**Create a service principal**

**Create a Service Principal**

az ad sp create-for-rbac --name <service-principal-name>

This command will return a JSON object containing the following information: *appId* - the application (client) ID; password: The *client secret* (use this for authentication); *tenant*: The Azure AD tenant ID.

{

"appId": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",

"displayName": "my-service-principal",

"password": "your-client-secret",

"tenant": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"

}

By default, this above command does not return the principalId in its output. We can get by:

principalId = $(az ad sp show --id your-app-id --query objectId -o tsv)

The result is like this:

principalId="aef1cdb5-b4ae-4e47-bf44-e5cd31401bc5"

**Assign Roles to the Service Principal**

az role assignment create --assignee $principalId --role Contributor --scope /subscriptions/<subscription-id>/resourceGroups/<resource-group-name>/providers/Microsoft.Web/sites/<app-service-name>

*Note:If you're using --assignee directly without specifying whether it's an appId or principalId, Azure CLI will try to resolve this automatically.*

**Authenticate to Access using the Service Principal**

Once you have the client ID (appId), client secret (password), and tenant ID (tenant), you can authenticate using the az login command with the service principal.

az login --service-principal -u <appId> -p <password> --tenant <tenant>

az webapp list --resource-group your\_resource\_group

Or we can use one single command line for access to the webapp list:

az webapp list --resource-group your\_resource\_group --subscription your\_subscription\_id --service-principal -u your\_app\_id -p your\_password --tenant your\_tenant\_id

**Create an user-assigned managed identity**

**Create an User-assigned Identity**

az identity create --name <identity-name> --resource-group <resource-group-name> --location <region>

This command will output JSON with details about the created identity including the id and principalId which you might need for further configurations.

{

"clientID": "a9f2c3a0-8b21-4fb9-bd2b-67e4de9a8c3d",

"id": "/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.ManagedIdentity/userAssignedIdentities/{identity-name}",

"principalId": "aef1cdb5-b4ae-4e47-bf44-e5cd31401bc5",

"tenantId": "e4d61e72-b153-4b7b-80b8-63cbce6d058b",

"location": "eastus",

"name": "{identity-name}",

"type": "Microsoft.ManagedIdentity/userAssignedIdentities"

}

id=$(az identity show --resource-group <resource-group-name> --name <identity-name> --query id --output tsv)

principalId=$(az identity show --resource-group <resource-group-name> --name <identity-name> --query principalId --output tsv)

clientId=$(az identity show --resource-group <resource-group-name> --name <identity-name> --query clientId --output tsv)

**Assign the Identity to Azure resources(like a VM)**

After creating the identity, you can assign it to Azure resources, app services, etc., by updating those resources with the identity's information. Here's an example for assigning to a VM:

az vm identity assign --resource-group <resource-group-name> --name <vm-name> --identities $id

<identity-resource-id> is the id(*resource ID*) of your user-assigned managed identity.

**Assign Roles to the Identity**

Now that the managed identity is assigned to the virtual machine, you can assign necessary roles to the identity. For example, if you want the VM to access an Azure Key Vault, you should assign the Reader or Secret Management role to the identity on the Key Vault.

az role assignment create --assignee $principalId --role <role-name> --scope /subscriptions/<subscription-id>/resourceGroups/<resource-group-name>/providers/Microsoft.KeyVault/vaults/<keyvault-name>

**Authenticate to Access using the Token of the Identity**

Finally, in the VM, it is able to use the clientId of the user-assigned identity to access the key vault.

# Get the access token for the user-assigned identity

token=$(az identity oauth2-token --client-id $clientId --resource https://myKeyVault.vault.azure.net --query accessToken -o tsv)

# Access the secret from the Key Vault using the token

curl -X GET -H "Authorization: Bearer $token" https://myKeyVault.vault.azure.net/secrets/mySecret?api-version=7.2

--resource <https://vault.azure.net:> This specifies the resource you are requesting the token for. This means the token being requested will be used to authenticate to Azure Key Vault.