Rapport de stage

Table des matières

[1. 📌 Introduction 1](#_Toc201906968)

[2. 🏢 Contexte 2](#_Toc201906969)

[3. 🎯 Objectifs 3](#_Toc201906970)

[4. 🛠️ Actions Réalisées 3](#_Toc201906971)

[🔐 4.1 Script de création des comptes utilisateurs AD 3](#_Toc201906972)

[🧩 4.2 Mise à jour en masse du suffixe UPN 4](#_Toc201906973)

[💼 4.3 Automatisation de la configuration des applications Office 365 5](#_Toc201906974)

[🧹 4.4 Script de suppression automatique des profils locaux inactifs 8](#_Toc201906975)

[🔁 4.5 Redémarrage hebdomadaire des postes 10](#_Toc201906976)

[💽 4.6 Automatisation complète : Windows 11, Wi-Fi et jonction au domaine 10](#_Toc201906977)

[🗂️ 4.7 Inventaire des postes Windows dans l'Active Directory 13](#_Toc201906978)

[🗃️ 4.8 Migration de la base de données GLPI de MariaDB vers MySQL 13](#_Toc201906979)

[5. ✅ Résultats Obtenus 14](#_Toc201906980)

[6. 🎓 Bilan et Compétences Acquises 15](#_Toc201906981)

[Annexe A – Procédure KeePass 15](#_Toc201906982)

[Annexe B – Procédure Mise à niveau WS 11 20](#_Toc201906983)

[Annexe C – Script Création utilisateur 23](#_Toc201906984)

# 📌 Introduction

Durant mon stage au sein de l’entreprise Feron-Vrau, j’ai mené plusieurs actions concrètes dans le but d’optimiser la gestion des utilisateurs dans l’environnement informatique de l’entreprise.

Ces missions avaient pour but de renforcer la sécurité 🔒, de standardiser les procédures 🧷, et de faciliter l’intervention du support informatique 💬. Elles s’inscrivaient dans un contexte d’administration Active Directory et d’intégration d’Office 365 Business Premium.

Parmi les actions réalisées, on peut citer :

* 🔐 Rédaction d’une démarche technique d’import de mots de passe dans KeePass à partir d’un fichier Excel, à destination d’utilisateurs sans appétence informatique.
* 🧩 Adaptation d’un script PowerShell de création de comptes utilisateurs Active Directory, pour générer automatiquement les identifiants (UPN), définir un mot de passe initial et appliquer les bonnes pratiques internes.
* 📧 Modification en masse des UPN des utilisateurs Active Directory afin de garantir une cohérence entre les identifiants locaux et ceux de la messagerie Office 365.
* 🔁 Mise en place d’un redémarrage hebdomadaire automatique des postes utilisateurs via GPO, pour favoriser l’application des mises à jour système.
* 🧹 Suppression automatique des profils utilisateurs inactifs depuis plus de 45 jours, avec génération de logs et déploiement par script planifié.
* 👤 Simplification de la première connexion des utilisateurs, en automatisant la configuration initiale des outils bureautiques Office 365 (Outlook, OneDrive, Teams) dès la première ouverture de session.
* 💽 Mise à jour de Windows 11 via clé USB semi-automatisée
* 🗂️ Inventaire des postes Windows dans l’AD

Ces missions m’ont été l’occasion de mettre en pratique mes compétences en administration système 🖥️, scripting PowerShell ⚙️, stratégies de groupe (GPO) 📚, et rédaction de documentation technique 📝, tout en respectant les contraintes organisationnelles et les profils variés des utilisateurs.

# 🏢 Contexte

L’entreprise utilise un environnement Windows Server avec Active Directory (AD) et Office 365 Business Premium. Le processus initial de mise en service des utilisateurs posait des difficultés opérationnelles en termes de gestion des profils, de configuration des outils collaboratifs (Outlook, Teams, OneDrive) et de sécurité. Par ailleurs, la gestion manuelle des comptes générait des erreurs et prenait beaucoup de temps.

# 🎯 Objectifs

* Simplifier et automatiser la création des comptes utilisateurs avec les bons paramètres (UPN, mot de passe générique).
* Automatiser la gestion des profils utilisateurs, en particulier la suppression des profils locaux inactifs.
* Configurer automatiquement les outils Office 365 à la première connexion.
* Réduire le temps d’intervention du support informatique.
* Optimiser les ressources serveur et postes clients.
* Automatiser la mise à niveau des postes Windows 10 vers Windows 11, via clé USB bootable, en conservant données et paramètres existants.
* Automatiser la connexion au réseau Wi-Fi lors de l’installation d’un poste, sans intervention humaine ni connexion filaire nécessaire.
* Identifier les postes ayant besoin d’un passage avec la clé bootable

# 🛠️ Actions Réalisées

## 🔐 4.1 Script de création des comptes utilisateurs AD

J’ai adapté un [script PowerShell existant](#_Annexe_C_–) pour automatiser la création des comptes utilisateurs dans Active Directory avec :

* Génération automatique du User Principal Name (UPN) au format `@feron-vrau.com`.
* Attribution d’un mot de passe générique sécurisé conforme à la politique interne.
* Forçage de la modification du mot de passe lors de la première connexion.

Bénéfices : Réduction des erreurs manuelles, optimisation du temps de traitement, conformité et sécurité renforcées.

