

AAD - Managed Service Identity

Last updated by | Vitor Tomaz | Feb 18, 2021 at 2:30 AM PST

Contents

- [Issue](#)
 - [Managed Service Identity supportability](#)
- [Setup](#)
- [More Information](#)
 - [Limitations](#)
 - [How does the managed identities for Azure resources work?](#)
 - [How a system-assigned managed identity works with an A...](#)
 - [How a user-assigned managed identity works with an Azur...](#)
- [Public Doc Reference](#)

Issue

Managed Service Identity (MSI) allows Azure resources the ability to identify themselves to Azure Active Directory without needing any explicit credentials like the username and password. And this happens by way of pre-configuring this identity information in the Azure Active Directory by "trusting" those resources.

These resources then can access the data and/or services without requiring explicit credentials. This enables these resources to be utilized for applications that communicate with Azure services. The independence from managing identity, hard coding usernames and passwords and security concerns are alleviated with Managed Service Identity.

Managed Service Identity supportability

The supportability of Managed Service Identity is divided in two categories.

- One category is the Azure services that natively support MSI.
- The other category is the extended services that support Azure Active Directory authentication that can indirectly benefit from MSI via the first category.

In other words, following Azure services currently support Managed Service Identity for direct implementation with them without any workarounds:


- Azure Virtual Machines
- Azure App Service
- Azure Functions
- Azure Data Factory V2
- **The Azure SQL service DOES NOT directly/natively support Managed Service Identity.**
 - In order to use MSI with Azure SQL Database, we must first use the Azure services that natively support MSI to create identity that can be provisioned on Azure SQL Database. And then, use this

identity in the end applications with the access tokens issued by the respective Azure service

- [Services that support managed identities for Azure resources](#) 

Setup


To follow up step-by-step follow information on [Using Managed Service Identity \(MSI\) to authenticate on Azure SQL DB](#) 

1. Creating MSI
2. Adding and granting permissions on SQL DB
3. Sample code in Powershell and also in .NET using NuGet [Microsoft.Azure.Services.AppAuthentication](#) 

You can also check source information on public links below

More Information

Limitations

Be aware of some limitations as documented at [Troubleshooting problems related to Azure AD authentication with Azure SQL DB and DW] (<https://techcommunity.microsoft.com/t5/azure-sql-database/troubleshooting-problems-related-to-azure-ad-authentication-with/ba-p/1062991> ).

Limitation	Problem description	Troubleshooting
Service principal not able to create a new Azure AD user	Currently, this operation is not supported	Create AAD user FROM EXTERNAL PROVIDER - TSQL fails

