# Objective

To setup proactive monitoring of secrets and certificates of all Azure app registrations within the tenant. The alert reports for app registrations that expires in the next 90 days. This proactive monitoring will eliminate the incidents of pipelines/jobs failing due to expired app registrations.

# Pre-requisites

* SMTP for sending out the alert email
  + If no existing SMTP, can deploy Azure Communication Service (Email). CA$0.00034/Email sent.

[*https://learn.microsoft.com/en-us/azure/communication-services/concepts/email/email-overview*](https://learn.microsoft.com/en-us/azure/communication-services/concepts/email/email-overview)

* Automation Account with system-assigned managed identity enabled
* Global Admin to run below script to grant the Automation Account Identity with permission to view all app registrations

*Ref:* [*https://learn.microsoft.com/en-us/graph/permissions-reference#applicationreadall*](https://learn.microsoft.com/en-us/graph/permissions-reference#applicationreadall)

$TenantID="<TenantID>"

$GraphAppId = "00000003-0000-0000-c000-000000000000"

$DisplayNameOfMSI="<AutomationAccountName>"

$PermissionName = "Application.Read.All"

# Install the module

Install-Module AzureAD

Connect-AzureAD -TenantId $TenantID

$MSI = (Get-AzureADServicePrincipal -Filter "displayName eq '$DisplayNameOfMSI'")

Start-Sleep -Seconds 10

$GraphServicePrincipal = Get-AzureADServicePrincipal -Filter "appId eq '$GraphAppId'"

$AppRole = $GraphServicePrincipal.AppRoles | `

Where-Object {$\_.Value -eq $PermissionName -and $\_.AllowedMemberTypes -contains "Application"}

New-AzureAdServiceAppRoleAssignment -ObjectId $MSI.ObjectId -PrincipalId $MSI.ObjectId -ResourceId $GraphServicePrincipal.ObjectId -Id $AppRole.Id

* To verify the permission is set, go to Microsoft Entra ID 🡪 Enterprise applications, search for the Automation Account managed-identity and click on it
* On the Automation Account managed-identity page, expand Security and select Permissions, and you should see below

A screenshot of a computer

Description automatically generated

# Azure Automation Modules

Install the following modules on the Automation Account:

* Az - 7.2 runtime version
* Microsoft.Graph.Authentication - 7.2 runtime version
* Microsoft.Graph.Applications - 7.2 runtime version

# Azure Automation Global Variables

Create variables to store SMTP username and password. Ensure the created variables are encrypted.

Sample:

A screenshot of a computer

Description automatically generated

# Azure Automation Runbook

1. Create a PowerShell runbook with runtime version of 7.2
2. Paste below script

Param(

  [Parameter(Mandatory=$True)]

  [String] $destEmailAddress,

  [Parameter(Mandatory=$True)]

  [String] $fromEmailAddress,

  [Parameter(Mandatory=$True)]

  [String] $subject,

  [Parameter(Mandatory=$True)]

  [String] $content

)

# Ensures runbook does not inherit an AzContext

$null = Disable-AzContextAutosave -Scope Process

# Connect to Microsoft Graph

$null = Connect-MgGraph -Identity

# Set timezone to EST

$tempDate =(Get-Date).ToUniversalTime()

$tzEST = [System.TimeZoneInfo]::FindSystemTimeZoneById("Eastern Standard Time")

$CurrentTimeEST = [System.TimeZoneInfo]::ConvertTimeFromUtc($tempDate, $tzEST)

$Date = Get-Date -Date $CurrentTimeEST

# Retrieve SMTP keys from global variables

$SMTP\_ApiKey = Get-AutomationVariable -Name "smtpapikey"

$SMTP\_ApiKey\_Secret = Get-AutomationVariable -Name "smtpapikeysecret"

$password = ConvertTo-SecureString $SMTP\_ApiKey\_Secret -AsPlainText -Force

$Cred = New-Object System.Management.Automation.PSCredential ($SMTP\_ApiKey, $password)

# Create CSV file

$tempCSVfile = Get-ChildItem ([IO.Path]::GetTempFileName()) | Rename-Item -NewName { [IO.Path]::ChangeExtension($\_, ".csv") } -PassThru

"DisplayName,AppId,ExpiryDate,logDate" | Add-Content -Path $tempCSVfile

# Collect App registrations

$Apps = Get-MgApplication -All

$ctr1 = 0

ForEach ($App in $Apps){

    $ctr2 = 0

    $DisplayName = $App.DisplayName

    $AppId = $App.AppId

    [Array]$ExpiryDate = $App.PasswordCredentials.EndDateTime

    If ($ExpiryDate.Count -gt 0){

        While ($ExpiryDate[$ctr2]){

            If (($ExpiryDate[$ctr2] -gt $Date) -and ($ExpiryDate[$ctr2] -lt ($Date).AddDays(90))){

            $ctr1 = 1

            $Expiry = $ExpiryDate[$ctr2]

            "$DisplayName,$AppId,$Expiry,$Date" | Add-Content -Path $tempCSVfile

            }

            $ctr2++

        }

    }

}

If ($ctr1 -eq 1){

    # Email the CSV file

    Send-MailMessage `

            -smtpServer "in-v3.mailjet.com" `

            -Credential $Cred `

            -Usessl `

            -Port 587 `

            -from $fromEmailAddress `

            -to $destEmailAddress `

            -subject $subject `

            -Body $content `

            -Attachments $tempCSVfile

}

# EoF

1. Publish the runbook and setup execution schedule with below parameters (as applicable per client environment)

A screenshot of a phone

Description automatically generated