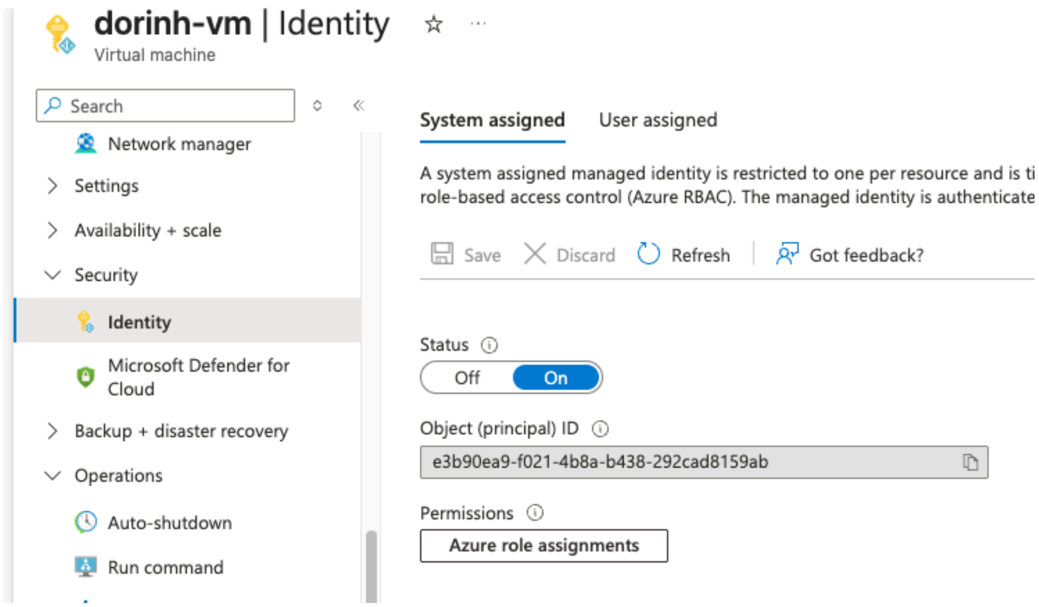
Azure Blob Storage File Upload using Python

1. Initial Setup:
    - Make sure Python is installed on your VM
2. Package Installation:
    - Ensure the following packages are installed:
        - azure-identity: For Azure SDK authentication.
            - Install with: *pip install azure-identity*
        - azure-storage-blob: SDK for Azure Blob Storage operations.
            - Install with: *pip install azure-storage-blob*
3. Code Setup:
    - Refer to the Lab2/main.py directory and integrate the provided Python code into your project.

Enable system-assigned managed identity on an existing VM



A system assigned managed identity is restricted to one per resource and is ti
role-based access control (Azure RBAC). The managed identity is authenticate

Status: Off [✓ On]

Object (principal) ID: e3b90ea9-f021-4b8a-b438-292cad8159ab

Permissions: Azure role assignments

1. Go to the Virtual Machine in the Azure Portal
    - Open https://portal.azure.com
    - Navigate to Virtual Machines
    - Select the VM you want to enable the identity on.
2. Enable the System-Assigned Managed Identity
    - Inside the VM blade, in the left menu, click Identity

- Under the System assigned tab:
- Switch Status: Off → On
- Click **Save**

Azure will automatically create a Managed Identity and associate it with the VM.

!!! Now you have a service principal for your VM, make sure you provide it the proper roles in order to access the blob.
Test the connection

Remove system-assigned managed identity from a VM

1. Go to Azure Portal → Virtual Machines
2. Select your VM
3. In the left menu, click Identity
4. Under the System assigned tab:
5. Change Status: On → Off
6. Click Save

