TP 2 : Objet Logs et utilisateurs

Sébastien Bindel

Mars 2024

Introduction

Ce deuxième TP a pour objectif de vous faire découvrir les journaux sous Windows et la gestion des utilisateurs. Vous utiliserez durant ce TP une machine virtuelle **Windows server 2019**.

Au cours de ce TP, vous devrez produire un compte rendu au format PDF qui contient les commandes et scripts PowerShell associés aux questions accompagnés d'une explication. La première page du rapport devra contenir votre nom et prénom ainsi que le numéro du TP réalisé. Ces comptes rendus devront être déposés sur Moodle, les noms des comptes rendus devront être de la forme *TP1-PowerShell-Nom-Prénom*.

I. Gestion locale des utilisateurs

Il est possible de gérer les comptes des utilisateurs locaux ainsi que les groupes avec Powershell. L'ensemble des commandes est disponible avec la commande ci-dessous.

```
Get-Command -Module Microsoft.PowerShell.LocalAccounts
```

exercice 1:
Dresser la liste des utilisateurs de la machine.

exercice 2:
Après avoir créer un nouvel utilisateur avec Powershell, dans quel groupe celui-ci est présent ?

exercice 3:
Créer un script qui créer un utilisateur dans le groupe Utilisateurs, le nom et le mot de passe sont demandés par le script et le compte sera désactivé par défaut. L'absence de nom d'utilisateur et de mot de passe seront gérés par l'affichage d'un message d'erreur. Pour l'entrée du mot de passe l'option -AsSecureString peut être mise à la commande Read-Host.

exercice 4:
Écrire un script qui affiche le nom d'utilisateur avec la date de sa dernière connexion.

Dans le programme suivant on désire

exercice 5:
Écrire un script qui se nomme verifie_doublon.ps1 et qui vérifie si le groupe ou l'utilisateur existe. Le programme prendra en paramètre le nom d'utilisateur ou de groupe que l'on pourra distinguer avec une option (-g pour groupe ou -u pour utilisateur). Le code ci-dessous montre un exemple d'utilisation.

```
./verifie_doublon -g Invite # verification de l'existene du groupe
    Invite
./verifie_doublon -u toto # verification de l'existence de l'
    utilisateur toto
```

La gestion des arguments donnés par l'utilisateur se fait en déclarant une fonction param au début du script. En testant le code ci-dessous, modifiez le pour qu'il correspond à vos besoins.

```
param(
      [Parameter()]
      [string]$Parameter1,

      [Parameter()]
      [string]$Parameter2
 )

 Write-Host "Parameter 1 value is $Parameter1"
 Write-Host "Parameter 2 value is $Parameter2"
```

II. Gestion des logs

L'accès aux journaux Windows se fait avec la commande `Get-EventLog`.

exercice 6:
Quelle option permet d'afficher la liste des logs disponibles ?

exercice 7:
Afficher les sources du log système, sans doublon de préférence.

exercice 8:
En utilisant en inspectant le log *System* et la source *user32*, écrire un script qui ecrit dans un fichier CSV (la commande Export-CSV peut être utilisée) les raisons de l'extinction du serveur avec la date.

exercice 9:
Récupérer un nombre d'erreurs systèmes de votre choix et compter les erreurs survenues sur plusieurs journées. Exporter ces informations dans un ficher csv avec la commande Export-Csv.