**Managed identity types**

There are two types of managed identities:

* **System-assigned**. Some Azure resources, such as virtual machines allow you to enable a managed identity directly on the resource. When you enable a system-assigned managed identity:
  + A service principal of a special type is created in Microsoft Entra ID for the identity. The service principal is tied to the lifecycle of that Azure resource. When the Azure resource is deleted, Azure automatically deletes the service principal for you.
  + By design, only that Azure resource can use this identity to request tokens from Microsoft Entra ID.
  + You authorize the managed identity to have access to one or more services.
  + The name of the system-assigned service principal is always the same as the name of the Azure resource it's created for. For a deployment slot, the name of its system-assigned identity is <app-name>/slots/<slot-name>.
* **User-assigned**. You may also create a managed identity as a standalone Azure resource. You can [create a user-assigned managed identity](https://learn.microsoft.com/en-us/entra/identity/managed-identities-azure-resources/how-manage-user-assigned-managed-identities?pivots=identity-mi-methods-azp) and assign it to one or more Azure Resources. When you enable a user-assigned managed identity:
  + A service principal of a special type is created in Microsoft Entra ID for the identity. The service principal is managed separately from the resources that use it.
  + User-assigned identities can be used by multiple resources.
  + You authorize the managed identity to have access to one or more services.

User-assigned identities, which are provisioned independently from compute and can be assigned to multiple compute resources, are the recommended managed identity type for Microsoft services.

Resources that support system assigned managed identities allow you to:

* Enable or disable managed identities at the resource level.
* Use role-based access control (RBAC) to [grant permissions](https://learn.microsoft.com/en-us/azure/role-based-access-control/role-assignments-portal).
* View the create, read, update, and delete (CRUD) operations in [Azure Activity logs](https://learn.microsoft.com/en-us/azure/azure-monitor/essentials/activity-log).
* View sign in activity in Microsoft Entra ID [sign in logs](https://learn.microsoft.com/en-us/entra/identity/monitoring-health/concept-sign-ins).

If you choose a user assigned managed identity instead:

* You can [create, read, update, and delete](https://learn.microsoft.com/en-us/entra/identity/managed-identities-azure-resources/how-manage-user-assigned-managed-identities?pivots=identity-mi-methods-azp) the identities.
* You can use RBAC role assignments to [grant permissions](https://learn.microsoft.com/en-us/azure/role-based-access-control/role-assignments-portal).
* User assigned managed identities can be used on more than one resource.
* CRUD operations are available for review in [Azure Activity logs](https://learn.microsoft.com/en-us/azure/azure-monitor/essentials/activity-log).
* View sign in activity in Microsoft Entra ID [sign in logs](https://learn.microsoft.com/en-us/entra/identity/monitoring-health/concept-sign-ins).

Operations on managed identities can be performed by using an Azure Resource Manager template, the Azure portal, Azure CLI, PowerShell, and REST APIs.

**Differences between system-assigned and user-assigned managed identities**

| **Property** | **System-assigned managed identity** | **User-assigned managed identity** |
| --- | --- | --- |
| Creation | Created as part of an Azure resource (for example, Azure Virtual Machines or Azure App Service). | Created as a stand-alone Azure resource. |
| Life cycle | Shared life cycle 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. |
| Sharing across Azure resources | Can’t 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 contained within a single Azure resource. Workloads needing independent identities. For example, an application that runs on a single virtual machine. | Workloads that run on multiple resources and can share a single identity. Workloads needing preauthorization 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. |

A service principal in Azure is a security identity that an application, service, or automation tool uses to access Azure resources. It's essentially a "non-human" identity, distinct from individual user accounts, and is used to grant controlled access to resources within Azure Active Directory.

Key aspects of Azure service principals:

* **Application-based:**

Service principals are associated with an application registration within Azure Active Directory.

* **Secure Access:**

They provide a secure way for applications to interact with Azure resources without needing user credentials.

* **Controlled Permissions:**

Access to resources is managed through role-based access control (RBAC), allowing you to specify exactly what resources the service principal can access and at what level.

* **Automation:**

They are frequently used for automation tasks, such as deploying resources, managing infrastructure, or interacting with Azure services.

* **Different from Managed Identities:**

Unlike Managed Identities, which are automatically created and managed by Azure for specific resources, service principals need to be manually created and configured.

* **Authentication:**

Service principals authenticate using credentials like client secrets or certificates.

* **Use Cases:**

Service principals are used in various scenarios, including:

* + Connecting to Azure resources in CI/CD pipelines.
  + Automating deployments with tools like Azure DevOps.
  + Accessing Azure services like Azure Container Registry.
  + Integrating with other Azure services like Azure Databricks.

In essence, service principals enable secure and automated access to Azure resources, allowing applications and services to interact with the cloud platform without relying on individual user accounts.