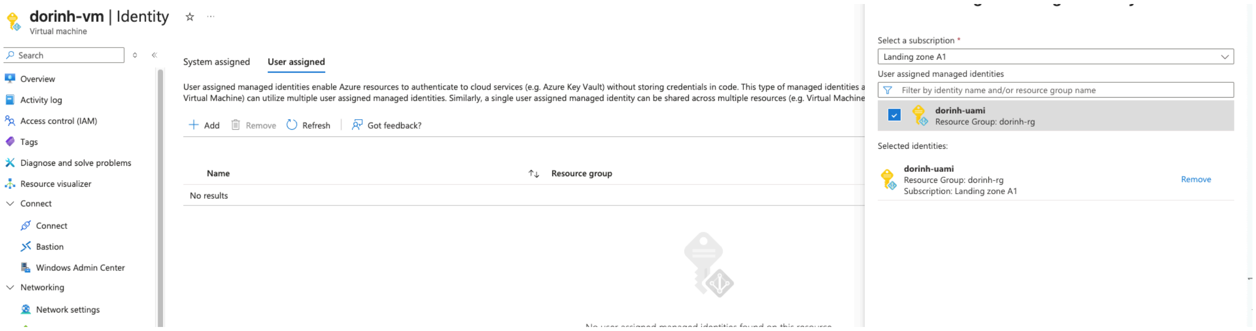
Create a user-assigned managed identity

1. Go to Azure Portal
2. Search for Managed Identities
3. Click Create
4. Select:
    a. Subscription
    b. Resource group: your-rg
    c. Region
    d. Name (example: yourname-uami)
5. Click Review + Create
6. Click Create

Assign the User-Assigned Identity to a VM

1. Go to Virtual Machines
2. Select your VM
3. In the left menu, click Identity
4. Open the User assigned tab
5. Click + Add
6. Select the identity you created (myUAMI)

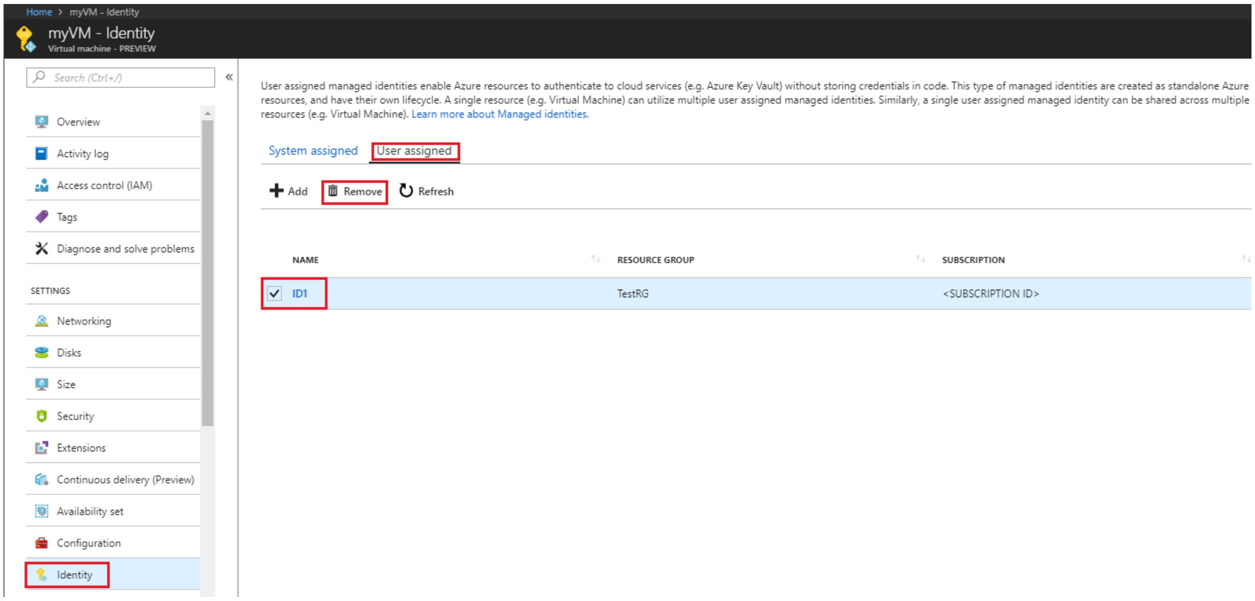
7. Click Add



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 a 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

Remove a user-assigned managed identity from a VM

1. Go to Azure Portal
2. Open Virtual Machines
3. Select the VM
4. In the left menu, click Identity
5. Go to the User assigned tab
6. Find the identity you want to remove
7. Click the Delete (trash bin) icon or Remove
8. Confirm the removal

