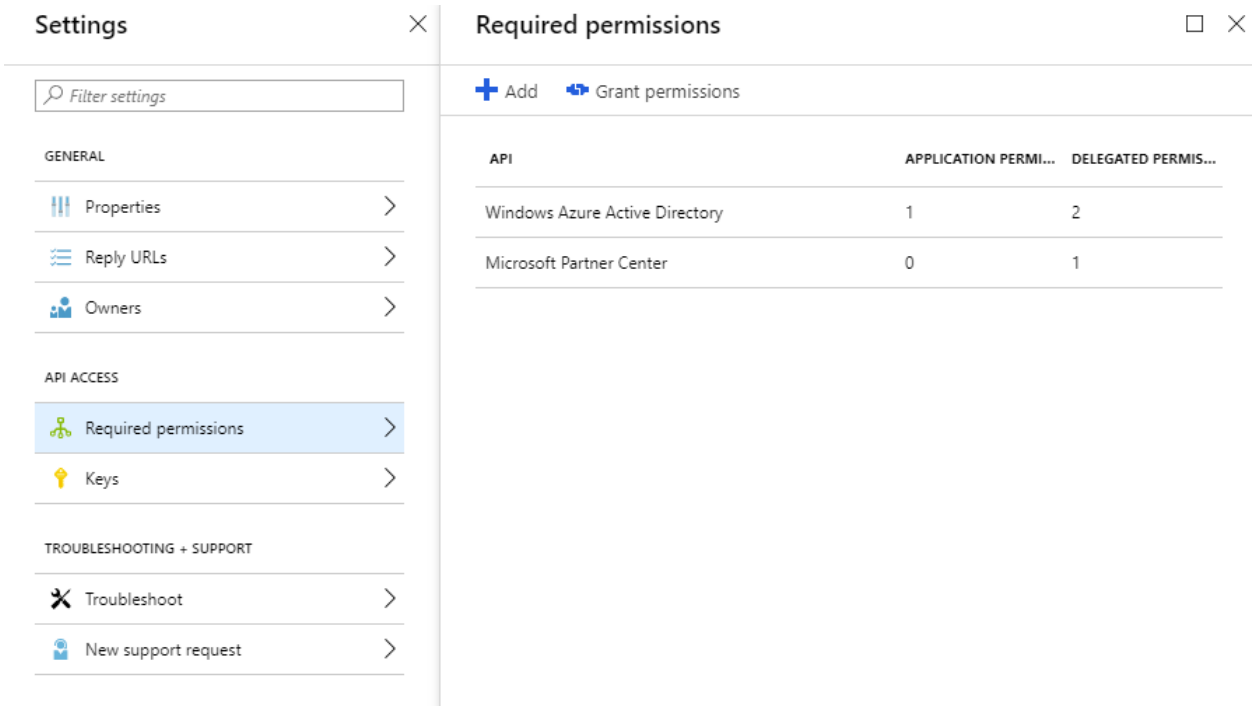
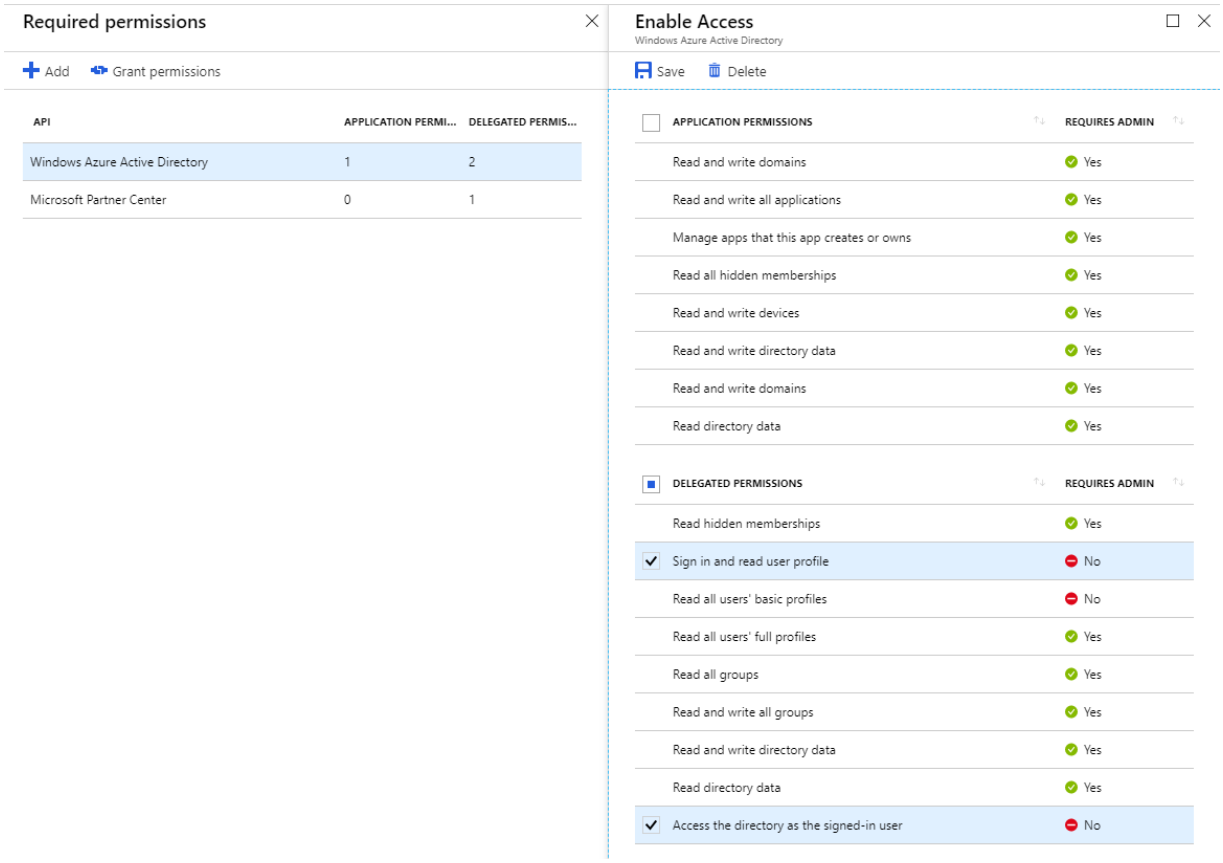
AUTHORIZATION CODE FLOW USING REST API CALLS