Une image contenant texte, capture d’écran, Police, Bleu électrique

Le contenu généré par l’IA peut être incorrect.Une image contenant texte, Police, capture d’écran

Le contenu généré par l’IA peut être incorrect.



🖼️ \*Figure 4.1 – Interface AD montrant un compte utilisateur créé automatiquement\*

## 🧩 4.2 Mise à jour en masse du suffixe UPN

Pour homogénéiser les identifiants et simplifier la synchronisation des outils Microsoft Office, j’ai développé un script PowerShell permettant de modifier en masse le suffixe UPN des comptes AD de `@feronvrau.local` vers `@feron-vrau.com`.

# Importation du module Active Directory pour utiliser les cmdlets AD comme Get-ADUser et Set-ADUser

Import-Module ActiveDirectory

# Liste des anciens suffixes UPN que l'on souhaite remplacer

$oldSuffixes = @("@FeronVrau.local", "@feronvrau.local")

# Nouveau suffixe UPN à appliquer à la place des anciens

$newSuffix = "@feron-vrau.com"

# Récupération de tous les utilisateurs activés avec leur UserPrincipalName (UPN)

$users = Get-ADUser -Filter {Enabled -eq $true} -Properties UserPrincipalName | Where-Object {

# Mise en minuscule de l’UPN pour normaliser la comparaison

$userUPN = $\_.UserPrincipalName.ToLower()

# Extraction de la partie suffixe du UPN (tout ce qui suit le "@") et vérification s’il est dans la liste des anciens suffixes

$oldSuffixes -contains $userUPN.Substring($userUPN.IndexOf("@"))

}

# Vérifie s’il y a des utilisateurs concernés par le changement

if ($users.Count -eq 0) {

Write-Host "Aucun utilisateur trouvé avec le suffixe ancien." -ForegroundColor Yellow

} else {

Write-Host "Nombre d'utilisateurs à modifier : $($users.Count)" -ForegroundColor Cyan

}

# Boucle sur tous les utilisateurs à modifier

foreach ($user in $users) {

# Stockage de l’UPN actuel

$currentUPN = $user.UserPrincipalName

Write-Host "Traitement : $($user.SamAccountName), UPN actuel : $currentUPN"

try {

# Séparation de l’UPN en deux parties : avant et après le @

$username = $currentUPN.Split("@")[0]

# Construction du nouvel UPN avec le nouveau suffixe

$newUPN = "$username$newSuffix"

# Modification de l’UPN dans Active Directory

Set-ADUser -Identity $user -UserPrincipalName $newUPN

Write-Host "Succès : UPN modifié pour $($user.SamAccountName) -> $newUPN" -ForegroundColor Green

}

catch {

# Gestion des erreurs : affiche l’erreur si Set-ADUser échoue pour un utilisateur

Write-Host "Erreur sur $($user.SamAccountName) : $($\_.Exception.Message)" -ForegroundColor Red

}

}





🖼️ \*Figure 4.2 – Console PowerShell affichant le nombre d’utilisateurs traités\*

## 💼 4.3 Automatisation de la configuration des applications Office 365

J’ai créé des scripts PowerShell exécutés au premier login, qui configurent automatiquement Outlook, Teams et OneDrive, en paramétrant les comptes et en synchronisant les données.

# ========================

# Détection de l'utilisateur connecté (UPN Microsoft 365)

# ========================

# Récupération du User Principal Name (UPN) via la commande whoami (nécessite un environnement connecté à Azure AD)

$upn = (whoami /upn).Trim()

# Si la récupération échoue (par exemple hors ligne), on le reconstitue à partir du nom d’utilisateur local

if (-not $upn) {

$upn = "$env:USERNAME@feron-vrau.com"

}

# Affichage du résultat

Write-Host "UPN détecté : $upn"

# ========================

# Définition des chemins des applications à lancer

# ========================

# Chemin d'installation de OneDrive

$oneDrivePath = "$Env:LOCALAPPDATA\Microsoft\OneDrive\OneDrive.exe"

# Chemin d'Outlook (version Microsoft 365 installée via Click-to-Run)

$outlookPath = "${Env:ProgramFiles}\Microsoft Office\root\Office16\OUTLOOK.EXE"

# Chemin du client Microsoft Teams

$teamsPath = "$Env:LOCALAPPDATA\Microsoft\Teams\current\Teams.exe"

# ========================

# Fonction pour lancer une application si elle existe

# ========================

function Launch-App {

param ([string]$path)

# Vérifie si le fichier exécutable existe

if (Test-Path $path) {

Start-Process $path

} else {

Write-Host "Application introuvable : $path"

}

}

# ========================

# Configuration de OneDrive

# ========================

# Chemin vers le fichier de configuration OneDrive Business

$configFile = "$Env:LOCALAPPDATA\Microsoft\OneDrive\settings\Business1\Business1.ini"

# Si le fichier n'existe pas, c’est que OneDrive n’a pas encore été configuré pour l’utilisateur

if (-Not (Test-Path $configFile)) {

Write-Host "OneDrive : première configuration..."

# Lance OneDrive en mode configuration automatique avec l’UPN détecté

Start-Process $oneDrivePath "/configure\_business:$upn"

# Pause de 10 secondes pour laisser le processus se lancer

Start-Sleep -Seconds 10

} else {

Write-Host "OneDrive déjà configuré."