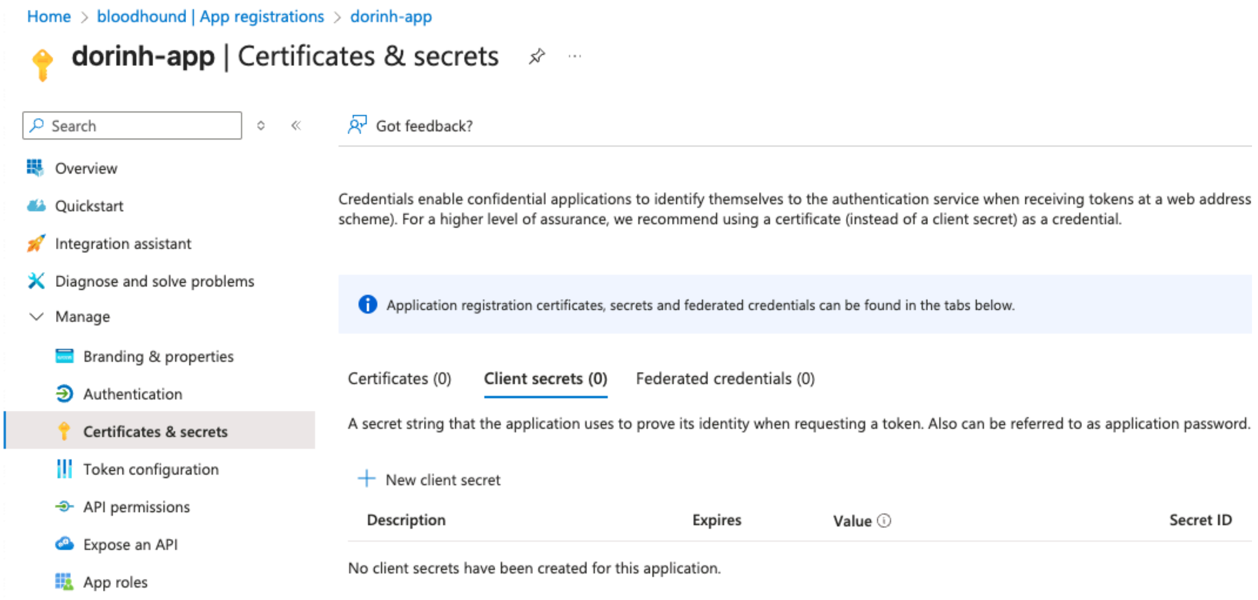


User assigned managed identities enable Azure resources to authenticate to cloud services (e.g. Azure Key Vault) without storing credentials in code. This type of managed identities are created as standalone Azure resources, and have their own lifecycle. A single resource (e.g. Virtual Machine) can utilize 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>

Use a service principal for authentication

1. Open the Azure Portal
    - Go to: https://portal.azure.com
2. Open Microsoft Entra ID
    - In the left menu, search for "Microsoft Entra ID"
    - Click it.
3. Go to App Registrations
    - In the Microsoft Entra ID blade, click App registrations
    - Click + New registration
4. Fill in the App Registration details
    - Name:
        i. Enter a name (e.g., myname-App).
    - Supported account types: choose one:
        i. Accounts in this organizational directory only
    - Redirect URI:
        i. Leave empty unless you need it for OAuth login.
5. Click Create
    - Azure will create your app registration.
6. You will now see:
    - Application (client) ID
    - Directory (tenant) ID
    - Object ID
    - Create a Client secret and write it down



Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web address scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Application registration certificates, secrets and federated credentials can be found in the tabs below.

