Get Azure AD tokens for service principals

learn.microsoft.com/en-us/azure/databricks/dev-tools/api/latest/aad/service-prin-aad-token

- Article
- 11/21/2022
- 10 minutes to read

This article describes how a service principal defined in Azure Active Directory (Azure AD) can also act as a principal on which authentication and authorization policies can be enforced in Azure Databricks. Service principals in an Azure Databricks workspace can have different fine-grained access control than regular users (user principals).

A service principal acts as a client role and uses the OAuth 2.0 client credentials flow to authorize access to Azure Databricks resources.

You can manage service principals by using the Databricks SCIM API 2.0 (ServicePrincipals) for workspaces API or by using the following procedure from the Azure portal.

You can also use the Microsoft Authentication Library (MSAL) to programmatically get an Azure AD access token for a user instead of a service principal. See Get Azure AD tokens for users by using MSAL.

Provision a service principal in Azure portal

1. Sign in to the Azure portal.

   Note

   The portal to use is different depending on whether your Azure AD application runs in the Azure public cloud or in a national or sovereign cloud. For more information, see National clouds.

2. If you have access to multiple tenants, subscriptions, or directories, click the **Directories + subscriptions** (directory with filter) icon in the top menu to switch to the directory in which you want to provision the service principal.

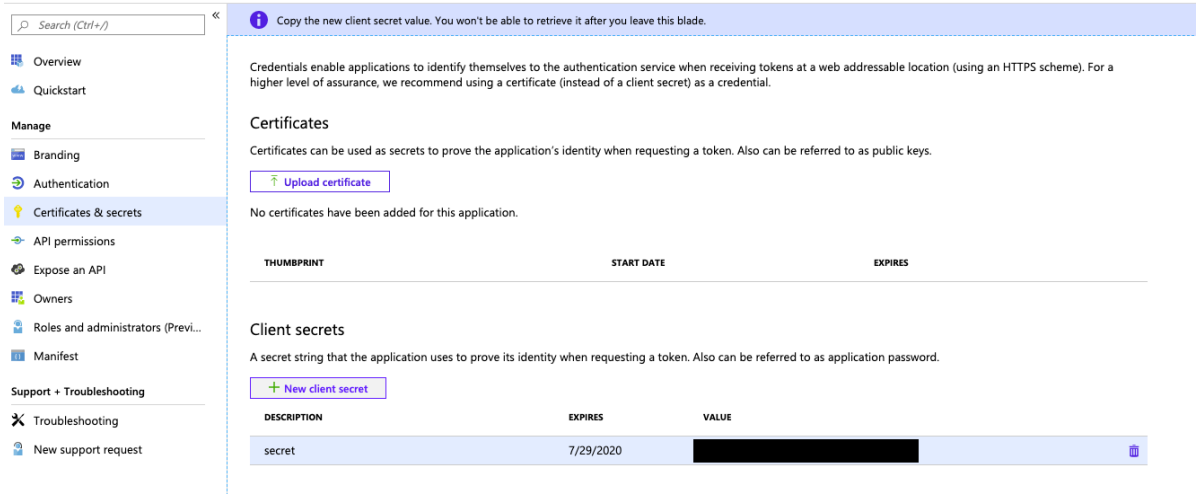
3. Search for and select **Azure Active Directory**.

4. Within **Manage**, click **App registrations > New registration**.

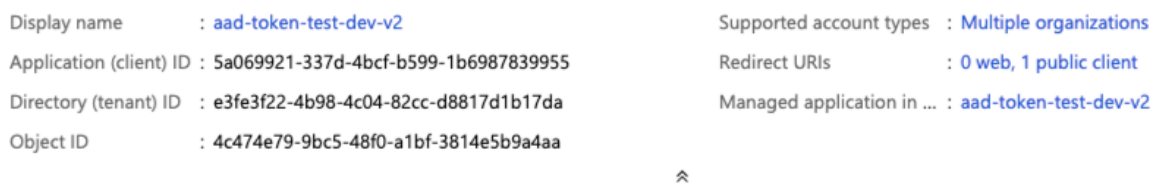
5. For **Name**, enter a name for the application.

6. In the **Supported account types** section, select **Accounts in this organizational directory only (Single tenant)**.

7. Click **Register**.
8. Within **Manage**, click **Certificates & secrets**.
9. On the **Client secrets** tab, click **New client secret**.