}

# ========================

# Préparation d’Outlook (profil automatique)

# ========================

# Objectif : forcer Outlook à détecter automatiquement l’adresse e-mail via AutoDiscover (gain de temps pour l'utilisateur)

try {

# Crée la clé de registre si elle n'existe pas déjà

New-Item -Path "HKCU:\Software\Microsoft\Office\16.0\Outlook\AutoDiscover" -Force | Out-Null

# Insère le champ EmailAddress dans le registre avec la valeur UPN pour forcer la détection automatique du compte

Set-ItemProperty -Path "HKCU:\Software\Microsoft\Office\16.0\Outlook\AutoDiscover" -Name "EmailAddress" -Value $upn

# Configure l’option d’identification unique (SSO) dans Outlook

Set-ItemProperty -Path "HKCU:\Software\Microsoft\Office\16.0\Common\Identity" -Name "SignInOptions" -Value 1

Write-Host "Outlook : configuration AutoDiscover forcée."

} catch {

Write-Host "Erreur lors de la configuration d'Outlook : $\_"

}

# ========================

# Lancement des applications utilisateur

# ========================

Write-Host "Lancement d'Outlook..."

Launch-App $outlookPath

Write-Host "Lancement de Teams..."

Launch-App $teamsPath

# ========================

# Message d’accueil graphique (Windows MessageBox)

# ========================

# Chargement de l'assembly .NET pour créer une boîte de dialogue Windows

Add-Type -AssemblyName PresentationFramework

# Affiche un message de bienvenue à l'utilisateur

[System.Windows.MessageBox]::Show("Bienvenue ! Vos applications OneDrive, Outlook et Teams sont prêtes à l'emploi.", "Première connexion")

## 🧹 4.4 Script de suppression automatique des profils locaux inactifs

* Mise en place de stratégies de groupe pour standardiser la gestion des profils itinérants.
* Activation de la suppression automatique des profils locaux inactifs depuis plus de 45 jours.

# Seuil d’inactivité en jours : les profils non utilisés depuis ce nombre de jours seront supprimés

$DaysThreshold = 45

# Liste des noms d'utilisateurs à exclure de la suppression

$ExcludedUsers = @('Administrator', 'Public', 'Default')

# Date et heure actuelle (servira pour le calcul de l’ancienneté)

$Now = Get-Date

# Fichier de log pour tracer les suppressions effectuées

$LogFile = "C:\Logs\ProfilCleanup.log"

# Si le dossier C:\Logs n'existe pas, on le crée pour stocker les logs

if (!(Test-Path "C:\Logs")) {

New-Item -Path "C:\Logs" -ItemType Directory

}

# Récupération de tous les profils utilisateurs présents sur la machine

Get-CimInstance Win32\_UserProfile | Where-Object {

# On ne prend que les profils standards (pas système) et qui ne sont pas chargés (non connectés)

$\_.Special -eq $false -and $\_.Loaded -eq $false

} | ForEach-Object {

# Récupère le nom d'utilisateur à partir du chemin local du profil

$UserName = Split-Path $\_.LocalPath -Leaf

# Date de dernière utilisation du profil

$LastUsed = $\_.LastUseTime

# Calcul de l'ancienneté du profil en jours

$Age = ($Now - $LastUsed).Days

# Si le profil est dans la liste d'exclusion (ex : Administrator), on le saute

if ($ExcludedUsers -contains $UserName) { return }

# Si le profil est plus ancien que le seuil défini, on le supprime

if ($Age -ge $DaysThreshold) {

# Suppression du profil

Remove-CimInstance -InputObject $\_

# Écriture dans le fichier log

Add-Content -Path $LogFile -Value "$(Get-Date) - Profil $UserName supprimé (inactif depuis $Age jours)"

}

}

Une image contenant texte, Police, ligne, capture d’écran

Le contenu généré par l’IA peut être incorrect.

🖼️ \*Figure 4.4 – Tâche créé par le script et mise en place par une GPO\*

Déployé via une GPO qui copie le script localement et crée une tâche planifiée exécutée chaque semaine.  
Une image contenant texte, capture d’écran, affichage, nombre

Le contenu généré par l’IA peut être incorrect.

🖼️ \*Figure 4.4 – GPO appliquée aux postes pour la gestion des profils\*

## 🔁 4.5 Redémarrage hebdomadaire des postes

Déploiement via une GPO configurée pour exécuter un script PowerShell au démarrage. Ce script crée localement une tâche planifiée qui redémarre automatiquement certains postes chaque samedi à 4h du matin, afin de garantir l’application des mises à jour et le bon fonctionnement du système.

Le ciblage est géré depuis un groupe de sécurité Active Directory, ce qui permet de contrôler précisément quelles machines sont concernées, sans modifier le script.

