PowerShell : commandes, comptes et scripts

Formation : BTS SIO — Option SISR



1. Rôle et fonctionnement

PowerShell manipule des objets. Le pipeline transmet ces objets entre commandes, ce qui permet de filtrer des services ou des comptes puis d'afficher leurs propriétés. Les verbes `Get`, `New`, `Set` et `Remove` indiquent généralement la nature de l'action.

2. Inspection avant modification

```
Get-Help Get-Service -Examples
Get-Command *LocalUser*
Get-Service | Where-Object Status -eq 'Running'
Get-NetAdapter | Select-Object Name, Status, LinkSpeed
Get-LocalUser | Select-Object Name, Enabled
```

`Select-Object` choisit les propriétés ; `Where-Object` filtre les objets. Une session élevée est nécessaire pour de nombreuses modifications système, mais pas pour toute consultation.

3. Compte local d'administration

Exemple à adapter au compte attendu, après vérification de son absence :

```
$CredentialPassword = Read-Host 'Mot de passe du compte' -AsSecureString
New-LocalUser -Name 'ADM-SRV' -Password $CredentialPassword
```

L'appartenance à un groupe doit découler du rôle demandé. Un utilisateur de passerelle WAC ne doit pas recevoir automatiquement tous les droits locaux du serveur.

4. Scripts et mises à jour

Enregistrer les scripts en `.ps1`, relire les modifications avec Git puis exécuter une version identifiée. Utiliser `-WhatIf` lorsqu'une commande le propose. Une stratégie d'exécution PowerShell encadre le

lancement des scripts ; elle ne remplace pas l'analyse de leur contenu.

Le module **PSWindowsUpdate**, utilisé dans la maquette, est un module tiers distribué via PowerShell Gallery. Avant installation, vérifier son éditeur, sa version et les dépendances ; ne pas le présenter comme une commande native de Windows.

5. Vérification et dépannage

Relever la version avec `$PSVersionTable`, consulter `Get-Help` et identifier la commande ayant échoué. Conserver les sorties utiles à la recette, sans secret ni transcription de saisie confidentielle. Un script modifié doit être relu et testé avant son utilisation sur plusieurs serveurs.

6. Renommer un compte sans changer son identité

```
Get-LocalUser -Name "Administrateur" | Select-Object Name, SID
Rename-LocalUser -Name "Administrateur" -NewName "ADM-SRV-00"
Get-LocalUser -Name "ADM-SRV-00" | Select-Object Name, SID
```

Dans l'observation du laboratoire, le SID reste identique après renommage. Le suffixe RID 500 identifie toujours le compte administrateur intégré.

Pour modifier le mot de passe sans l'inscrire dans l'historique de commandes :

```
$NouveauSecret = Read-Host "Nouveau mot de passe" -AsSecureString
Set-LocalUser -Name "ADM-SRV-00" -Password $NouveauSecret
```

7. Script de diagnostic réseau

Cet exemple peut être enregistré dans `diagnostic-reseau.ps1`, puis versionné avec Git.

```
Write-Host "Configuration réseau"
Get-NetIPConfiguration
Write-Host "Test de la passerelle"
Test-NetConnection -ComputerName 192.168.3.126
Write-Host "Résolution DNS"
Resolve-DnsName -Name ns0.californie.cub.sioplc.fr -Server 192.168.3.10
```

Les commandes affichent des observations distinctes : une réponse DNS ne démontre pas que l'accès applicatif fonctionne.

8. Références

- Microsoft — documentation PowerShell

- PowerShell Gallery — PSWindowsUpdate

From:
https://wiki.ewengadonnaud.xyz/ - **Base de savoir réseaux/cyber/devops**

Permanent link:
https://wiki.ewengadonnaud.xyz/doku.php?id=administration:windows:powershell

Last update: **2026/10/18 01:23**