

PHASE 16: Automation & Scripting with PowerShell & Microsoft Graph API

Automate and streamline IT tasks in your hybrid infrastructure using secure, script-driven processes. This includes onboarding users, Intune configuration management, AD data cleanup, and backup exports—all using free tools and Microsoft APIs.

Technology Used:

- PowerShell 7+
 - Microsoft Graph API
 - AzureAD / Entra ID PowerShell module
 - Microsoft.Graph.Intune SDK
 - Task Scheduler
-

I. Setup PowerShell Modules & Environment

Actions Completed:

Step 1: Install Required Modules

Open PowerShell as Administrator and install the following:

1. `Install-Module Microsoft.Graph -Scope CurrentUser`
2. `Install-Module Microsoft.Graph.Intune -Scope CurrentUser`
3. `Install-Module AzureAD -Scope CurrentUser`
4. `Install-Module ImportExcel -Scope CurrentUser`

Step 2: Connect to Services:

1. `Connect-MgGraph -Scopes "User.Read.All", "Device.Read.All", "Directory.ReadWrite.All"`
2. `Connect-AzureAD`
3. `Import-Module Microsoft.Graph.Intune`

```
DC01 on DESKTOP-8Q8HBQ5 - Virtual Machine Connection
File Action Media View Help
Windows PowerShell
Connect-MgGraph : InteractiveBrowserCredential authentication failed: User canceled authentication.
at line:1 char:1
+ Connect-MgGraph -Scopes "User.Read.All", "Device.Read.All", "Director ...
+ ~~~~~
+ CategoryInfo          : NotSpecified: (:) [Connect-MgGraph], AuthenticationFailedException
+ FullyQualifiedErrorId : Microsoft.Graph.PowerShell.Authentication.Cmdlets.ConnectMgGraph

PS C:\Users\Dipesh.admin> Connect-MgGraph -Scopes "User.Read.All", "Device.Read.All", "Directory.ReadWrite.All"
Welcome to Microsoft Graph!

Connected via delegated access using 14d82eec-204b-4c2f-b7e8-296a70dab67e
Readme: https://aka.ms/graph/sdk/powershell
SDK Docs: https://aka.ms/graph/sdk/powershell/docs
API Docs: https://aka.ms/graph/docs

NOTE: You can use the -NoWelcome parameter to suppress this message.

PS C:\Users\Dipesh.admin> Connect-AzureAD Import-Module Microsoft.Graph.Intune
Connect-AzureAD : A positional parameter cannot be found that accepts argument 'Import-Module'.
at line:1 char:1
+ Connect-AzureAD Import-Module Microsoft.Graph.Intune
+ ~~~~~
+ CategoryInfo          : InvalidArgument: (:) [Connect-AzureAD], ParameterBindingException
+ FullyQualifiedErrorId : PositionalParameterNotFound,Microsoft.Open.Azure.CommonLibrary.ConnectAzureAD

PS C:\Users\Dipesh.admin> Connect-AzureAD
WARNING: Install the latest PowerShell module, the Microsoft Graph PowerShell SDK, for new features and improvements!
https://aka.ms/graphPSmigration

Account Environment TenantId TenantDomain AccountType
-----
admin@nxhz.onmicrosoft.com AzureCloud 34941541-7999-4106-82f9-84caa8d867cf nxhz.onmicrosoft.com User

PS C:\Users\Dipesh.admin> Import-Module Microsoft.Graph.Intune
PS C:\Users\Dipesh.admin> _
```

✓ Outcome:

Your workstation is now authenticated and authorized to read/write users, groups, devices, and Intune configuration from Microsoft 365.

II. Key Automation Scripts Implemented

Actions Completed:

1. Cloud-Only User Onboarding & License Assignment

```
$user = @{}
accountEnabled = $true
```

```

displayName = "John Doe"
mailNickname = "jdoe"
userPrincipalName = "jdoe@dipeshcorp.onmicrosoft.com"
passwordProfile = @{
    forceChangePasswordNextSignIn = $true
    password = "Welcome123@"
}
}
New-MgUser -BodyParameter $user

```

```

PS C:\Users\Dipesh.admin> $user = @{
>> accountEnabled = $true
>> displayName = "John Do"
>> mailNickname = "jdoe"
>> userPrincipalName = "jdoe@nxhz.onmicrosoft.com"
>> passwordProfile = @{
>> forceChangePasswordNextSignIn = $true
>> password = "Welcome123@"
>> }
>> }
>> New-MgUser -BodyParameter $user

```

DisplayName	Id	Mail	UserPrincipalName
John Do	d55bb7df-9990-4d15-ac14-a03ce7f8eb82		jdoe@nxhz.onmicrosoft.com

```

PS C:\Users\Dipesh.admin>

```

Assign M365 license

```

Assign-MgUserLicense -UserId "jdoe@dipeshcorp.onmicrosoft.com" -AddLicenses @{Skuld =
"<your-sku-guid>"}

```

Result: Created user and assigned license in Entra ID using a single script.

2. BitLocker Recovery Key Export (On-Prem Devices)

```

Get-ADComputer -Filter * -Properties 'msFVE-RecoveryInformation' |
Where-Object { $_.'msFVE-RecoveryInformation' } |

```

```
Select-Object Name, @{Name="RecoveryKey";  
Expression={{$. 'msFVE-RecoveryInformation'.RecoveryPassword}} |  
Export-Csv BitLockerKeys.csv -NoTypeInfoInformation
```

Result: Periodic backup of BitLocker recovery keys to encrypted storage.

3. Intune Configuration Backup

```
Get-IntuneConfigurationPolicy | Export-Clixml -Path "C:\backups\IntuneProfiles.xml"
```

Result: Exported all configuration profiles from Intune for disaster recovery or audit purposes.

4. Autopilot Hardware Hash Export for New Devices

```
Install-Script -Name Get-WindowsAutoPilotInfo  
Set-ExecutionPolicy RemoteSigned -Scope Process  
Get-WindowsAutoPilotInfo -OutputFile "autopilot-hash.csv"
```

Result: Hardware hash captured for seamless Autopilot provisioning.

5. Identify Stale Devices (>60 Days Inactive)

```
$devices = Get-AzureADDevice  
$stale = $devices | Where-Object { $_.ApproximateLastSignInTimeStamp -lt  
(Get-Date).AddDays(-60) } $stale | Export-Csv .\StaleDevices.csv
```

Result: Enabled cleanup of inactive cloud-registered devices and better license hygiene.

6. Automate Script Execution via Task Scheduler

Scheduled a PowerShell script (e.g., StaleDeviceScan.ps1) to run every morning:

Trigger: Daily at 6 AM

Action: powershell.exe -ExecutionPolicy Bypass -File "C:\scripts\StaleDeviceScan.ps1"

Output emailed to admin inbox using Send-MailMessage.

7. Secure Credential Management for Automation

Store credentials securely:

```
$cred = Get-Credential
```

```
$cred | Export-Clixml -Path "C:\secure\admin.xml"
```

Load in script:

```
$creds = Import-Clixml "C:\secure\admin.xml"
```

Used for automated scripts that access Graph API or Exchange Online without interactive login.