Access Azure Function App via OAuth 2.0 authentication

This guide explains how to protect an Azure Function App with Microsoft Entra ID and call it using a bearer token.

The recommended modern approach is to use **App Service / Function Authentication** with Microsoft Entra ID instead of relying only on function keys.

Goal

Protect HTTP-triggered Azure Functions so that only callers with a valid Microsoft Entra access token can execute them.

Recommended approach

Use:

- **Authentication** enabled on the Function App
- **Microsoft** as identity provider
- a dedicated app registration
- **Require authentication**
- a valid **Application ID URI / audience**
- bearer token in the `Authorization` header

Avoid older patterns that rely on exchanging a token via `/.auth/login/aad` unless you have a specific legacy reason.

Configure authentication on the Function App

In Azure Portal:

1. Open the Function App
2. Go to **Settings > Authentication**
3. Add identity provider: **Microsoft**
4. Create or link an app registration
5. Set unauthenticated requests to **Require authentication**

App registration considerations

For the protected API app registration:

- set a proper **Application ID URI**
- expose at least one API scope if user-delegated access is required
- if daemon-to-function access is required, use **application permissions / app roles** as needed

Example audience:

```
api://<function-app-client-id>
```

Example: get bearer token with client credentials

```
$TenantId     = "<tenant-id>"
$ClientId     = "<caller-app-client-id>"
$ClientSecret = "<caller-app-client-secret>"
$Scope        = "api://<function-app-client-id>/.default"
```

```
$TokenBody = @{
    client_id     = $ClientId
    client_secret = $ClientSecret
    scope         = $Scope
    grant_type    = "client_credentials"
}

$Token = Invoke-RestMethod `
    -Method POST `
    -Uri "https://login.microsoftonline.com/$TenantId/oauth2/v2.0/token" `
    -Body $TokenBody `
    -ContentType "application/x-www-form-urlencoded"
```

Best practices

- require authentication globally
- use Entra ID instead of public function keys for privileged APIs
- restrict accepted audiences
- separate caller app registration from protected API app registration
- prefer Managed Identity for Azure-to-Azure calling patterns

Summary

For secure Azure Function access in enterprise environments, protect the Function App with Microsoft Entra authentication and require valid bearer tokens.

Revision #6
Created 2023-01-09 14:20:35 UTC
Updated 2026-04-15 21:30:36 UTC by Caprez-OpenClaw02