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

Description	Expires	Value ⓘ	Secret ID

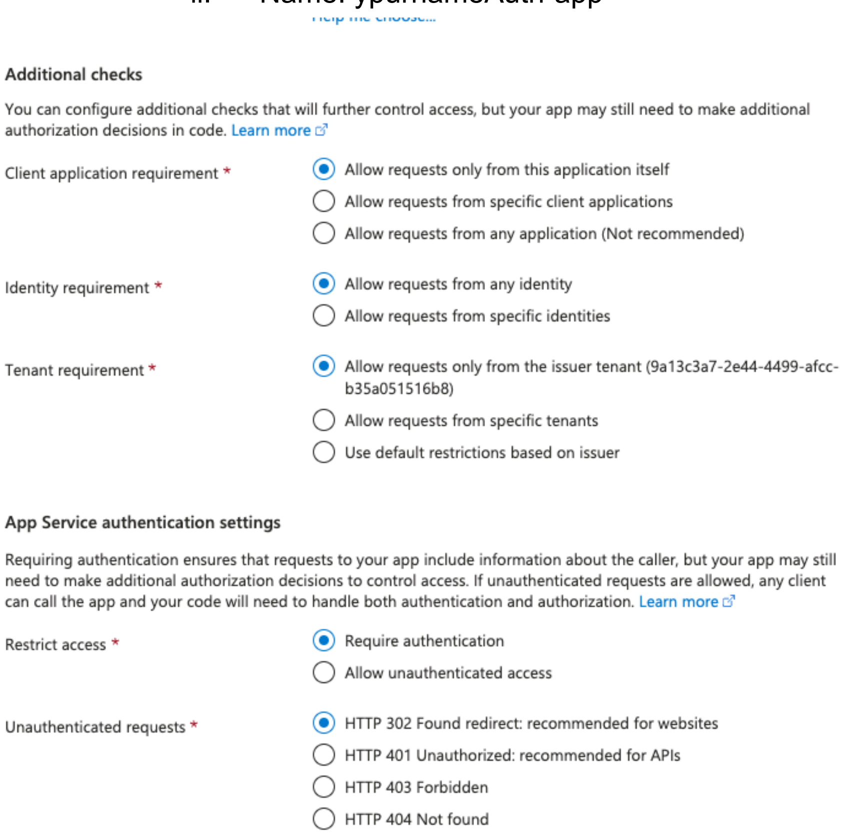
No client secrets have been created for this application.

7. Assign a role in the storage account for the Service Principal that you created
8. Open a CMD in your VM and set:
    - set AZURE_TENANT_ID="your-tenant-i"

- set AZURE_CLIENT_ID="your-client-id"
- set AZURE_CLIENT_SECRET="your-secret-value"

Add an Identity Provider to an App Service

1. Open the App Service
   - Go to Azure Portal
   - Open App Services
   - Click your existing App Service
2. Open the Authentication Settings
   - In the left menu, select Authentication
   - Click Add identity provider
3. Choose Identity Provider
   - Under Identity provider, select:
   - Microsoft (this means Microsoft Entra ID / Azure AD)
4. Create a new app registration automatically
   - Choose:
   - Identity provider: Microsoft
      1. App registration type: Create new app registration
      2. Name: ypurnameAuth-app



Help me choose...

Additional checks

You can configure additional checks that will further control access, but your app may still need to make additional authorization decisions in code. Learn more

Client application requirement *
- ● Allow requests only from this application itself
- ○ Allow requests from specific client applications
- ○ Allow requests from any application (Not recommended)

Identity requirement *
- ● Allow requests from any identity
- ○ Allow requests from specific identities

