Quickstart: Configure an application to expose web APIs

08/14/2019 • 5 minutes to read •

In this article

You can develop a web API and make it available to client applications by exposing permissions/scopes and roles. A correctly configured web API is made available just like the other Microsoft web APIs, including the Graph API and the Office 365 APIs.

In this quickstart, you'll learn how to configure an application to expose a new scope to make it available to client applications.

Prerequisites

To get started, make sure you complete these prerequisites:

- Learn about the supported permissions and consent, which is important to understand when building applications that need to be used by other users or applications.
- Have a tenant that has applications registered to it.
  - If you don't have apps registered, learn how to register applications with the Microsoft identity platform.

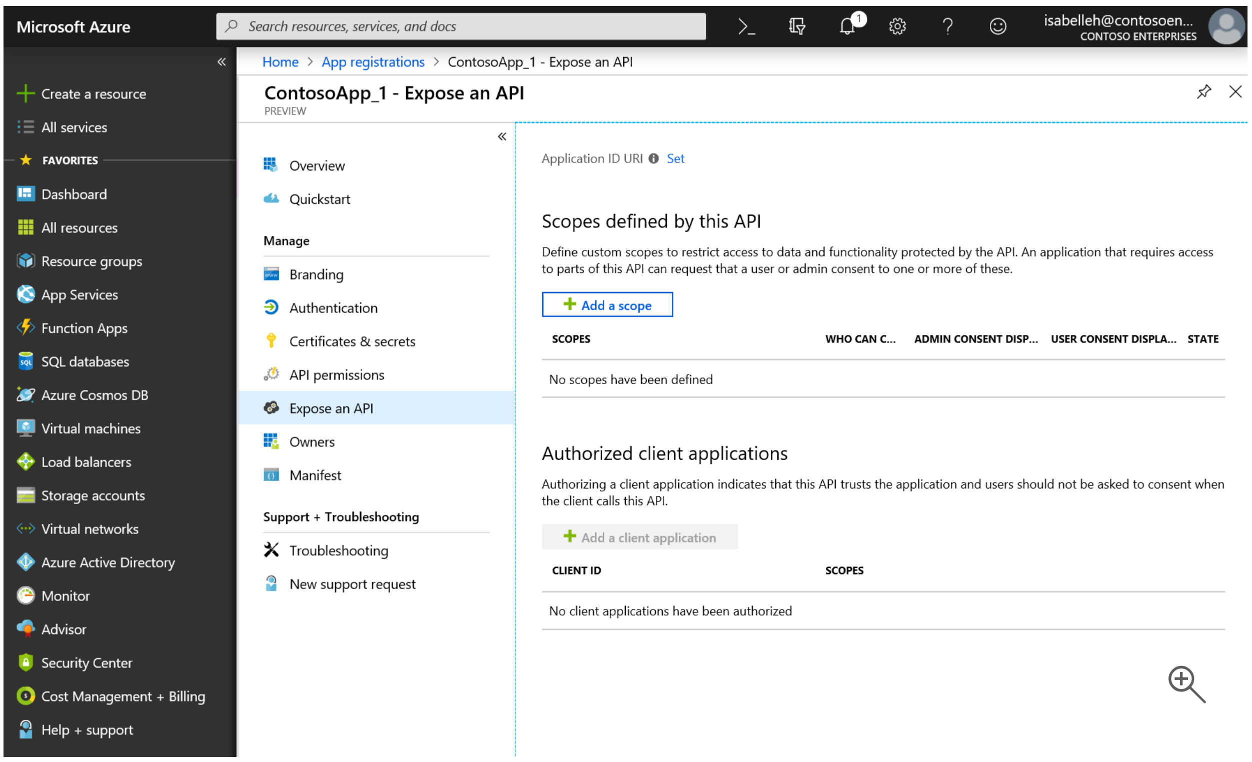
Sign in to the Azure portal and select the app

Before you can configure the app, follow these steps:

1. Sign in to the Azure portal using either a work or school account or a personal Microsoft account.
2. If your account gives you access to more than one tenant, select your account in the top right corner, and set your portal session to the desired Azure AD tenant.
3. In the left-hand navigation pane, select the **Azure Active Directory** service and then select **App registrations**.
4. Find and select the application you want to configure. Once you've selected the app, you'll see the application's **Overview** or main registration page.
5. Choose which method you want to use, UI or application manifest, to expose a new scope:

    - Expose a new scope through the UI
    - Expose a new scope or role through the application manifest

Expose a new scope through the UI



Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API. An application that requires access to parts of this API can request that a user or admin consent to one or more of these.

SCOPES	WHO CAN C...	ADMIN CONSENT DISP...	USER CONSENT DISPLA...	STATE
No scopes have been defined				

Authorized client applications

Authorizing a client application indicates that this API trusts the application and users should not be asked to consent when the client calls this API.

CLIENT ID	SCOPES
No client applications have been authorized	

To expose a new scope through the UI:

1. From the app's **Overview** page, select the **Expose an API** section.

2. Select **Add a scope**.

3. If you have not set an **Application ID URI**, you will see a prompt to enter one. Enter your application ID URI or use the one provided and then select **Save and continue**.