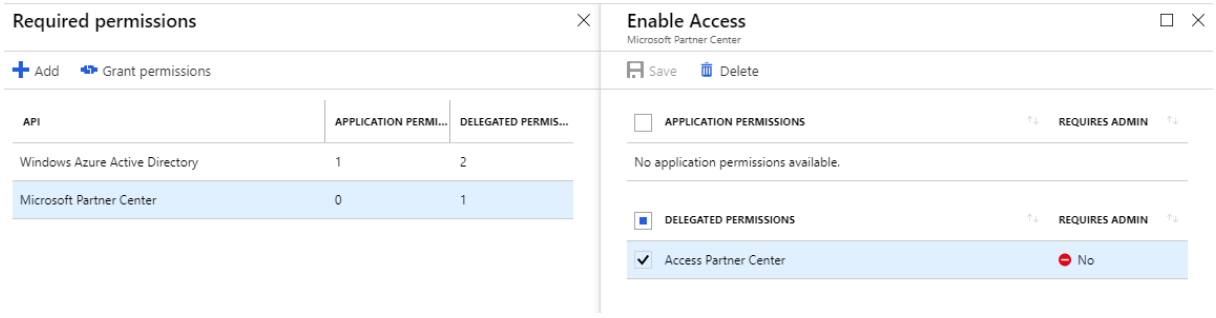
AZURE ACTIVE DIRECTORY APPLICATION PREREQUISITES:

A Web application must be created and registered in partner center.