10. In the **Add a client secret** pane, for **Description**, enter a description for the client secret.
11. For **Expires**, select an expiry time period for the client secret, and then click **Add**.
12. Copy and store the client secret's **Value** in a secure place, as this client secret is the password for your application.
13. On the application page's **Overview** page, in the **Essentials** section, copy the following values:
    - **Application (client) ID**
    - **Directory (tenant) ID**



Provision a service principal with the Azure CLI

See Create an Azure service principal with the Azure CLI.

Get an Azure AD access token with the Microsoft identity platform REST API

To access the Databricks REST API with the service principal, you get and then use an Azure AD access token for the service principal. For more information, see First case: Access token request with a shared secret.

Tip

For more detailed, step-by-step instructions about how to get an Azure AD access token, see Service principals for Azure Databricks automation.

You can also use the Azure CLI to get the Azure AD access token. See Get an Azure AD access token with the Azure CLI.

1. Gather the following information:

Parameter	Description
Tenant ID	The **Directory (tenant) ID** for the application registered in Azure AD.
Client ID	The **Application (client) ID** for the application registered in Azure AD.
Client secret	The **Value** of the client secret for the application registered in Azure AD.

2. Use the preceding information along with curl to get the Azure AD access token.

   Bash

   ```
   curl -X POST -H 'Content-Type: application/x-www-form-urlencoded' \
   https://login.microsoftonline.com/<tenant-id>/oauth2/v2.0/token \
   -d 'client_id=<client-id>' \
   -d 'grant_type=client_credentials' \
   -d 'scope=2ff814a6-3304-4ab8-85cb-cd0e6f879c1d%2F.default' \
   -d 'client_secret=<client-secret>'
   ```

   Replace:

   - `<tenant-id>` with the registered application's tenant ID.
   - `<client-id>` with the registered application's client ID.
   - `<client-secret>` with the registered application's client secret value.

   Do not change the value of the `scope` parameter. It represents the programmatic ID for Azure Databricks (`2ff814a6-3304-4ab8-85cb-cd0e6f879c1d`) along with the default scope (`/.default` , URL-encoded as `%2f.default`).

   For example:

   Bash

   ```
   curl -X POST -H 'Content-Type: application/x-www-form-urlencoded' \
   https://login.microsoftonline.com/a1bc2d34-5e67-8f89-01ab-
   c2345d6c78de/oauth2/v2.0/token \
   -d 'client_id=12a34b56-789c-0d12-e3fa-b456789c0123' \
   -d 'grant_type=client_credentials' \
   -d 'scope=2ff814a6-3304-4ab8-85cb-cd0e6f879c1d%2F.default' \
   -d 'client_secret=abc1D~Ef...2ghIJKlM3'
   ```

   The Azure AD access token is in the `access_token` value within the output of the call.

Get an Azure AD access token with the Azure CLI

To access the Databricks REST API with the service principal, you get and then use an Azure AD access token for the service principal.

1. Gather the following information:

Parameter	Description
Tenant ID	The **Directory (tenant) ID** for the application registered in Azure AD.
Client ID	The **Application (client) ID** for the application registered in Azure AD.
Client secret	The **Value** of the client secret for the application registered in Azure AD.

2. Sign in to Azure by using the Azure CLI to run the az login command. Use the `--service-principal` option along with specifying the values for the parameters of Tenant ID (**Directory (tenant) ID**), Client ID (**Application (client) ID**), and Client secret (**Value**) for the application registered in Azure AD. Use the `--output` option to display the command's output in a different format (in this case, a table instead of JSON) for better readability.

   Bash

   ```
   az login \
   --service-principal \
   --tenant <Tenant-ID>
   --username <Client-ID> \
   --password <Client-secret> \
   --output table
   ```

3. Confirm that you are signed in to the correct subscription for which you want to create the Azure AD access token for the signed-in Azure AD service principal. To do this, in the output of the `az login` command, look for the table row where `IsDefault` is set to `True`. The subscription's `Name` indicates the current subscription. See also Get subscription and tenant IDs in the Azure portal.

   If you need to switch to a different subscription, run the az account set command. Use the `--name` or `--subscription` option to specify the desired subscription name or ID. You can get the subscription ID for an Azure Databricks workspace by clicking **Azure Portal** in the workspace navigation bar and looking for `Resource ID: /subscriptions/00000000-0000-0000-0000-000000000000`, where `00000000-0000-0000-0000-000000000000` is the subscription ID.

