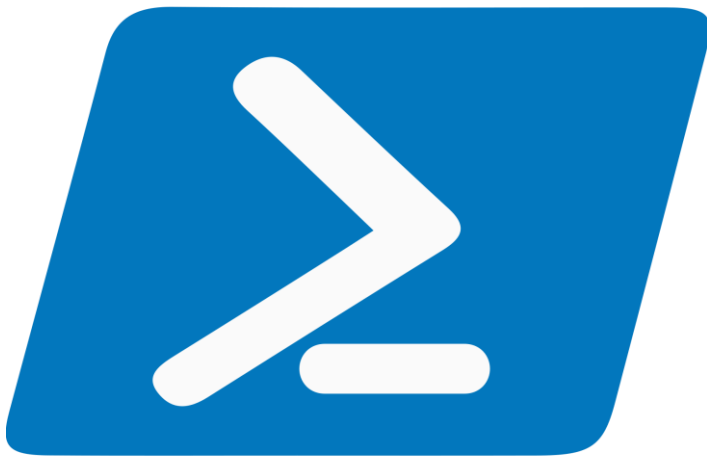


# PowerShell

---



29 MARS

---

**GUISSE**  
**Benjamin**



---

# Utilisation de PowerShell pour Ajouter des Utilisateurs

En informatique, gérer les utilisateurs est une tâche essentielle, mais elle peut souvent être longue et exigeante. L'ajout d'utilisateurs dans une infrastructure peut prendre beaucoup de temps et demande une attention particulière pour assurer la sécurité et l'intégrité des informations.

Microsoft a développé PowerShell, un langage de script et d'automatisation, pour simplifier ce type de tâche. Grâce à son interface en ligne de commande (CLI), PowerShell permet d'automatiser de nombreuses opérations, y compris l'ajout d'utilisateurs à un système.

L'objectif est ici de montrer comment utiliser PowerShell pour ajouter une liste d'utilisateurs automatiquement, ce qui évite de le faire manuellement un par un.

## Vérification et Outils Utilisés

Pour garantir que le processus fonctionne correctement, nous effectuerons les vérifications suivantes :

- Assurer que les utilisateurs sont correctement ajoutés dans Active Directory.
- Vérifier les données importées à partir d'un fichier CSV.
- Utiliser les protocoles LDAP pour vérifier l'intégrité des informations.

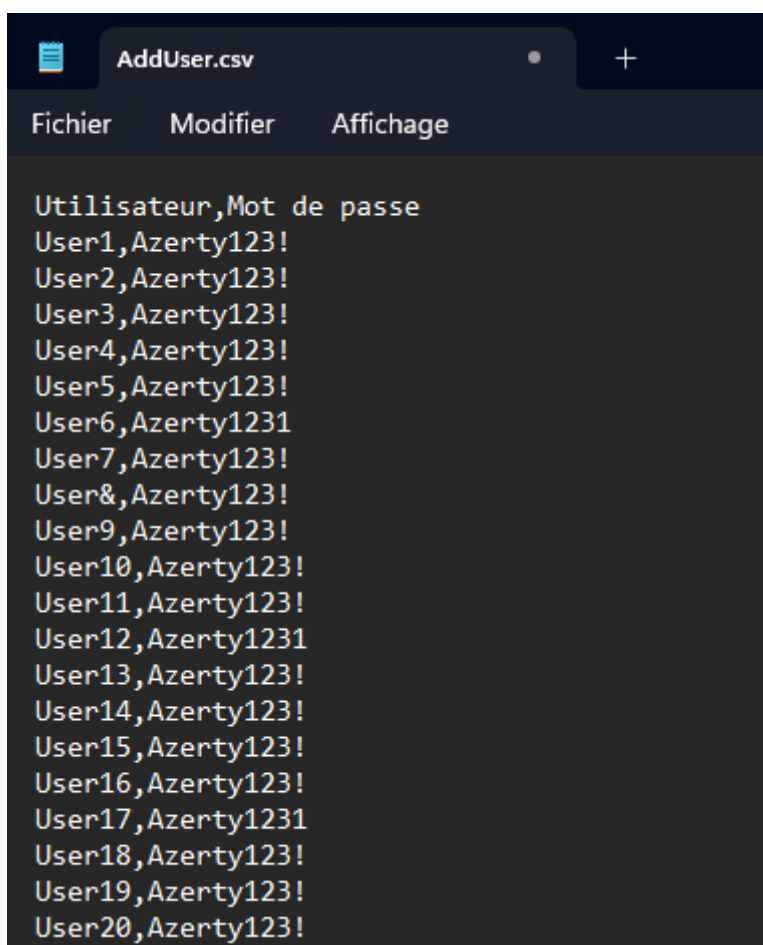
Les outils et technologies nécessaires incluent :

- Active Directory pour la gestion des utilisateurs.
- PowerShell pour automatiser le processus.
- Fichiers CSV pour importer les données des utilisateurs.

# Préparation du Fichier CSV pour Ajouter des Utilisateurs

Pour commencer, il est nécessaire de créer un fichier CSV contenant les informations requises pour la création des utilisateurs. Dans cet exemple, nous inclurons uniquement les noms d'utilisateurs et leurs mots de passe, mais il est tout à fait possible d'ajouter des informations supplémentaires comme l'adresse e-mail, une description, etc.

