Script to Pull Azure Active Directory B2B users into On-Premises Active Directory (Preview)

You can use this script sample to pull Azure Active Directory (Azure AD) B2B collaboration users into your on-premises Active Directory.

This script is currently a preview release. Before using, please review the Supplemental Terms of Use for Microsoft Azure Previews.

Contents

Environment prerequisites

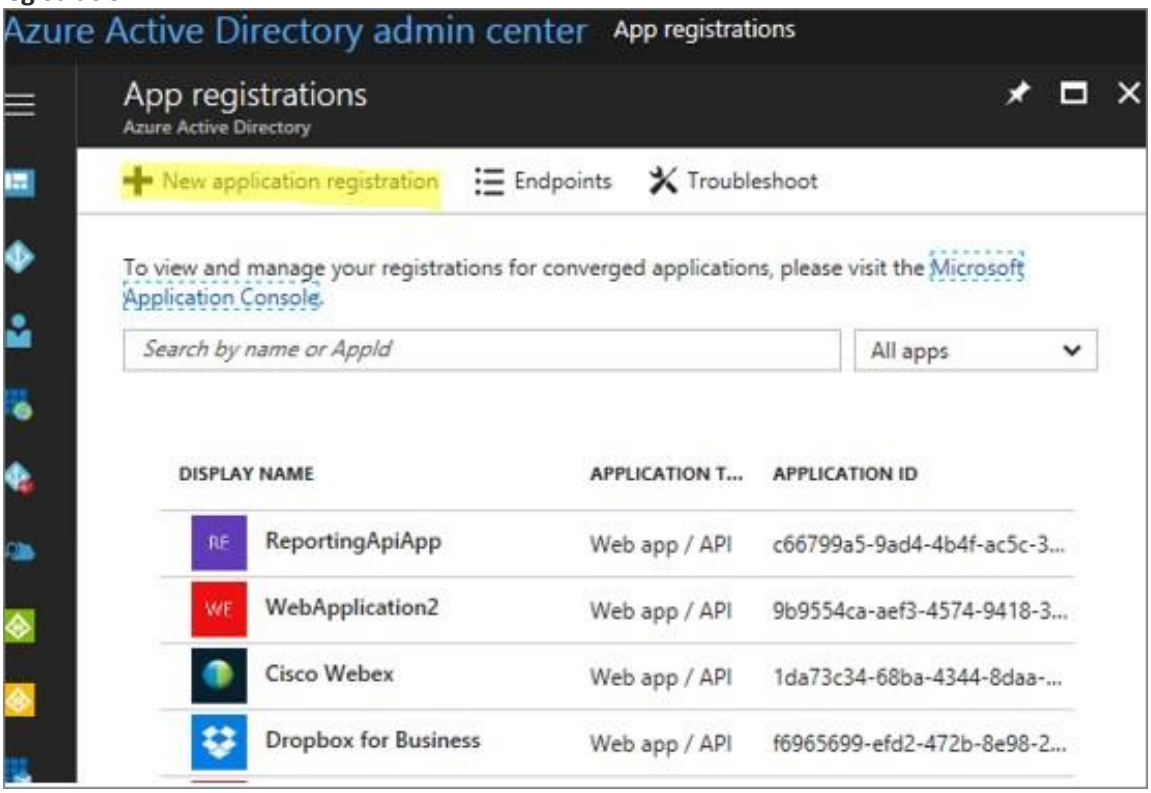
To use this script, you must have certain components already set up in your environment. Make sure that:

- Azure AD Application Proxy is installed and configured. See Get started with Application Proxy and install the connector.
- One or more on-premises apps are published to the cloud as enterprise apps. See Publish applications using Azure AD Application Proxy.
- The apps are configured to user Kerberos constrained delegation. See Kerberos Constrained Delegation for single sign-on to your apps with Application Proxy.
- Business-to-business (B2B) guest user accounts exist in Azure AD. These users are assigned to the enterprise apps through user or group membership. See How do Azure Active Directory admins add B2B collaboration users?