# Script PowerShell : création d’une tâche planifiée pour redémarrage

$Action = New-ScheduledTaskAction -Execute "shutdown.exe" -Argument "/r /f /t 0"

$Trigger = New-ScheduledTaskTrigger -Weekly -DaysOfWeek Saturday -At 4am

$Settings = New-ScheduledTaskSettingsSet -AllowStartIfOnBatteries -DontStopIfGoingOnBatteries -StartWhenAvailable

Register-ScheduledTask -TaskName "RedemarrageHebdo" `

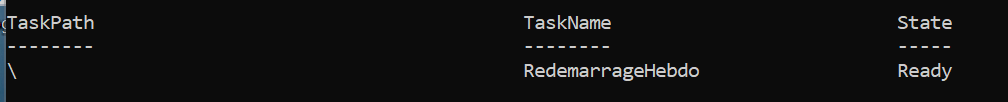
-Action $Action `

-Trigger $Trigger `

-Settings $Settings `

-RunLevel Highest `

-User "SYSTEM"

Déploiement réalisé via une GPO configurée dans Paramètres Windows > Scripts (Démarrage), à l’aide d’un script .bat appelant ce fichier .ps1.  


🖼️ \*Figure 4.5.1 – Résultat du script utiliser en administrateur en test\*  
Une image contenant texte, ligne, Police, nombre

Le contenu généré par l’IA peut être incorrect.

🖼️ \*Figure 4.5.2 – Tâche visible dans le planificateur de tâche\*

## 💽 4.6 Automatisation complète : Windows 11, Wi-Fi et jonction au domaine

🎯 Objectifs

Automatiser intégralement la mise à niveau des postes Windows 10 vers Windows 11, tout en assurant la connexion automatique à un réseau Wi-Fi et la jonction immédiate au domaine Active Directory, sans intervention humaine ni connexion Internet nécessaire.

📋 Étapes de déploiement

1️⃣ Création de la clé USB personnalisée

* Clé bootable créée avec Rufus.
* ISO Windows 11 personnalisé via UUP Dump.
* Intégration du fichier de réponse AutoUnattend.xml à la racine de la clé.

📄 Script XML AutoUnattend.xml (extrait) :

<?xml version="1.0" encoding="utf-8"?>

<unattend xmlns="urn:schemas-microsoft-com:unattend">

<settings pass="windowsPE">

<component name="Microsoft-Windows-Setup">

<ImageInstall>

<OSImage>

<InstallFrom>

<MetaData>

<Key>/IMAGE/INDEX</Key>

<Value>6</Value>

</MetaData>

</InstallFrom>

<WillShowUI>OnError</WillShowUI>

</OSImage>

</ImageInstall>

<UserData>

<AcceptEula>true</AcceptEula>

<FullName>IT</FullName>

<Organization>Entreprise</Organization>

</UserData>

</component>

</settings>

<settings pass="specialize">

<component name="Microsoft-Windows-Shell-Setup">

<TimeZone>Romance Standard Time</TimeZone>

<ComputerName>\*</ComputerName>

<CopyProfile>true</CopyProfile>

</component>

</settings>

<settings pass="oobeSystem">

<component name="Microsoft-Windows-International-Core">

<InputLocale>fr-FR</InputLocale>

<SystemLocale>fr-FR</SystemLocale>

<UILanguage>fr-FR</UILanguage>

<UserLocale>fr-FR</UserLocale>

</component>

<component name="Microsoft-Windows-Shell-Setup">

<OOBE>

<HideEULAPage>true</HideEULAPage>

<HideLocalAccountScreen>true</HideLocalAccountScreen>

<HideOnlineAccountScreens>true</HideOnlineAccountScreens>

<NetworkLocation>Work</NetworkLocation>

<ProtectYourPC>1</ProtectYourPC>

</OOBE>

<RegisteredOrganization>Entreprise</RegisteredOrganization>

<RegisteredOwner>IT</RegisteredOwner>

</component>

</settings>

</unattend>

2️⃣ Automatisation de la connexion Wi-Fi

* Export du profil Wi-Fi préconfiguré :

netsh wlan export profile name="WIFI-SADP" folder="D:\wifi" key=clear

Une image contenant texte, Police, ligne, nombre

Le contenu généré par l’IA peut être incorrect.

* Script PowerShell automatisant l'import et la connexion Wi-Fi :

# connect-wifi.ps1

# Chemin du fichier de profil Wi-Fi (profil XML exporté)

$profilePath = "$PSScriptRoot\wifi\Wi-Fi-NOMDUWIFI.xml" # <-- Anonymisé

# Nom du SSID à connecter

$ssid = "NOMDUWIFI" # <-- Anonymisé

Write-Output "Importation du profil Wi-Fi..."

netsh wlan add profile filename="$profilePath" user=all

Write-Output "Connexion au réseau Wi-Fi..."

netsh wlan connect name="$ssid"

# Attente de quelques secondes, puis test de connectivité

Start-Sleep -Seconds 5

Test-Connection -ComputerName 8.8.8.8 -Count 2 | Out-String

* Exécution automatique via script global (lancer.cmd) :

PowerShell.exe -ExecutionPolicy Bypass -File "%~dp0connect-wifi.ps1"

3️⃣ Jonction automatique au domaine

* Script PowerShell pour jonction automatique au domaine Active Directory dès l'installation :

# Vérifie si l’ordinateur est déjà membre d’un domaine

$domainName = "Domaine.local" # <-- Nom du domaine anonymisé

$currentDomain = (Get-WmiObject Win32\_ComputerSystem).PartOfDomain

if (-not $currentDomain) {

Write-Output "Ordinateur non joint à un domaine. Lancement de la jonction..."