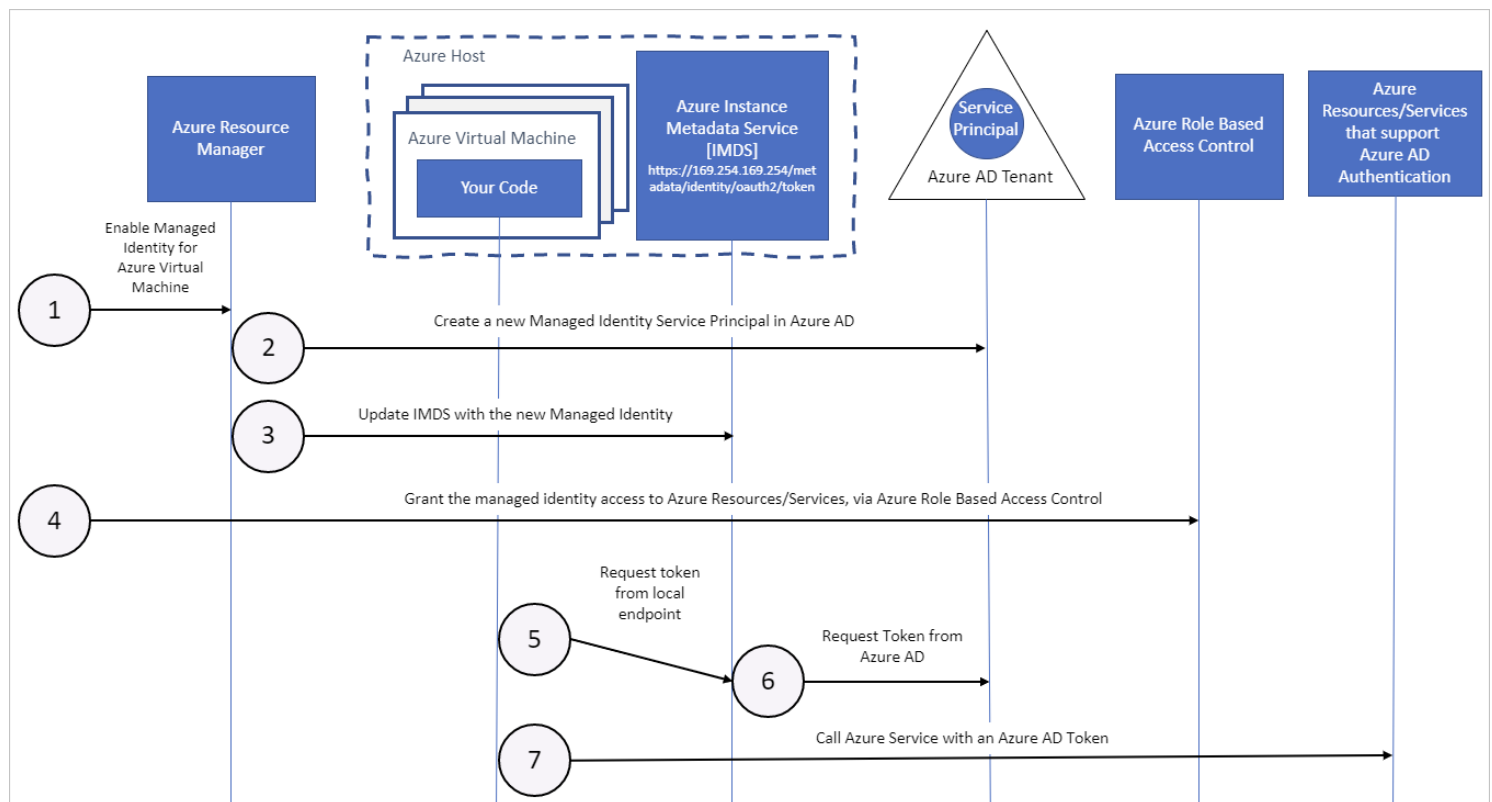
How does the managed identities for Azure resources work?

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 life cycle 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 life cycle of a user-assigned identity is managed separately from the life cycle 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. Also, when a User-Assigned or System-Assigned Identity is created, the Managed Identity Resource Provider (MSRP) issues a certificate internally to that identity.

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):



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 by updating the Azure Instance Metadata Service identity endpoint with the service principal client ID and certificate.
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.
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 and updates the Azure Instance Metadata Service identity endpoint with the user-assigned managed identity service principal client ID and certificate.
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.
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.

Public Doc Reference

- [Using Managed Service Identity \(MSI\) to authenticate on Azure SQL DB](#) ☐
- [Keep credentials out of code: Introducing Azure AD Managed Service Identity](#) ☐
- [Azure AD-managed identities for Azure resources documentation](#) ☐
- [Azure AD Managed Service Identity](#) ☐
- [Demystifying Managed Service Identities on Azure](#) ☐
- [Tutorial: Use a Windows VM system-assigned managed identity to access Azure SQL](#) ☐
- [Tutorial: Secure Azure SQL Database connection from App Service using a managed identity](#) ☐
- [What are managed identities for Azure resources?](#) ☐
- [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 to acquire an access token](#) ☐

How good have you found this content?