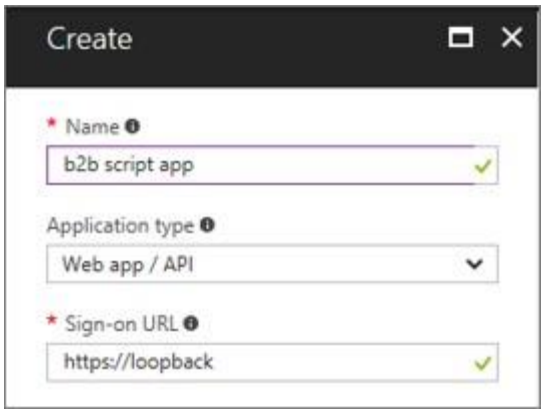
Script prerequisites

Create a new Azure AD application registration

1. Sign in to Azure AD Management Portal: App Registrations, and click **+ New application registration**.



2. Enter a name for the app (for example, *Pull B2B users on-prem*), select **Web app / API**, and then enter the sign-on URL. You can enter https://loopback as a placeholder. Click **Create**.

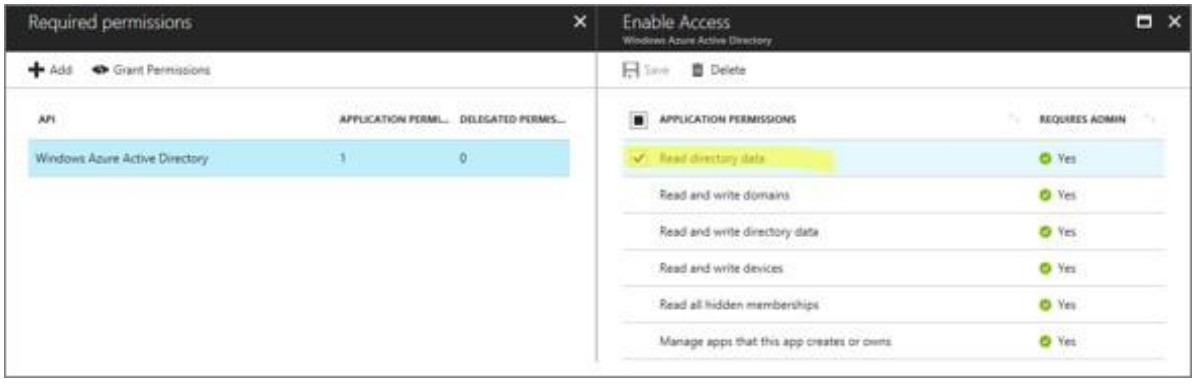


3. Under **App registrations**, make sure that **All apps** is selected in the search filter. From the application list, find the app you just created and click to open and edit it.

4. Click **Settings**, and then click **Required permissions**. Under API, click **Windows Azure Active Directory**.

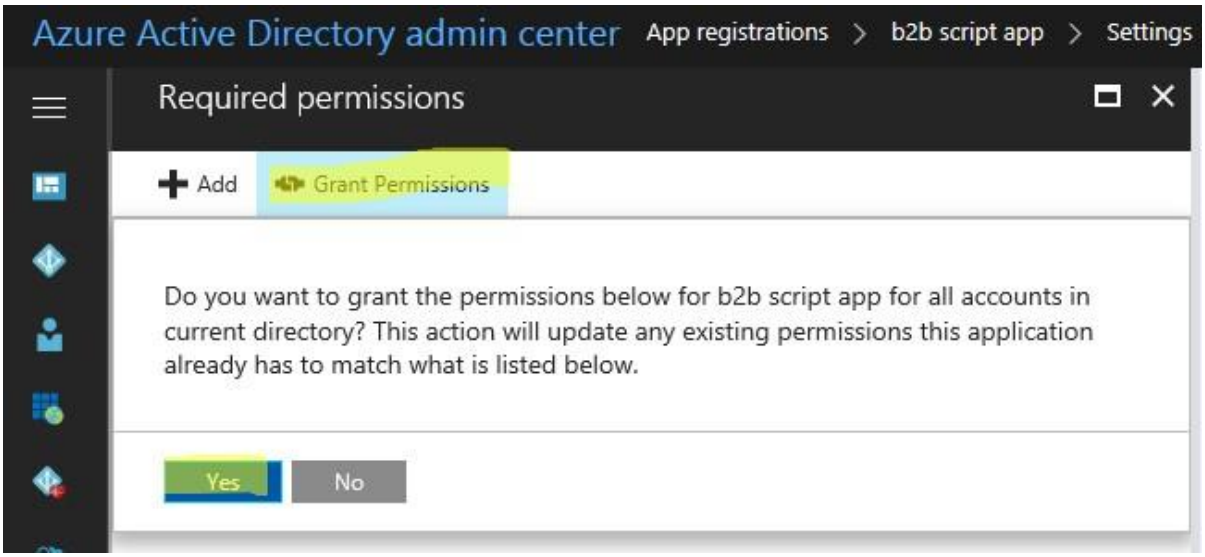
5. Under **Enable Access**, select the following permission:

    - **APPLICATION PERMISSIONS**
        - **Read directory data**



6. Click **Save**.

7. Under **Required permissions**, click **Grant Permissions**, and then click **Yes**.

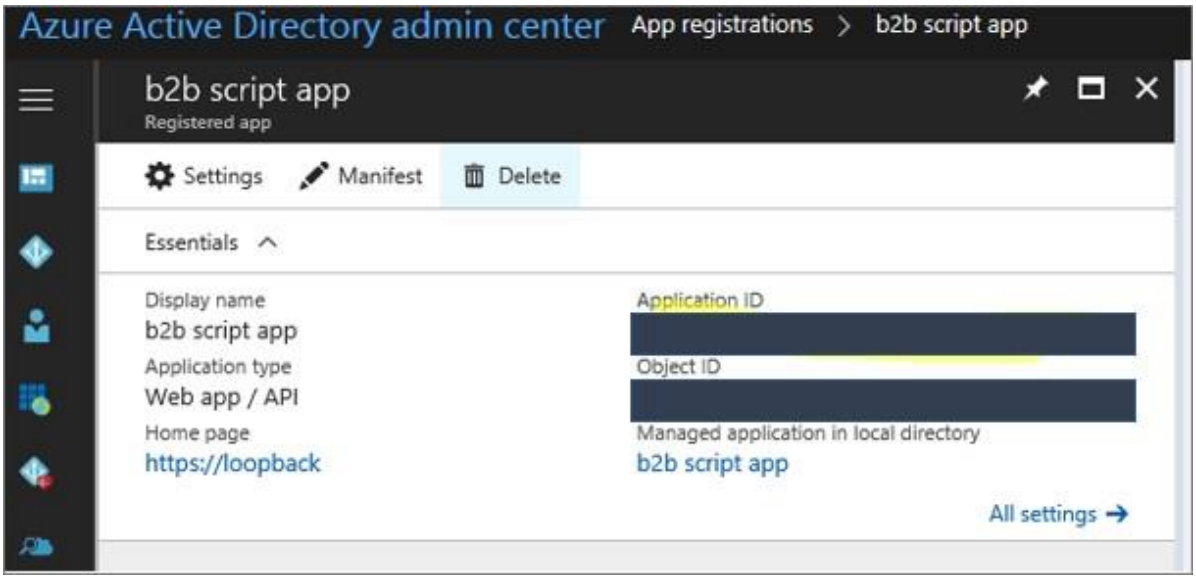


8. Under **Settings**, click **Keys**. Under **Description**, enter a name for the application key, like “Key 1”. Under **Expires**, select 1 or 2 years. (**NOTE**: You or someone in your organization will need to make a note to come back and refresh this key before it expires.)

9. Click **Save**. An application secret will be generated and displayed. **COPY** this key and record it. You'll need it in a minute when you set up the script.

   NOTE: This key will not be displayed again and cannot be retrieved. If you lose it, you'll have to come back, delete it, and create another one.

10. Finally, record the **Application ID**. You can click to the right of the ID value in the main panel and it will copy the value to your clipboard. Record it along with the app secret from above. These two strings will be needed to set up the script



Install required PowerShell modules

The script uses two PowerShell modules:

- Active Directory:

```
Install-WindowsFeature -Name RSAT-AD-PowerShell
```

- Azure Active Directory v2 Preview:

```
Install-Module azuread `
-Force `
-MinimumVersion 2.0.2.4 `
-SkipPublisherCheck
```

NOTE: The previous command requires Windows Server 2016 or Windows Management Framework 5.0.