# Identifiants du compte ayant les droits de jonction (admin anonymisé)

$domainUser = "Domaine\\AdminDomaine" # <-- Anonymisé

$password = "MotDePasse" # <-- À ne jamais stocker en clair en production

# Convertit le mot de passe en format sécurisé

$securePassword = ConvertTo-SecureString $password -AsPlainText -Force

$credential = New-Object System.Management.Automation.PSCredential($domainUser, $securePassword)

# Ajoute l’ordinateur au domaine et redémarre

Add-Computer -DomainName $domainName -Credential $credential -Force -Restart

} else {

Write-Output "Déjà membre d’un domaine. Aucune action effectuée."

}

## 🗂️ **4.7** Inventaire des postes Windows dans l'Active Directory

Avant de planifier la mise à niveau vers Windows 11, j’ai réalisé un inventaire des postes présents dans l’Active Directory afin d’identifier ceux encore sous Windows 10.

Le script PowerShell suivant permet de récupérer et de filtrer les machines selon leur système d’exploitation :  
**Powershell**

# Récupère tous les ordinateurs avec le système d’exploitation renseigné

Get-ADComputer -Filter \* -Properties OperatingSystem |

# Filtre les machines dont l’OS contient "Windows 10"

Where-Object { $\_.OperatingSystem -like "\*Windows 10\*" } |

Ce fichier CSV généré m’a permis de cibler avec précision les postes à migrer, en excluant ceux déjà à jour sous Windows 11.

## 🗃️ 4.8 Migration de la base de données GLPI de MariaDB vers MySQL

Dans le cadre de l’évolution de notre environnement GLPI, j’ai réalisé une migration de la base de données de **MariaDB** vers **MySQL**.  
L’objectif principal était d’améliorer la compatibilité avec les futures versions de GLPI et de suivre les recommandations officielles de la communauté GLPI.

Étapes de la migration :

Le processus s’est déroulé en plusieurs phases :

📌 Sauvegarde de la base existante sous MariaDB :

mysqldump -u root -p glpi > glpi\_backup.sql

👉 Cette commande a permis de sauvegarder l’ensemble de la base glpi dans un fichier SQL.

📌 Installation et préparation de MySQL :

* Installation de MySQL Server 8.x sur le nouveau serveur.
* Création manuelle de la nouvelle base de données avec la bonne collation UTF-8 :

CREATE DATABASE glpi CHARACTER SET utf8mb4 COLLATE utf8mb4\_general\_ci;

📌 Restauration de la base dans MySQL :

mysql -u root -p glpi < glpi\_backup.sql

👉 Cette étape a permis de réimporter toutes les données de GLPI dans le nouveau serveur MySQL.

📌 Reconfiguration de GLPI :

J’ai ensuite modifié le fichier de configuration de GLPI (config\_db.php) afin qu’il pointe vers le nouveau serveur MySQL.

php

define('DB\_TYPE', 'mysqli');

define('DB\_HOST', 'localhost');

define('DB\_USER', 'glpi\_user');

define('DB\_PASSWORD', 'password');

define('DB\_NAME', 'glpi');

✅ Remarque : Les identifiants MySQL ont été adaptés pour correspondre aux nouveaux accès.

# ✅ Résultats Obtenus

* Tous les comptes actifs ont vu leur UPN mis à jour avec succès.
* Diminution de l'ordre de 50 % du temps de mise en service des utilisateurs.
* Diminution des appels au support pour la configuration initiale.
* Meilleure gestion des profils et optimisation de l’espace disque.
* Configuration uniforme et sécurisée des outils Office 365.
* Mise à jour du parc de Windows 10 vers Windows 11 et mise à jour de sécurité effectuées.
* Installation automatisée de Windows 11 depuis clé USB.
* Connexion Wi-Fi établie automatiquement sans câble Ethernet.
* Jonction automatique au domaine garantissant sécurité et conformité.

# 🎓 Bilan et Compétences Acquises

**Automatisation & scripting :**  
PowerShell avancé ([création d'utilisateurs](#_🔐_4.1_Script), [modification UPN](#_🧩_4.2_Mise), [nettoyage de profils](#_🧹_4.4_Script), [redémarrage planifié](#_🔁_4.5_Redémarrage), [jonction domaine et Wi-Fi](#_💽_4.6_Automatisation), [inventaire AD](#_🗂️_4.7_Inventaire)), [scripts planifiés](#_🔁_4.5_Redémarrage), [traitement de fichiers XML](#_💽_4.6_Automatisation) et [CSV](#_🗂️_4.7_Inventaire).

**Systèmes & déploiement :**  
Réalisation du déploiement automatisé de Windows 11 à l’aide d’un média bootable personnalisé ([clé USB avec fichier de réponse](#_Annexe_B_–)), puis migration complète de la base de données GLPI depuis [MariaDB vers MySQL](#_🗃️_4.8_Migration), avec sauvegarde préalable, restauration sur le nouveau serveur, et contrôles de bon fonctionnement après la migration.

**Réseau & sécurité :**  
Connexion [Wi-Fi automatisée](#_💽_4.6_Automatisation), [jonction](#_💽_4.6_Automatisation) sécurisée à Active Directory, manipulation de profils Wi-Fi avec netsh, suppression de fichiers sensibles.