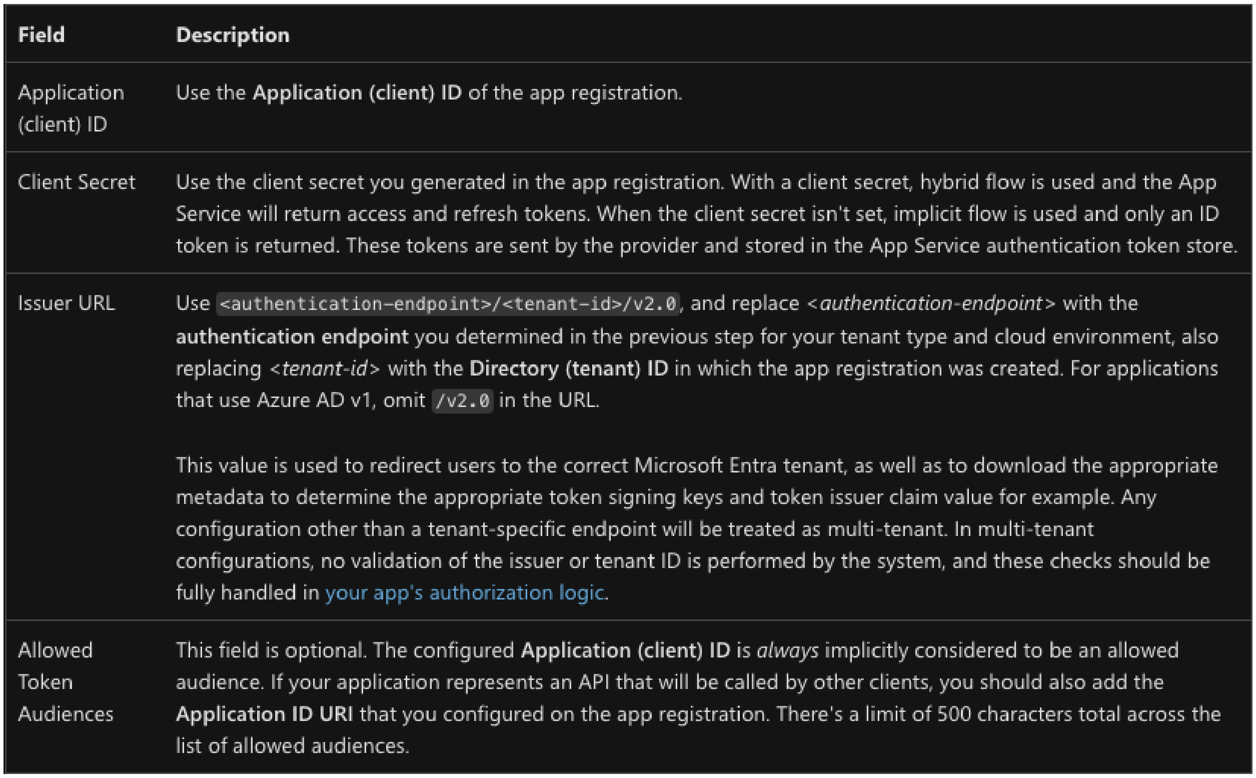
Tenant requirement *
- ● Allow requests only from the issuer tenant (9a13c3a7-2e44-4499-afcc-b35a051516b8)
- ○ Allow requests from specific tenants
- ○ Use default restrictions based on issuer

App Service authentication settings

Requiring authentication ensures that requests to your app include information about the caller, but your app may still need to make additional authorization decisions to control access. If unauthenticated requests are allowed, any client can call the app and your code will need to handle both authentication and authorization. Learn more

Restrict access *
- ● Require authentication
- ○ Allow unauthenticated access

Unauthenticated requests *
- ● HTTP 302 Found redirect: recommended for websites
- ○ HTTP 401 Unauthorized: recommended for APIs
- ○ HTTP 403 Forbidden
- ○ HTTP 404 Not found



Field	Description
Application (client) ID	Use the **Application (client) ID** of the app registration.
Client Secret	Use the client secret you generated in the app registration. With a client secret, hybrid flow is used and the App Service will return access and refresh tokens. When the client secret isn't set, implicit flow is used and only an ID token is returned. These tokens are sent by the provider and stored in the App Service authentication token store.
Issuer URL	Use `<authentication-endpoint>/<tenant-id>/v2.0`, and replace <*authentication-endpoint*> with the **authentication endpoint** you determined in the previous step for your tenant type and cloud environment, also replacing <*tenant-id*> with the **Directory (tenant) ID** in which the app registration was created. For applications that use Azure AD v1, omit `/v2.0` in the URL. This value is used to redirect users to the correct Microsoft Entra tenant, as well as to download the appropriate metadata to determine the appropriate token signing keys and token issuer claim value for example. Any configuration other than a tenant-specific endpoint will be treated as multi-tenant. In multi-tenant configurations, no validation of the issuer or tenant ID is performed by the system, and these checks should be fully handled in your app's authorization logic.
Allowed Token Audiences	This field is optional. The configured **Application (client) ID** is *always* implicitly considered to be an allowed audience. If your application represents an API that will be called by other clients, you should also add the **Application ID URI** that you configured on the app registration. There's a limit of 500 characters total across the list of allowed audiences.