Azure AD Access for SharePoint Online

Applications defined in Azure AD are allowed to make app-only calls by sharing a certificate with Azure AD. Azure AD will get the public key certificate and the app will get the private key certificate. Although a trusted certificate should be used for production deployments, makecert/self-signed certificates are fine for testing/debugging (similar to local web debugging with https). Here are the steps to generate a self-signed certificate with makecert.exe and exporting it for use with Azure AD.

Part 1: Generate a Self-signed Certificate

1. Open Visual Studio Tools Command Prompt.

   > **Note:** for Windows 10 you may have to download the Windows 10 SDK to get the makecert application.

2. Run makecert.exe with the following syntax:

```
makecert -r -pe -n "CN=SearchTechnologies SPOnline Cert" -b 10/15/2016 -e 10/15/2018 -ss my -len 2048
```

3. Run mmc.exe
4. Go to File Add/Remove Snap In
5. Add Certificates My User Account
6. Locate the certificate from step 2 in the Personal certificate store
7. Right-click and select All tasks >> Export
8. Complete the Certificate Export Wizard twice: once with the private key (specify a password and save as .pfx) and once without the private key (save as .cer)

Part 2: Prepare the certificate public key for Azure AD

1. Open Windows PowerShell and run the following commands:

```
$certPath = <Path to Cert>
$cert = New-Object System.Security.Cryptography.X509Certificates.X509Certificate2
$cert.Import($certPath)
$rawCert = $cert.GetRawCertData()
$base64Cert = [System.Convert]::ToBase64String($rawCert)
$rawCertHash = $cert.GetCertHash()
$base64CertHash = [System.Convert]::ToBase64String($rawCertHash)
$KeyId = [System.Guid]::NewGuid().ToString()
Write-Host $base64Cert
Write-Host $base64CertHash
Write-Host $KeyId
```

2. Copy the values output for $base64Cert, $base64CertHash, and $KeyId for Part 4

Part 3: Create the Azure AD Application

1. Log into the Azure Management Portal for your Office 365 tenant.
2. Go to the Azure Active Directory tab and select App Registrations.
3. Select "New Application Registration".
4. Give the application a name and keep the default selection of "Web App / API".
5. Enter a Sign-on URL (the value of this doesn't really matter other than being unique) and click "Create".
6. Look for your new application on the Regitered Applications list and click it.
7. Go to Required Permissions and click on "Add".
8. On the "Select an API" section, add the "Office 365 SharePoint Online" application
9. On "Select Permissions", select the following "Application Permissions":
   a. Read Managed Metadata.
   b. Have Full Control of all Site Collections.
   c. Read Items in all Site Collections.
10. After saving you have to click "Grant Permissions" to apply the changes.

> On the Configure section you'll also see the Application ID. Copy and save this ID, you are going to need it when configuring the connector.

Part 4: Configure certificate public key for App

1. Click the Manage Manifest button at the top of the Registered App Properties.
2. Update the keyCredentials attribute with the following settings:

```
"keyCredentials": [
        {
                "customKeyIdentifier": "<$base64CertHash FROM ABOVE>",
                "keyId": "<$KeyId FROM ABOVE>",
                "type": "AsymmetricX509Cert",
                "usage": "Verify",
                "value": "<$base64Cert FROM ABOVE>"
        }
],
```

3. Save the updated manifest.

> **Note:** If you try to download the manifest again, you'll notice that the expiration dates are now there and the cert value is now null. This is normal and it shouldn't prevent the app to work as expected.

4. Everything should now be setup in Azure AD for the app to run in the background and get app-only access tokens from Azure AD.

Part 5: Generate Private Key

> You may need to download OpenSSL for Windows to follow these steps.

1. Extract pem key

```
openssl pkcs12 -nocerts -in <PFX Path> -out <PEM Path>
```

2. Convert extracted pem key to der format

```
openssl pkcs8 -topk8 -inform PEM -outform DER -in <PEM Path> -out <DER Path> -nocrypt
```