**Active Directory & GPO :**  
[Création de comptes AD](#_🔐_4.1_Script), [gestion de suffixes UPN](#_🧩_4.2_Mise), [inventaire automatisé des OS](#_🗂️_4.7_Inventaire), compréhension des OU et stratégies d’intégration.

**Suite Office 365 :**  
[Installation silencieuse](#_💼_4.3_Automatisation) sur plusieurs postes, intégration à l’environnement Microsoft.

**Méthodologie & documentation :**  
Structuration d’une démarche technique, rédaction de procédures([1](#_Annexe_A_–), [2](#_Annexe_B_–), [3](#_Annexe_C_–)), captures d’écran explicatives, annexe complète.

# Annexe A – Procédure KeePass

Procédure utilisation KeePass :

1. **Tout d’abord lancez KeePass :**   
   Une image contenant logo, Police, symbole, Graphique

   Le contenu généré par l’IA peut être incorrect.

Faites « File » puis « New » ou « Fichier » puis « Nouveau ».  


Stockez le fichier dans le Sharepoint commun à votre service si besoin ou s’il est personnel dans votre OneDrive.   
Une image contenant texte, capture d’écran, logiciel, affichage

Le contenu généré par l’IA peut être incorrect.

Ensuite KeePass va vous demander un « Master password » c’est un mot de passe qui sera nécessaire pour l’ouverture des mots de passe.  
Il est recommandé d’utiliser 12 caractères, 1 caractère spécial, un chiffre, une minuscule ainsi qu’une majuscule.  
Une image contenant texte, capture d’écran, logiciel, Page web

Le contenu généré par l’IA peut être incorrect.

Validez puis indiquez le nom de votre fichier.  
Une image contenant texte, Appareils électroniques, capture d’écran, affichage

Le contenu généré par l’IA peut être incorrect.



Cliquez sur « Passer ou Skip » quand on demande pour l’imprimante.

1. **Préparer votre fichier Excel :**

📂 Ouvrir le fichier Excel où sont stockés vos mots de passe.   
✔ Vérifiez que chaque colonne correspond à une information importante et est exactement renommé comme tel :

* Nom du site ou du compte (ex : "Gmail") ou prénom et nom de l’utilisateur,
* Nom d'utilisateur (ex : "mon.email@gmail.com"),
* Mot de passe (ex : "abc123").

Une image contenant texte, capture d’écran, Police, nombre

Le contenu généré par l’IA peut être incorrect.

1. **Convertir le ficher en format compatible avec KeePass**

💡 KeePass ne comprend pas directement Excel, mais il accepte un format spécial appelé CSV. Pour convertir votre fichier :

* Cliquez sur "Fichier" dans Excel,
* Sélectionnez "Enregistrer sous",
* Choisissez le format "CSV (séparateur : point-virgule ou virgule)".
* Enregistrez le fichier sur votre ordinateur.  
  Une image contenant texte, capture d’écran, logiciel, Police

  Le contenu généré par l’IA peut être incorrect.

Vous allez ensuite ouvrir le fichier avec WordPad en faisant clic droit sur le fichier puis « Ouvrir avec » puis « WordPad ».

Quand le fichier sera ouvert vous allez cliquer sur « Remplacer » :   
Une image contenant texte, capture d’écran, affichage, nombre

Le contenu généré par l’IA peut être incorrect.

Ce qui fera apparaître cette fenêtre, vous allez ainsi mettre dans « Rechercher » un « ; » et dans « Remplacer par » une « , » et finir par Remplacer tout. Vous allez maintenant enregistrer le fichier et le fermer.

1. **Retour sur KeePass**

* Cliquez sur "Fichier" > "Importer".
* Choisissez "CSV Importer" et sélectionnez le fichier CSV que vous avez créé. Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

  Le contenu généré par l’IA peut être incorrect.
* Prévisualisez, votre fichier devrait apparaître comme cela :
* Validez et enregistrez ! ✅

Et voilà vous avez maintenant terminer d’importer votre fichier.

# Annexe B – Procédure Mise à niveau WS 11

**📌 Objectif**

Mettre à jour un poste Windows 10 vers Windows 11 **en local**, **sans connexion Internet**, tout en conservant les données et l’environnement de travail.  
L’image ISO utilisée inclut **plusieurs éditions de Windows** :

* Windows 11 **Professionnel**
* Windows 11 **Entreprise**

🎯 Cette procédure **n’intègre pas** de jonction au domaine, de configuration Wi-Fi automatique, ni de fichier AutoUnattend.xml.  
Elle reste **semi-automatisée**, sans effacement de données.

**📋 Étapes de mise en œuvre**

**1️⃣ Téléchargement de l’ISO**

* Aller sur le site [UUP Dump](https://uupdump.net)
* Rechercher : **Windows 11 23H2**
* Choisir : la dernière version disponible exemple : « [Windows 11, version 23H2 (22631.5548) amd64](https://uupdump.net/selectlang.php?id=c2151bd2-59f9-4a1a-8d49-49c53a9498fa) »
* Cochez les options suivantes :
  + ✅ Include updates
  + ✅ Run component cleanup
  + ✅ Integrate .NET Framework 3.5
  + ✅ Use solid (ESD) compression
* Télécharger le pack ZIP et l’extraire
* Exécuter le script uup\_download\_windows.cmd en **tant qu’administrateur**
* Une image ISO est générée, incluant les éditions Pro et Enterprise

Une image contenant texte, capture d’écran, logiciel, Site web

Le contenu généré par l’IA peut être incorrect.