   Bash

   ```
   az account set --name "<subscription-name>"

   # Or ...

   az account set --subscription <subscription-id>
   ```

   If the following message displays, you are signed in to the wrong tenant: `The subscription of '<subscription-id>' doesn't exist in cloud 'AzureCloud'.` To sign in to the correct tenant, you must run the `az login` command again, using the `--tenant` option to specify the correct tenant ID. You can the tenant ID for an Azure Databricks workspace by running the command `curl -v <per-workspace-URL>/aad/auth` and looking in the output `< location: https://login.microsoftonline.com/00000000-0000-0000-0000-000000000000`, where `00000000-0000-0000-0000-000000000000` is the tenant ID. See also Get subscription and tenant IDs in the Azure portal.

   Bash

   ```
   az login --tenant <tenant-id> --output table
   ```

4. Generate the Azure AD access token for the signed-in Azure AD service principal by running the az account get-access-token command. Use the `--resource` option to specify the unique resource ID for the Azure Databricks service, which is `2ff814a6-3304-4ab8-85cb-cd0e6f879c1d`. You can display just the Azure AD token's value in the output of the command by using the `--query` and `--output` options.

   Bash

   ```
   az account get-access-token \
   --resource 2ff814a6-3304-4ab8-85cb-cd0e6f879c1d \
   --query "accessToken" \
   --output tsv
   ```

Use the service principal's Azure AD access token to access the Databricks REST API

A service principal that is a Databricks user can authenticate to the Databricks REST API with Azure Active Directory tokens.

A service principal can also add itself as an admin to a workspace if it has the Contributor or Owner role on the target workspace resource in Azure. If the service principal is a Contributor or Owner on the target workspace and you want to add it to a workspace using its Azure Active Directory token, go to Workspace-level API access for service principals that are not Azure Databricks users.

Otherwise, continue to API access for service principals that are Azure Databricks users and admins.

API access for service principals that are Azure Databricks users and admins

To complete this procedure, you must first add the service principal to the Azure Databricks account or workspace. You can add your service principal directly to your account, without granting it workspace access, by using the SCIM (Account) API.

You can add your service principal to a workspace by using the Add service principal endpoint. This will also add the service principal to your Azure Databricks account. For example:

Bash

```
curl -X POST \
-H 'Authorization: Bearer <access-token>' \
https://adb-
1234567890123456.7.azuredatabricks.net/api/2.0/preview/scim/v2/ServicePrincipals \
-H 'Content-type: application/scim+json' \
-d @create-service-principal.json
```

`create-service-principal.json`:

JSON

```
{
  "displayName": "My Service Principal",
  "applicationId": "12a34b56-789c-0d12-e3fa-b456789c0123",
  "entitlements": [
    {
      "value": "allow-cluster-create"
    }
  ],
  "schemas": [
    "urn:ietf:params:scim:schemas:core:2.0:ServicePrincipal"
  ],
  "active": true
}
```

Skip ahead to Workspace-level API access for service principals that are not Azure Databricks users if either of the following are true:

- The Azure Databricks REST API that you want to call requires workspace admin access and the service principal is a member of the workspace, but does not currently have admin access to the workspace.
- The service principal has not already been added to the target Azure Databricks workspace.

1. Gather the following information.

Parameter	Description
Azure AD access token	The Azure AD access token returned from the request in Get an Azure AD access token with the Microsoft identity platform REST API or Get an Azure AD access token with the Azure CLI.

2. Use the Azure AD access token along with `curl` to call the Databricks REST API. For example:

   Bash

   ```
   curl -X GET \
   -H 'Authorization: Bearer <access-token>' \
   https://<databricks-instance>/api/2.0/clusters/list
   ```

   Replace:

   - `<access-token>` with the Azure AD access token.
   - `<databricks-instance>` with the per-workspace URL of your Azure Databricks deployment.
   - `GET` and `/api/2.0/clusters/list` with the appropriate HTTP operation and endpoint for the target Databricks REST API.

   For example:

   Bash

   ```
   curl -X GET \
   -H 'Authorization: Bearer <access-token>' \
   https://adb-1234567890123456.7.azuredatabricks.net/api/2.0/clusters/list
   ```

Workspace-level API access for service principals that are not Azure Databricks users

Follow this procedure if any of the following are true:

- The Azure Databricks REST API that you want to call requires workspace admin access and the service principal is a member of the workspace, but does not currently have admin access to the workspace.
- The service principal has not already been added to the target Azure Databricks workspace.

