to the managed identities for Azure resources identity and continues to point to it in the old resource group.

Workaround

Trigger an update on the VM so it can get correct values for the managed identities for Azure resources. You can do a VM property change to update the reference to the managed identities for Azure resources identity. For example, you can set a new tag value on the VM with the following command:

```
az  vm update -n <VM Name> -g <Resource Group> --set tags.fixVM=1
```

This command sets a new tag "fixVM" with a value of 1 on the VM.

By setting this property, the VM updates with the correct managed identities for Azure resources resource URI, and then you should be able to start the VM.

Once the VM is started, the tag can be removed by using following command:

```
az vm update -n <VM Name> -g <Resource Group> --remove tags.fixVM
```

VM extension provisioning fails

Provisioning of the VM extension might fail due to DNS lookup failures. Restart the VM, and try again.

> **NOTE**
>
> The VM extension is planned for deprecation by January 2019. We recommend you move to using the IMDS endpoint.

Transferring a subscription between Azure AD directories

Managed identities do not get updated when a subscription is moved/transferred to another directory. As a result, any existent system-assigned or user-assigned managed identities will be broken.

Workaround for managed identities in a subscription that has been moved to another directory:

- For system assigned managed identities: disable and re-enable.
- For user assigned managed identities: delete, re-create and attach them again to the necessary resources (e.g. virtual machines)

Moving a user-assigned managed identity to a different resource group/subscription

Moving a user-assigned managed identity to a different resource group will cause the identity to break. As a result, resources (e.g. VM) using that identity will not be able to request tokens for it.

Services that support managed identities for Azure resources

5/10/2019 • 4 minutes to read • Edit Online

Managed identities for Azure resources provide Azure services with an automatically managed identity in Azure Active Directory. Using a managed identity, you can authenticate to any service that supports Azure AD authentication without having credentials in your code. We are in the process of integrating managed identities for Azure resources and Azure AD authentication across Azure. Check back often for updates.

NOTE

Managed identities for Azure resources is the new name for the service formerly known as Managed Service Identity (MSI).

Azure services that support managed identities for Azure resources

The following Azure services support managed identities for Azure resources:

Azure Virtual Machines

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Available	Preview	Preview	Preview
User assigned	Preview	Preview	Preview	Preview

Refer to the following list to configure managed identity for Azure Virtual Machines (in regions where available):

- Azure portal
- PowerShell
- Azure CLI
- Azure Resource Manager templates
- REST

Azure Virtual Machine Scale Sets

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Available	Preview	Preview	Preview
User assigned	Preview	Preview	Preview	Preview

Refer to the following list to configure managed identity for Azure Virtual Machine Scale Sets (in regions where available):

- Azure portal
- PowerShell

- Azure CLI
- Azure Resource Manager templates
- REST

Azure App Service

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Available	Available	Available	Available
User assigned	Preview	Not available	Not available	Not available

Refer to the following list to configure managed identity for Azure App Service (in regions where available):

- Azure portal
- Azure CLI
- Azure PowerShell
- Azure Resource Manager template

Azure Blueprints

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Preview	Not available	Not available	Not available
User assigned	Preview	Not available	Not available	Not available

Refer to the following list to use a managed identity with Azure Blueprints:

- Azure portal - blueprint assignment
- REST API - blueprint assignment

Azure Functions

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Available	Available	Available	Available
User assigned	Preview	Not available	Not available	Not available

Refer to the following list to configure managed identity for Azure Functions (in regions where available):

- Azure portal
- Azure CLI
- Azure PowerShell
- Azure Resource Manager template

Azure Logic Apps

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Preview	Preview	Not available	Preview
User assigned	Not available	Not available	Not available	Not available

Refer to the following list to configure managed identity for Azure Logic Apps (in regions where available):

- Azure portal
- Azure Resource Manager template

Azure Data Factory V2

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Available	Not available	Not available	Not available
User assigned	Not available	Not available	Not available	Not available

Refer to the following list to configure managed identity for Azure Data Factory V2 (in regions where available):

- Azure portal
- PowerShell
- REST
- SDK

Azure API Management

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Available	Available	Not available	Not available
User assigned	Not available	Not available	Not available	Not available

Refer to the following list to configure managed identity for Azure API Management (in regions where available):

- Azure Resource Manager template

Azure Container Instances

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
System assigned	Linux: Preview Windows: Not available	Not available	Not available	Not available

MANAGED IDENTITY TYPE	ALL GENERALLY AVAILABLE GLOBAL AZURE REGIONS	AZURE GOVERNMENT	AZURE GERMANY	AZURE CHINA 21VIANET
User assigned	Linux: Preview Windows: Not available	Not available	Not available	Not available

Refer to the following list to configure managed identity for Azure Container Instances (in regions where available):

- Azure CLI
- Azure Resource Manager template
- YAML

Azure services that support Azure AD authentication

The following services support Azure AD authentication, and have been tested with client services that use managed identities for Azure resources.

Azure Resource Manager

Refer to the following list to configure access to Azure Resource Manager:

- Assign access via Azure portal
- Assign access via Powershell
- Assign access via Azure CLI
- Assign access via Azure Resource Manager template

CLOUD	RESOURCE ID	STATUS
Azure Global	`https://management.azure.com/`	Available
Azure Government	`https://management.usgovcloudapi.net/`	Available
Azure Germany	`https://management.microsoftazure.de/`	Available
Azure China 21Vianet	`https://management.chinacloudapi.cn`	Available

Azure Key Vault

CLOUD	RESOURCE ID	STATUS
Azure Global	`https://vault.azure.net`	Available
Azure Government	`https://vault.usgovcloudapi.net`	Available
Azure Germany	`https://vault.microsoftazure.de`	Available
Azure China 21Vianet	`https://vault.azure.cn`	Available

Azure Data Lake

CLOUD	RESOURCE ID	STATUS
Azure Global	`https://datalake.azure.net/`	Available
Azure Government		Not Available
Azure Germany		Not Available
Azure China 21Vianet		Not Available

Azure SQL

CLOUD	RESOURCE ID	STATUS
Azure Global	`https://database.windows.net/`	Available
Azure Government	`https://database.usgovcloudapi.net/`	Available
Azure Germany	`https://database.cloudapi.de/`	Available
Azure China 21Vianet	`https://database.chinacloudapi.cn/`	Available

Azure Event Hubs

CLOUD	RESOURCE ID	STATUS
Azure Global	`https://eventhubs.azure.net`	Preview
Azure Government		Not Available
Azure Germany		Not Available
Azure China 21Vianet		Not Available

Azure Service Bus

CLOUD	RESOURCE ID	STATUS
Azure Global	`https://servicebus.azure.net`	Preview
Azure Government		Not Available
Azure Germany		Not Available
Azure China 21Vianet		Not Available

Azure Storage blobs and queues

CLOUD	RESOURCE ID	STATUS
Azure Global	`https://storage.azure.com/`	Available

CLOUD	RESOURCE ID	STATUS
Azure Government	`https://storage.azure.com/`	Available
Azure Germany	`https://storage.azure.com/`	Available
Azure China 21Vianet	`https://storage.azure.com/`	Available

Azure Analysis Services

CLOUD	RESOURCE ID	STATUS
Azure Global	`https://*.asazure.windows.net`	Available
Azure Government	`https://*.asazure.usgovcloudapi.net`	Available
Azure Germany	`https://*.asazure.cloudapi.de`	Available
Azure China 21Vianet	`https://*.asazure.chinacloudapi.cn`	Available