**2️⃣ Préparation de la clé USB**

* Formater une **clé USB de 16 Go minimum** en **NTFS**
* Monter l’image ISO nouvellement créée
* Copier **l’ensemble des fichiers du disque ISO monté** vers la clé USB
* Ajouter un répertoire : « X:\sources\$OEM$\$$\Setup\Scripts »
* Dans ce dossier ajouter les mises à jours de sécurité depuis le [Catalogue Microsoft Update](https://catalog.update.microsoft.com/home.aspx)
* Ajouter 2 scripts, SetupComplete.cmd qui servira à installer les mises à jour automatiquement :

@echo off

:: SetupComplete.cmd – Script post-installation automatique

:: Exécuté automatiquement à la fin de l'installation de Windows

set LOGDIR=%WINDIR%\Setup\Scripts\logs

set LOGFILE=%LOGDIR%\install.log

if not exist "%LOGDIR%" mkdir "%LOGDIR%"

echo ------------------------------------------ >> "%LOGFILE%"

echo Début SetupComplete.cmd - %DATE% %TIME% >> "%LOGFILE%"

if exist "%WINDIR%\Setup\Scripts\INSTALL-MAJ-W11.ps1" (

echo Lancement de INSTALL-MAJ-W11.ps1... >> "%LOGFILE%"

powershell.exe -ExecutionPolicy Bypass -File "%WINDIR%\Setup\Scripts\INSTALL-MAJ-W11.ps1" >> "%LOGFILE%" 2>&1

) else (

echo ❌ INSTALL-MAJ-W11.ps1 introuvable. >> "%LOGFILE%"

)

echo Fin SetupComplete.cmd - %DATE% %TIME% >> "%LOGFILE%"

echo ------------------------------------------ >> "%LOGFILE%"

exit

Et INSTALL-MAJ-W11.ps1 qui servira à ouvrir et lancer les fichiers exe ainsi que msu pour les mises à jours Windows :

# INSTALL-MAJ-W11.ps1

# ⚙️ Script de post-installation Windows 11 : Wi-Fi, Domaine, MAJ, .NET

$scriptDir = Split-Path -Parent $MyInvocation.MyCommand.Path

$logFolder = Join-Path $scriptDir "logs"

if (!(Test-Path $logFolder)) { New-Item -Path $logFolder -ItemType Directory | Out-Null }

# 🟢 Connexion Wi-Fi

$wifiScript = "C:\\connect-wifi.ps1"

if (Test-Path $wifiScript) {

Write-Host "📶 Connexion Wi-Fi..."

powershell.exe -ExecutionPolicy Bypass -File $wifiScript

} else {

Write-Warning "⚠️ Script Wi-Fi manquant : $wifiScript"

}

# 🏢 Jonction au domaine

$domainScript = "C:\\join-domain.ps1"

if (Test-Path $domainScript) {

Write-Host "🌐 Jonction domaine..."

powershell.exe -ExecutionPolicy Bypass -File $domainScript

} else {

Write-Warning "⚠️ Script de jonction domaine manquant : $domainScript"

}

# 🔒 Installation des MAJ Windows (.msu)

$msuUpdates = Get-ChildItem "$scriptDir" -Filter \*.msu

foreach ($update in $msuUpdates) {

Write-Host "🔧 Installation de $($update.Name)..."

Start-Process "wusa.exe" -ArgumentList "`"$($update.FullName)`" /quiet /norestart" -Wait

}

**3️⃣ Mise à niveau sur le poste cible**

* Insérer la clé USB dans le poste Windows 10 à mettre à jour
* Ouvrir la clé USB via l’explorateur de fichiers
* Lancer setup.exe en tant qu’administrateur
* Suivre les instructions :
  + ✅ Choisir **de conserver les fichiers et applications**
  + ✅ Sélectionner **Windows 11 Professionnel** ou **Windows 11 Enterprise**

# Annexe C – Script Création utilisateur

# Définition de l’unité d’organisation (OU) par défaut où l’utilisateur sera créé

$OUUser = 'OU=Utilisateurs,OU=Entreprise,DC=mondomaine,DC=local'

$DomainUserPrincipalName = '@entreprise.com'

# --- Étape 1 : Saisie des informations de l'utilisateur ---

# Demande du prénom de l’utilisateur et validation (lettres et éventuellement un tiret)

$Prenom\_User = Read-Host "Veuillez entrer le prénom de l'utilisateur"

if ($Prenom\_User -notmatch "^[a-zA-Z]\*-?[a-zA-Z]\*$") {

do {

Write-Host "Erreur : seuls les tirets sont autorisés comme caractères spéciaux." -ForegroundColor Red

$Prenom\_User = Read-Host "Veuillez entrer le prénom de l'utilisateur"

} while ($Prenom\_User -notmatch "^[a-zA-Z]\*-?[a-zA-Z]\*$")