Requirements:

The service principal requires the Contributor or Owner role on the target workspace resource in Azure.

1. Gather the following information:

Parameter	Description
Tenant ID	The **Directory (tenant) ID** for the application registered in Azure AD in Provision a service principal in Azure portal.
Client ID	The **Application (client) ID** for the application registered in Azure AD.
Client secret	The **Value** of the client secret for the application registered in Azure AD, which you created in Provision a service principal in Azure portal.
Azure AD access token	The Azure AD access token returned from the request in Get an Azure AD access token with the Microsoft identity platform REST API or Get an Azure AD access token with the Azure CLI.
Subscription ID	The ID (not the name) of the Azure subscription that is associated with the target Azure Databricks workspace. To get to this and the following information, see Open resources. To open the target resource, you can search on the **Azure Databricks** service type and any other information in Azure that you know about the target Azure Databricks workspace.
Resource group name	The name of the Azure resource group that is associated with the target Azure Databricks workspace.
Workspace name	The name in Azure of the target Azure Databricks workspace.

2. Use some of the preceding information along with `curl` to get an Azure AD management endpoint access token.

   Bash

   ```
   curl -X POST -H 'Content-Type: application/x-www-form-urlencoded' \
   https://login.microsoftonline.com/<tenant-id>/oauth2/token \
   -d 'client_id=<client-id>' \
   -d 'grant_type=client_credentials' \
   -d 'resource=https%3A%2F%2Fmanagement.core.windows.net%2F' \
   -d 'client_secret=<client-secret>'
   ```

   Replace:

   - `<tenant-id>` with the registered application's tenant ID.
   - `<client-id>` with the registered application's client ID.
   - `<client-secret>` with the registered application's client secret value.

   Do not change the value of the `resource` parameter. It represents the Azure AD management endpoint (`https://management.core.windows.net/` , URL-encoded as `https%3A%2F%2Fmanagement.core.windows.net%2F`).

   For example:

   Bash

   ```
   curl -X POST -H 'Content-Type: application/x-www-form-urlencoded' \
   https://login.microsoftonline.com/a1bc2d34-5e67-8f89-01ab-
   c2345d6c78de/oauth2/token \
   -d 'client_id=12a34b56-789c-0d12-e3fa-b456789c0123' \
   -d 'grant_type=client_credentials' \
   -d 'resource=https%3A%2F%2Fmanagement.core.windows.net%2F' \
   -d 'client_secret=abc1D~Ef...2ghIJKlM3'
   ```

   The Azure AD management endpoint access token is in the `access_token` value within the output of the call.

3. Use the Azure AD management endpoint access token along with the rest of the preceding information and `curl` to call the Databricks REST API, for example:

```
 curl -X GET \
 -H 'Authorization: Bearer <access-token>' \
 -H 'X-Databricks-Azure-SP-Management-Token: <management-access-token>' \
 -H 'X-Databricks-Azure-Workspace-Resource-Id: /subscriptions/<subscription-
id>/resourceGroups/<resource-group-
name>/providers/Microsoft.Databricks/workspaces/<workspace-name>' \
 https://<databricks-instance>/api/2.0/clusters/list
```

   Replace:

   - `<access-token>` with the Azure AD access token.
   - `<management-access-token>` with the Azure AD management endpoint access token.
   - `<subscription-id>` with the ID of the subscription that is associated with the target Azure Databricks workspace.
   - `<resource-group-name>` with the name of the resource group that is associated with the target Azure Databricks workspace.
   - `<workspace-name>` with the name of the target Azure Databricks workspace.
   - `<databricks-instance>` with the per-workspace URL of your Azure Databricks deployment.
   - `GET` and `/api/2.0/clusters/list` with the appropriate HTTP operation and endpoint for the target Databricks REST API.

   For example:

   Bash

```
curl -X GET \
-H 'Authorization:Bearer <access-token>' \
-H 'X-Databricks-Azure-SP-Management-Token: abC1dE...ghIj23kl' \
-H 'X-Databricks-Azure-Workspace-Resource-Id:
/subscriptions/12a345...bcd6789e/resourceGroups/my-resource-
group/providers/Microsoft.Databricks/workspaces/my-workspace' \
https://adb-1234567890123456.7.azuredatabricks.net/api/2.0/clusters/list
```

   Upon authenticating to the workspace, the service principal becomes an Azure Databricks workspace admin and no longer needs the Contributor or Owner role to access the workspace.