# Service Principal Password Rotation

Advanced Conversational Engagement

Exported on 06/06/2025

# Table of Contents

At the time of Service Principal creation we push three secrets into our Key Vault, namely, AppId, ObjectId and password of a particular SPN. As per FSCP policies and best practice guidelines it is necessary to all the secrets pushed into the key vault to have an expiry date. As well as the Function App  API which we use to create the SPN provided by the FSCP team also adds an expiry date to the SPN password in Microsoft entraID. If the secrets and password are not updated and rotated before it's expiry it result in a non-compliancy on the RCF dashboard.
To overcome this non-compliancy, we have come up with an SPN password rotation pipeline[1] which runs every morning 5AM CET.

1 https://dev.azure.com/cbsp-abnamro/GRD0001014/_build?definitionId=99864

# 1 Pipeline structure

The Aim of this pipeline is to rotate the password before it's expiry and to achieve this we are making use of a PowerShell script.

```
              $daysBeforeExpiration = (Get-Date).AddDays(30)
              $secrets = Get-AzKeyVaultSecret -VaultName ${{ variables.keyVaultName }}
| Where-Object Name -like "*sp*-password"
              # Loop through each secret
              foreach ($secret in $secrets) {
                  # Get the expiration date of the secret
                  # Check if the secret is expiring in the next 30 days
                  if ( $secret.Expires -ILE ([DateTime]($daysBeforeExpiration)))
                  {
                    $namewithoutpassword = $secret.Name.Replace("-password","")
                    $secretName = $secret.Name
                    $appOutput = Get-AzADApplication -displayname $namewithoutpassword
                    $appObjectId= $appOutput.Id
                    Write-Host $secret.Name is going to expire and should be renewed

                    #Add client secret for SPN

                    $token = (Get-AzAccessToken -ResourceUrl "https://
graph.microsoft.com/").Token
                    $uri = "https://graph.microsoft.com/v1.0/applications/$appObjectId/
addPassword"

                    $headers = @{
                        "Authorization" = "Bearer $token"
                        "Content-Type"  = "application/json"
                    }
                    $body = @{
                        "passwordCredential" = @{
                            "displayName"    = "($namewithoutpassword)"
                            "startDateTime" = (Get-Date).ToUniversalTime().ToString("o")
                            "endDateTime"    = (Get-Date).AddDays(365).ToUniversalTime()
.ToString("o")
                        }
                    } | ConvertTo-Json -Compress

                    $result = Invoke-RestMethod -Uri $uri -Method 'Post' -Headers
$headers -Body $body -UseBasicParsing
                    $secretText = ConvertTo-SecureString $($result).secretText
-AsPlainText -Force
                    $endDateTime = $($result).endDateTime
                    $expireDateTime =
([DateTimeOffset]::Parse($endDateTime)).ToString("yyyy-MM-ddTHH:mm:ssZ")
```

```
                  # Add secret to Key VAult: https://dev.azure.com/cbsp-abnamro/
Azure/_wiki/wikis/Azure.wiki/73442/Service-Principal-Deploy-using-Function-App?
anchor=serviceprincipal-secrets-/-key-rotation
                  Set-AzKeyVaultSecret -VaultName ${{ variables.keyVaultName }} -Name
$secretName -SecretValue $secretText -Expires $expireDateTime

                  # Delete near expiry secretID's
                  $spnSecrets = Get-AzADAppCredential -DisplayName
$namewithoutpassword
                  foreach ( $spnSecret in $spnSecrets ){
                      if ( $spnSecret.EndDateTime -NE $expireDateTime)
                      {
                          Write-Host $spnSecret.keyId will expire on
$spnSecret.EndDateTime
                          Remove-AzADAppCredential -DisplayName
$spnSecret.displayName -KeyId $spnSecret.keyId
                      }
                      else{
                          Write-Host $spnSecret.keyId is current secret which will
expire on $spnSecret.EndDateTime
                      }
                  }
                }
                else
                {
                  Write-Host "$($error[0].exception.Message)" -ForegroundColor Yellow
                  Write-Host $secret.Name is not going to expire.
                  Write-Host "##vso[task.setvariable variable=secretexpiring]false"
                }
              }
```

Steps involving in the script:

1. Declare a variable to check if the expiry date of the secrets is less then the 30 days from the day this pipeline is running.
2. Filter out all the secrets related to the SPN form the Key vault using PowerShell command and store it in a variable.
3. Loop through each of this secrets to check if they are going to expiry in less than 30 days.
4. If the secrets is going to expire in less then 30 days,
    a. then call the FSCP provided function app API by passing the SPN's appID in the URI header and get the new password.
    b. Push this new password into the Key Vault with the updated expiry date.
    c. Delete the older passwords form **Microsoft EntraID** using another loop and PowerShell command.
5. Else, when the secrets is not going to expire we display the output for the same.

# 2 Refrence

FSCP WIKI: Service Principal - Deploy using Function App - Overview[2]

Repository: spn_rotation.yaml - Repos[3]

Pipeline: Pipelines - Runs for SPN-password-rotation[4]

2 https://dev.azure.com/cbsp-abnamro/Azure/_wiki/wikis/Azure.wiki/73442/Service-Principal-Deploy-using-Function-App?anchor=serviceprincipal-secrets-/-key-rotation

3 https://dev.azure.com/cbsp-abnamro/GRD0001014/_git/acep-platform-infrastructure?path=/scheduler/spn_rotation.yaml

4 https://dev.azure.com/cbsp-abnamro/GRD0001014/_build?definitionId=99864

# 3 Note

- Our key vault contains secrets for the spn with the suffix sp-password and spn-password   we are filtering SPN's considering both naming conventions.
- We are adding tags to the SPN appId and ObjectId secrets at the time of SPN creation. So the rotation for them is taken care by the secret rotation pipeline.
- Even after Successful SPN password rotation, the SPN's in Microsoft entraID display near to expiry status. Hence it is crucial to delete the older secret's for the Microsoft entraID at the time of rotation so as to maintain consistence in the secret value and compliances with the policies.