4. When the **Add a scope** page appears, enter your scope's information:

Field	Description
Scope name	Enter a meaningful name for your scope. For example, `Employees.Read.All`.
Who can consent	Select whether this scope can be consented to by users, or if admin consent is required. Select **Admins only** for higher-privileged permissions.
Admin consent display name	Enter a meaningful description for your scope, which admins will see. For example, `Read-only access to Employee records`
Admin consent description	Enter a meaningful description for your scope, which admins will see. For example, `Allow the application to have read-only access to all Employee data.`

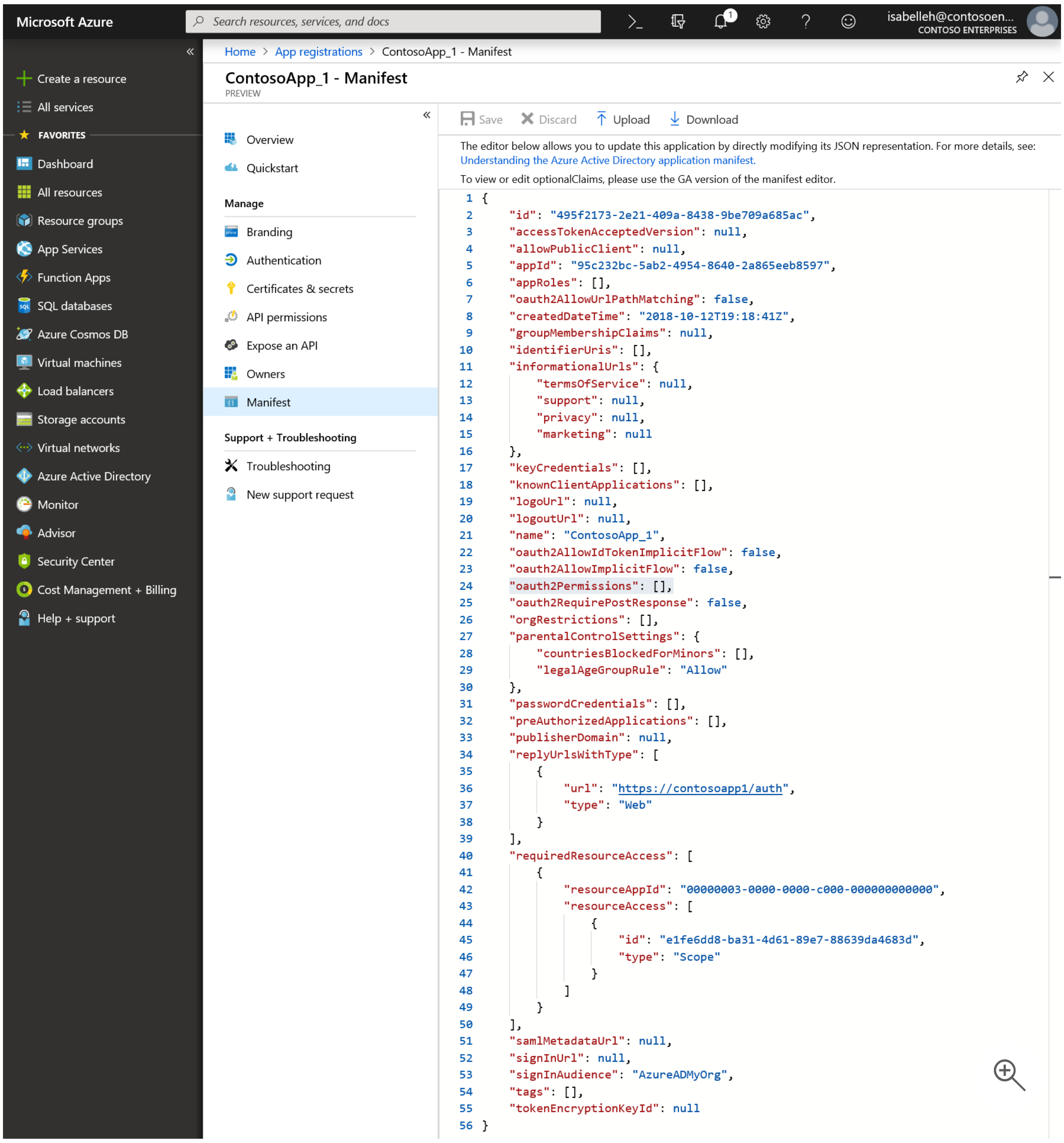
If users can consent to your scope, also add values for the following fields:

Field	Description
User consent display name	Enter a meaningful name for your scope, which users will see. For example, `Read-only access to your Employee records`
User consent description	Enter a meaningful description for your scope, which users will see. For example, `Allow the application to have read-only access to your Employee data.`

5. Set the **State** and select **Add scope** when you're done.

6. Follow the steps to verify that the web API is exposed to other applications.

Expose a new scope or role through the application manifest



The editor below allows you to update this application by directly modifying its JSON representation. For more details, see: Understanding the Azure Active Directory application manifest.