1. Go to https://portal.azure.com and go to “Azure Active Directory” to create a web application
2. Give delegated application permissions to “Microsoft Partner Center” (some tenant show this as SampleBECApp)
3. Give delegated application permissions to “Azure Management APIs” if you are planning to call Azure APIs.
4. Give delegated application permissions to “Windows Azure Active Directory”
5. Make sure home URL of the application set to endpoint where a live web application is running to accept authorization code from the login call.
    a. Sample uses: https://localhost:44395/
6. Capture the following settings from Web application definition on active directory page
    a. Application Id
    b. Application Secret: You can create an application key using azure portal. We recommend you use certificate as secret. Documentation: https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-certificate-credentials
    c. the sample below uses “application key”.
7. Below screenshots explain sample configuration to be done.







WEB APPLICATION TO CAPTURE THE CONSENT

Pre-requisites:

A web endpoint must be running to accept the authorization code from Azure Active Directory (AAD) login call. For example, in the sample below, a web application is running at endpoint: https://localhost:44395/

GETTING CONSENT

STEP1: GET AUTHORIZATION CODE

1. Use the following AAD login link to login with the user account (Admin agent/sales agent). This account is the user account that you plan to use for making Partner Center API calls.
    a. Link: https://login.microsoftonline.com/common/oauth2/authorize?client_id=Application-Id&response_mode=form_post&response_type=code%20id_token&scope=openid%20profile

b. Replace Application-Id with AAD application id (GUID)

2. The above link will prompt for user login.
3. Login with the user account (the user account should have MFA configured). AAD will prompt for additional information, either phone or email to confirm multi factor authentication.
4. Once login complete, the browser will redirect the call to web application endpoint with authorization code (in this case to https://localhost:44395/)
5. Authorization code call trace:

```
POST https://localhost:44395/ HTTP/1.1
Origin: https://login.microsoftonline.com
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: https://login.microsoftonline.com/kmsi
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie:
OpenIdConnect.nonce.hOMjjrivcxzuI4YqAw4uYC%2F%2BILFk4%2FCx3kHTHP3lBvA%3D=dHVyRXdlbk9WVUZFdlFO
NVdiY01nNEpUc0JRR0RiYWFLTHhQYlRGNl9VeXJqNjdLTGV3cFpIWFg1YmpnWVdQUURtN0dvMkdHS2kzTm02NGdQS09ve
VNEbTZJMDk1TVVNYkczYmstQmlKUzFQaTBFMEdhNVJGVHlES2d3WGlCSlVlN1c2UE9sd2kzckNrVGN2RFNULWdHY2JET3
RDQUxSaXRfLXZQdGO0RnlUM0E1TUo1YWNKOWxvQXRwSkhRYklQbmZUV3d3eHVfNEpMUUthMFlQUFgzSO1RS2NvMXYtbnV
4UVJOYkl4TTNOcw%3D%3D

code=AuthorizationCodeValue&id_token=IdTokenValue&<rest of properties for state>
```

STEP2: EXCHANGE AUTHORIZATION CODE TO GET REFRESHTOKEN

1. A POST call to AAD login endpoint https://login.microsoftonline.com/CSPTenantID/oauth2/token with the authorization code will return a refresh token
2. The sample call below makes a call in context of CSP tenant with authorization code.

   Placeholder Request:

   ```
   POST https://login.microsoftonline.com/CSPTenantID/oauth2/token HTTP/1.1
   Content-Type: application/x-www-form-urlencoded
   Host: login.microsoftonline.com
   Content-Length: 966
   Expect: 100-continue

   Body:
   resource=https%3a%2f%2fapi.partnercenter.microsoft.com&client_id=Application-
   Id&client_secret=Application-
   Secret&grant_type=authorization_code&code=AuthorizationCodeValue
   ```

   Placeholder Response:

   ```
   HTTP/1.1 200 OK
   Cache-Control: no-cache, no-store
   Content-Type: application/json; charset=utf-8

   Body:
   {"token_type":"Bearer","scope":"user_impersonation","expires_in":"3599","ext_expires_in":
   "3599","expires_on":"1547579127","not_before":"1547575227","resource":"https://api.partne
   rcenter.microsoft.com","access_token":"Access
   tokenValue","refresh_token":"RefreshTokenVlaue"}
   ```

3. Store refresh token in key vault:
   a. Key Vault APIs are documented here: https://docs.microsoft.com/en-us/rest/api/keyvault/

b. The refresh token must be stored as a secret. API reference: https://docs.microsoft.com/en-us/rest/api/keyvault/setsecret/setsecret

STEP 3: USE THE REFRESH TOKEN TO GET ACCESS TOKEN.