}

# Demande du nom de famille et validation (lettres et éventuellement un espace)

$NOM\_User = Read-Host "Veuillez entrer le nom de l'utilisateur"

if ($NOM\_User -notmatch "^[a-zA-Z]\* ?[a-zA-Z]\*$") {

do {

Write-Host "Erreur : seuls les espaces sont autorisés comme caractères spéciaux." -ForegroundColor Red

$NOM\_User = Read-Host "Veuillez entrer le nom de l'utilisateur"

} while ($NOM\_User -notmatch "^[a-zA-Z]\* ?[a-zA-Z]\*$")

}

# Mise en forme du prénom (majuscules pour premières lettres, gestion des prénoms composés)

if ($Prenom\_User -match '-') {

$tiret = $Prenom\_User.IndexOf('-')

$Prenom\_User = ($Prenom\_User.Substring(0,1).ToUpper()) +

($Prenom\_User.Substring(1,$tiret-1).ToLower()) + '-' +

($Prenom\_User.Substring($tiret+1,1).ToUpper()) +

($Prenom\_User.Substring($tiret+2).ToLower())

} else {

$Prenom\_User = $Prenom\_User.Substring(0,1).ToUpper() + $Prenom\_User.Substring(1).ToLower()

}

# Nom en majuscules

$NOM\_User = $NOM\_User.ToUpper()

# Demande de création d'une boîte mail

do {

$Boite\_mail = Read-Host "Faut-il créer une adresse mail pour l'utilisateur ? (o/n)"

} while (($Boite\_mail -ne 'o') -and ($Boite\_mail -ne 'n'))

# --- Étape 2 : Génération du login utilisateur ---

# Mot de passe générique (à sécuriser idéalement)

$MotDePasseGenerique = "MotDePasse123!"

$SecurePassword = ConvertTo-SecureString -String $MotDePasseGenerique -AsPlainText -Force

# Création du login à partir du prénom et du nom (max 20 caractères)

if ($Prenom\_User -match '-') {

$prenom\_user\_login = ($Prenom\_User.Substring(0,1) + $Prenom\_User.Substring($Prenom\_User.IndexOf('-')+1,1)).ToLower()

} else {

$prenom\_user\_login = $Prenom\_User.ToLower()

}

$nom\_user\_login = $NOM\_User.Replace(' ', '').ToLower()

$login\_windows = "$prenom\_user\_login.$nom\_user\_login"

# Si le login dépasse 20 caractères, demande de modification manuelle

if ($login\_windows.Length -gt 20) {

do {

Write-Host "Le login dépasse 20 caractères, veuillez le raccourcir." -ForegroundColor Red

$login\_windows = Read-Host "Entrez un login plus court (ex : p.nomcourt)"

} while ($login\_windows.Length -gt 20)

}

Clear-Host

# --- Étape 3 : Vérification de l'existence de l'utilisateur ---

if (Get-ADUser -Filter \* | Where { $\_.Surname -eq $NOM\_User -and $\_.GivenName -eq $Prenom\_User }) {

Write-Host "Un utilisateur avec ce nom et prénom existe déjà. Abandon du script." -ForegroundColor Red

Start-Sleep 6

exit

} else {

# Adresse mail si demandée

if ($Boite\_mail -eq 'o') {

$mail\_user = "$($Prenom\_User.ToLower()).$($nom\_user\_login)@entreprise.com"

} else {

$mail\_user = $null

}

$Name\_ADObject = "$Prenom\_User $NOM\_User"

}

# --- Étape 4 : Récapitulatif à l’utilisateur ---

Write-Host "`nCréation de l'utilisateur suivant :"

Write-Host "Nom : $NOM\_User"

Write-Host "Prénom : $Prenom\_User"

Write-Host "Login Windows : $login\_windows" -ForegroundColor Green

if ($mail\_user) {

Write-Host "Adresse Mail : $mail\_user" -ForegroundColor Green

} else {

Write-Host "Pas de boîte mail créée."

}

Write-Host "Mot de passe temporaire : $MotDePasseGenerique" -ForegroundColor Red

# --- Étape 5 : Confirmation de la création ---

$OK\_PourCreerUser = Read-Host 'Confirmer la création de l’utilisateur ? (o/n)'

if ($OK\_PourCreerUser -eq 'o') {

$UserPrincipalName = "$login\_windows$DomainUserPrincipalName"

$DisplayName = "$NOM\_User $Prenom\_User"

# Création de l'utilisateur dans l'Active Directory

New-ADUser -Name $Name\_ADObject `

-Enabled $true `

-UserPrincipalName $UserPrincipalName `

-GivenName $Prenom\_User `

-Surname $NOM\_User `

-DisplayName $DisplayName `

-AccountPassword $SecurePassword `

-SamAccountName $login\_windows `

-ChangePasswordAtLogon $true `

-Path $OUUser

# Ajout de l’adresse mail dans l’AD si nécessaire

if ($mail\_user) {

Set-ADUser $login\_windows -EmailAddress $mail\_user

Set-ADUser $login\_windows -Add @{"extensionAttribute1"="Sync"}

}

Write-Host "`nL’utilisateur a été créé. Il devra changer son mot de passe à la première connexion." -ForegroundColor Red

Write-Host "Synchronisation avec Azure AD : jusqu'à 30 minutes."

Start-Sleep 15

exit

} else {

Write-Host "Création annulée. Fermeture dans 15 secondes..."

Start-Sleep 15

exit

}