To view or edit optionalClaims, please use the GA version of the manifest editor.

```json
{
    "id": "495f2173-2e21-409a-8438-9be709a685ac",
    "accessTokenAcceptedVersion": null,
    "allowPublicClient": null,
    "appId": "95c232bc-5ab2-4954-8640-2a865eeb8597",
    "appRoles": [],
    "oauth2AllowUrlPathMatching": false,
    "createdDateTime": "2018-10-12T19:18:41Z",
    "groupMembershipClaims": null,
    "identifierUris": [],
    "informationalUrls": {
        "termsOfService": null,
        "support": null,
        "privacy": null,
        "marketing": null
    },
    "keyCredentials": [],
    "knownClientApplications": [],
    "logoUrl": null,
    "logoutUrl": null,
    "name": "ContosoApp_1",
    "oauth2AllowIdTokenImplicitFlow": false,
    "oauth2AllowImplicitFlow": false,
    "oauth2Permissions": [],
    "oauth2RequirePostResponse": false,
    "orgRestrictions": [],
    "parentalControlSettings": {
        "countriesBlockedForMinors": [],
        "legalAgeGroupRule": "Allow"
    },
    "passwordCredentials": [],
    "preAuthorizedApplications": [],
    "publisherDomain": null,
    "replyUrlsWithType": [
        {
            "url": "https://contosoapp1/auth",
            "type": "Web"
        }
    ],
    "requiredResourceAccess": [
        {
            "resourceAppId": "00000003-0000-0000-c000-000000000000",
            "resourceAccess": [
                {
                    "id": "e1fe6dd8-ba31-4d61-89e7-88639da4683d",
                    "type": "Scope"
                }
            ]
        }
    ],
    "samlMetadataUrl": null,
    "signInUrl": null,
    "signInAudience": "AzureADMyOrg",
    "tags": [],
    "tokenEncryptionKeyId": null
}
```

To expose a new scope through the application manifest:

1. From the app's **Overview** page, select the **Manifest** section. A web-based manifest editor opens, allowing you to **Edit** the manifest within the portal. Optionally, you can select **Download** and edit the manifest locally, and then use **Upload** to reapply it to your application.

   The following example shows how to expose a new scope called `Employees.Read.All` in the resource/API by adding the following JSON element to the `oauth2Permissions` collection.

   JSON Copy

   ```json
   {
     "adminConsentDescription": "Allow the application to have read-only access to all Employee data.",
   ```

```
    "adminConsentDisplayName": "Read-only access to Employee records",
    "id": "2b351394-d7a7-4a84-841e-08a6a17e4cb8",
    "isEnabled": true,
    "type": "User",
    "userConsentDescription": "Allow the application to have read-only
  access to your Employee data.",
    "userConsentDisplayName": "Read-only access to your Employee
  records",
    "value": "Employees.Read.All"
  }
```

> **Note**
>
> The `id` value must be generated programmatically or by using a GUID generation tool such as **guidgen**. The `id` represents a unique identifier for the scope as exposed by the web API. Once a client is appropriately configured with permissions to access your web API, it is issued an OAuth 2.0 access token by Azure AD. When the client calls the web API, it presents the access token that has the scope (scp) claim set to the permissions requested in its application registration.
>
> You can expose additional scopes later as necessary. Consider that your web API might expose multiple scopes associated with a variety of different functions. Your resource can control access to the web API at runtime by evaluating the scope (`scp`) claim(s) in the received OAuth 2.0 access token.

2. When finished, click **Save**. Now your web API is configured for use by other applications in your directory.

3. Follow the steps to verify that the web API is exposed to other applications.

Verify the web API is exposed to other applications

1. Go back to your Azure AD tenant, select **App registrations**, find and select the client application you want to configure.
2. Repeat the steps outlined in Configure a client application to access web APIs.
3. When you get to the step to select an API, select your resource. You should see the new scope, available for client permission requests.

More on the application manifest

The application manifest serves as a mechanism for updating the application entity, which defines all attributes of an Azure AD application's identity configuration. For more information on the Application entity and its schema, see the Graph API Application entity documentation. The article contains complete reference information on the Application entity members used to specify permissions for your API, including:

- The appRoles member, which is a collection of AppRole entities, used to define application permissions for a web API.
- The oauth2Permissions member, which is a collection of OAuth2Permission entities, used to define delegated permissions for a web API.

For more information on application manifest concepts in general, see Understanding the Azure Active Directory application manifest.

Next steps

Learn about these other related app management quickstarts for apps:

- Register an application with the Microsoft identity platform
- Configure a client application to access web APIs
- Modify the accounts supported by an application
- Remove an application registered with the Microsoft identity platform

To learn more about the two Azure AD objects that represent a registered application and the relationship between them, see Application objects and service principal objects.

To learn more about the branding guidelines you should use when developing applications with Azure Active Directory, see Branding guidelines for applications.