1. Access token should be used to call Partner Center APIs. Access token generally has very limited lifetime like an hour or less.
2. Access token must be acquired before making calls to APIs. The following rest calls explain how refresh token can be used to acquire access token.

   Placeholder Request:

```
POST https://login.microsoftonline.com/ CSPTenantID/oauth2/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Host: login.microsoftonline.com
Content-Length: 1212
Expect: 100-continue

Body:
resource=https%3a%2f%2fapi.partnercenter.microsoft.com&client_id=Application-Id
&client_secret= Application-
Secret&grant_type=refresh_token&refresh_token=RefreshTokenVlaue&scope=openid
```

   Placeholder Response:

```
HTTP/1.1 200 OK
Cache-Control: no-cache, no-store
Content-Type: application/json; charset=utf-8

{"token_type":"Bearer","scope":"user_impersonation","expires_in":"3600","ext_expires_in":
"3600","expires_on":"1547581389","not_before":"1547577489","resource":"https://api.partne
rcenter.microsoft.com","access_token":"AccessTokenValue","id_token":"IDTokenValue"}
```

USING ACCESS TOKEN TO MAKE PARTNER CENTER API CALLS

Sample call:

```
GET https://api.partnercenter.microsoft.com/v1/customers/CustomerTenantId/users HTTP/1.1
Authorization: Bearer AccessTokenValue
Accept: application/json
X-Locale: en-US
Host: api.partnercenter.microsoft.com
```

APPENDIX A: USING THE PARTNER CENTER POWERSHELL MODULE

The Partner Center PowerShell module can be utilized to reduce the required infrastructure to exchange an authorization code for an access token. Through use of this module you will be able to eliminate steps 1 and 2. These steps can be replaced with the following process

1. Install the Azure AD and Partner Center PowerShell modules. The following commands show how this is accomplished

Microsoft PowerShell commands

Enter each cmdlet on a single line, even though they may appear word-wrapped across several lines here because of formatting constraints.

```
Install-Module AzureAD
Install-Module PartnerCenter
```

2. Add *urn:ietf:wg:oauth:2.0:oob* as a reply URL for the Azure AD application. You will need to use PowerShell to perform this operation. The following commands show how this is accomplished

Microsoft PowerShell commands

Enter each cmdlet on a single line, even though they may appear word-wrapped across several lines here because of formatting constraints.

```
Connect-AzureAD

Set-AzureADApplication -ObjectId 659dd68d-3414-4254-a48b-c081b5631b86 -ReplyUrls
@("urn:ietf:wg:oauth:2.0:oob")
```

Be sure to replace the value for the object identifier parameter with the object identifier for you Azure AD application. This value can be found in the Azure management portal.

3. Through use of the New-PartnerAccessToken command you can perform the consent process and capture the required refresh token. The following commands show how this is accomplished

Microsoft PowerShell commands

Enter each cmdlet on a single line, even though they may appear word-wrapped across several lines here because of formatting constraints.

```
$credential = Get-Credential
$token = New-PartnerAccessToken -Consent -Credential $credential -Resource
https://api.partnercenter.microsoft.com -ServicePrincipal

# Copy the refresh token value to the clipboard.
$token.RefreshToken | clip
```

When the *Get-Credential* command is invoked you will be prompted to enter a username and password. Specify the application identifier as the username and the application secret as the password. When the New-PartnerAccessToken command is invoked you will be prompted for credentials once again. This time you will need to specify the credentials for the service account that you will be using. Please note that this should be a partner account with the appropriate permissions. After the successfully execution of the command you will find that the *$token* variable contains the response from Azure Active Directory for a token. Included in this response is a refresh token, you will want to store this value in a secure repository such as Azure Key Vault.

The ServicePrincipal parameter is being used with the New-PartnerAccessToken command because an Azure AD application of type web/API should be used. This type of application requires that a client identifier and secret be included in the request for an access token.

Please see Partner Center PowerShell | Secure Application Model for more details regarding this process.