Remarque : Lors de la création du fichier CSV, assurez-vous que la première ligne correspond aux noms des colonnes. Les différentes informations doivent être séparées par des virgules.



```

AddUser.csv
Fichier  Modifier  Affichage

Utilisateur,Mot de passe
User1,Azerty123!
User2,Azerty123!
User3,Azerty123!
User4,Azerty123!
User5,Azerty123!
User6,Azerty1231
User7,Azerty123!
User&,Azerty123!
User9,Azerty123!
User10,Azerty123!
User11,Azerty123!
User12,Azerty1231
User13,Azerty123!
User14,Azerty123!
User15,Azerty123!
User16,Azerty123!
User17,Azerty1231
User18,Azerty123!
User19,Azerty123!
User20,Azerty123!
```

# Importation et Traitement du Fichier CSV

Une fois le fichier CSV préparé, il faut spécifier son chemin d'accès et l'importer dans PowerShell. Le script va alors stocker les informations dans des variables, comme `Utilisateur.Prenom`, `Utilisateur.Login`, et `Utilisateur.MotDePasse`.

Pour améliorer la sécurité du processus de création, il est possible de définir des conditions. Par exemple, vérifier si l'utilisateur n'existe pas déjà dans Active Directory pour éviter les doublons. Si les conditions sont remplies, le script utilise les variables pour créer les nouveaux utilisateurs.

```
Teeest.ps1  Untitled-1  AddUser.ps1 X
AddUser.ps1 > ...
1  $CSVFile = "C:\Windows\Temp\AddUser.csv"
2  $CSVData = Import-CSV -Path $CSVFile -Delimiter "," -Encoding UTF8
3
4  foreach ($Utilisateur in $CSVData) {
5      $UtilisateurPrenom = $Utilisateur.Prenom
6      $UtilisateurNom = $Utilisateur.Nom
7      $UtilisateurLogin = $Utilisateur.Login
8      $UtilisateurMotDePasse = $Utilisateur."Mot de passe"
9
10     # Vérifier la présence de l'utilisateur dans l'AD
11     if (Get-ADUser -Filter {SamAccountName -eq $UtilisateurLogin}) {
12         Write-Warning "L'identifiant $UtilisateurLogin existe déjà dans l'AD"
13     } else {
14         New-ADUser -Name "$UtilisateurPrenom $UtilisateurNom" `
15             -DisplayName "$UtilisateurPrenom $UtilisateurNom" `
16             -GivenName $UtilisateurPrenom `
17             -Surname $UtilisateurNom `
18             -SamAccountName $UtilisateurLogin `
19             -Path "OU=Utilisateur,OU=RT,DC=MOSCOU,DC=LOCAL" `
20             -AccountPassword (ConvertTo-SecureString $UtilisateurMotDePasse -AsPlainText -Force) `
21             -ChangePasswordAtLogon $true `
22             -Enabled $true
23
24         Write-Output "Création de l'utilisateur : $UtilisateurLogin ($UtilisateurNom $UtilisateurPrenom)"
25     }
26 }
27
```

Exécution du script, puis nous observons par la suite la création des utilisateurs.

```
Création de l'utilisateur : User50 (User50)
PS C:\Users\Administrateur> C:\Users\Administrateur\Desktop\Sans titre1.ps1
Création de l'utilisateur : User1 (User1)
Création de l'utilisateur : User2 (User2)
Création de l'utilisateur : User3 (User3)
Création de l'utilisateur : User4 (User4)
Création de l'utilisateur : User5 (User5)
Création de l'utilisateur : User6 (User6)
Création de l'utilisateur : User7 (User7)
Création de l'utilisateur : User8 (User8)
Création de l'utilisateur : User9 (User9)
Création de l'utilisateur : User10 (User10)
Création de l'utilisateur : User11 (User11)
Création de l'utilisateur : User12 (User12)
Création de l'utilisateur : User13 (User13)
Création de l'utilisateur : User14 (User14)
Création de l'utilisateur : User15 (User15)
Création de l'utilisateur : User16 (User16)
Création de l'utilisateur : User17 (User17)
Création de l'utilisateur : User18 (User18)
Création de l'utilisateur : User19 (User19)
Création de l'utilisateur : User20 (User20)
```

User10	Utilisateur
User11	Utilisateur
User12	Utilisateur
User13	Utilisateur
User14	Utilisateur
User15	Utilisateur
User16	Utilisateur
User17	Utilisateur
User18	Utilisateur
User19	Utilisateur
User2	Utilisateur
User20	Utilisateur
User21	Utilisateur
User22	Utilisateur
User23	Utilisateur
User24	Utilisateur
User25	Utilisateur
User26	Utilisateur
User27	Utilisateur
User28	Utilisateur
User29	Utilisateur
User3	Utilisateur
User30	Utilisateur
User31	Utilisateur
User32	Utilisateur
User33	Utilisateur

## CONCLUSION

L'adoption de PowerShell pour l'ajout d'utilisateurs représente une avancée majeure dans l'administration système, offrant une multitude d'avantages et de gains significatifs. En automatisant cette tâche répétitive, PowerShell permet non seulement d'économiser un temps précieux, mais aussi de réduire les efforts requis, en évitant la saisie manuelle laborieuse des informations utilisateur, une par une. La simplicité et l'efficacité de cette approche libèrent les administrateurs système pour se concentrer sur des tâches plus stratégiques et à valeur ajoutée, tout en garantissant une cohérence et une fiabilité accrues dans la gestion des utilisateurs au sein de l'infrastructure informatique.