Create an Azure AD group for guests that should be pulled into the on-premises directory

To prevent pulling all users into your on-premises environment, the sample script filters guest users by group membership. In Azure AD, make sure that the guest user objects that you want to pull into your on-premises directory are stored in a specific group.

You can identify the group with PowerShell:

```
Get-AzureADGroup
```

You can verify group membership by using this cmdlet:

```
Get-AzureADGroupMember -ObjectId <someObjectID>
```

Create a dedicated OU to store the guest user accounts

You must create a dedicated organizational unit (OU) that will store the guest user objects that are pulled into the on-premises directory.

> **IMPORTANT**: This OU must **NOT** be synchronized back to Azure AD using Azure AD Connect. Make sure to exclude the guest user objects from the synchronization scope. See Organizational unit based filtering.

You can pull the guest user objects into either a specific OU in a production forest/domain, or to a forest/domain that's dedicated to hold these guest user objects.

Create a dedicated OU to store retired guest user accounts

You must create a dedicated organizational unit (OU) that will store retired guest user objects that were previously pulled into the on-premises directory.

> **IMPORTANT**: This OU must **NOT** be synchronized back to Azure AD using Azure AD Connect. Make sure to exclude the guest user objects from the synchronization scope. See Organizational unit based filtering.

Add the UPN suffix to the forest into which external accounts will be pulled

You must use add the .onmicrosoft.com name of your Azure AD tenant as a UPN suffix to the forest into which the guest user objects will be pulled. To do this, you can use the `Set-ADForest` PowerShell cmdlet. For example:

```
Get-ADForest | Set-ADForest -UPNSuffixes @{Add="contoso.onmicrosoft.com"}
```

Security considerations

By default, the sample script configures the guest user objects to use a random strong password. Also, to prevent unauthorized access, the `SmartcardLogonRequired` parameter is set to `$true`. To help prevent unauthorized access, do not change these settings.

For increased security, consider modifying script to use certificate-based authentication to impersonate a Service Principal in Azure AD: Using a Service Principal to connect to a directory in PowerShell.

Edit and run the script

Copy the script and save it locally to the computer on which you have the required PowerShell modules installed. In the **Set up variables** section, edit the values. This is where you'll specify values such as the application ID and the key that you copied earlier. Test the script manually before you configure it as a scheduled task (if desired).

Optional: Run the script as a scheduled task using a GMSA

You can set up the script to run regularly as a scheduled task. We recommend that you use a Group Managed Service Account to run the scheduled task.

Step 1: Create a Group Managed Service Account

The following example creates a GMSA:

```
#running this command requires Domain Administrator Credentials
$cpu = Get-ADComputer <ComputerName> #Server that will be running the script
$acctName = "gmsa_b2b_script"

New-ADServiceAccount -Description "Account for creating B2B users" `
-DisplayName $acctName `
-DNSHostName "$acctName.contoso.com" `
-Name $acctName `
-PrincipalsAllowedToRetrieveManagedPassword $cpu

Install-ADServiceAccount $acctName
```

Step 2: Register the scheduled task

The following example registers a scheduled task to run the script on a daily basis.

```
$action = New-ScheduledTaskAction -Execute powershell.exe `
-Argument "-NonInteractive -NoLogo -NoProfile -File c:\scripts\AppProxy-
GuestAccountCreation-v1.0.3.ps1"

$trigger = New-ScheduledTaskTrigger -At 7:00 -Daily

$principal = New-ScheduledTaskPrincipal -UserId corp\gmsa_b2b_script$ `
-LogonType Password

Register-ScheduledTask CreateB2BGuestUserObjects `
-Principal $principal `
-Action $action `
-Trigger $trigger
```

Step 3: Delegate permissions to both target OUs for GMSA

The account running the script requires Write permissions to the OU that's dedicated to the guest user accounts that you want to pull in, as shown in the following example:

```
dsacls.exe "OU=B2BGuestUserObjects,DC=corp,DC=contoso,DC=com" /G
"corp\gmsa_b2b_script$:GA" /I:T


dsacls.exe "OU=B2BArchivedGuestUserObjects,DC=corp,DC=contoso,DC=com" /G
"corp\gmsa_b2b_script$:GA" /I:T
```