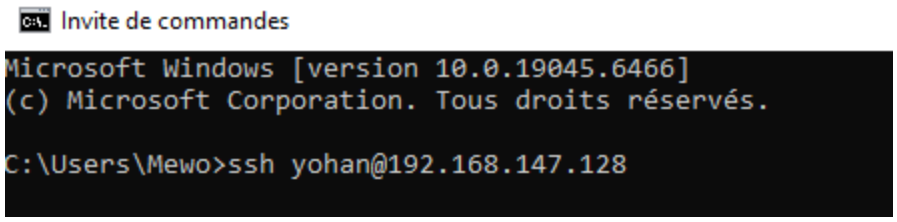
2025.11.26 - Bloc 2 - Architecture complète GLPI (LinusApacheMysqlPhp)

(A FAIRE PETIT MEMENTO SUR CE QU EST LES APP INSTALLER : APACHE2 MARIADB ETC!!!!)

1 GLPI LAMP

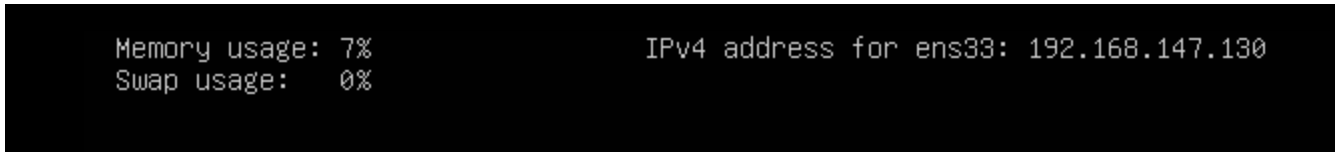
Pour copier/coller des commandes dans ta VM mets en place un SSH sur ta machine physique (CMD)



(ssh, "nom"@"IP")

Et maintenant tu peux utiliser ton cmd de ta machine physique pour mettre ton code sur ta VM et ainsi pouvoir utiliser les copier/coller en faisant clique droit !

Premièrement, installer un Ubuntu Server en ligne de commande.



Premièrement mettre à jour le système

```
sudo apt update && sudo apt upgrade -y
```

*La commande sudo te donne les droits d'administrateur (SuperUser DO). Si le terminal te demande un mot de passe, tape-le. Attention, rien ne s'affiche quand tu tapes le mot de passe (ni étoiles, ni points), c'est une sécurité normale sous Linux. Appuie simplement sur Entrée.***

Étape 1 : Installation du serveur Web et de la Base de Données (LAMP)

GLPI fonctionne grâce à un empilement de technologies appelé LAMP (Linux, Apache, MySQL/MariaDB, PHP).

1.1 Installer Apache (Le serveur Web)

C'est lui qui va "servir" les pages de GLPI à ton navigateur.

```
sudo apt install apache2 -y
```

Vérification : Ouvre Firefox et tape http://localhost. Si tu vois une page "Apache2 Default Page", c'est gagné ! (Tape "ip a" pour avoir l'IP de ton Ubuntu Server 192.168.147.128)

1.2 Installer MariaDB (la base de données)

```
sudo apt install mariadb-server -y
```

1.3 Installer PHP et ses extensions (Le moteur)

C'est l'étape où beaucoup d'étudiants bloquent. GLPI a besoin de briques spécifiques (extensions) pour fonctionner. Sur Ubuntu 24.04, nous utilisons PHP 8.3 par défaut.
Copie cette commande en une seule fois :

```
sudo apt install php php-xml php-common php-json php-mysql php-mbstring php-curl php-gd php-intl php-zip php-bz2 php-imap php-apcu php-ldap -y
```

Erreur fréquente : L'oubli d'extensions
Si tu oublies php-dom ou php-gd, l'installateur web de GLPI te bloquera plus tard avec des messages d'erreur rouges. Cette liste ci-dessus couvre les besoins essentiels de GLPI 11.

Étape 2 : Configuration de la base de données

Nous devons créer une "boîte" (la base de données) et donner la clé à un utilisateur spécifique. Ne jamais utiliser l'utilisateur root de la base de données pour une application web en production !

Connecte-toi à la console SQL :

```
sudo mariadb
```

Une fois que ton invite de commande change pour MariaDB [(none)]>, tape les lignes suivantes une par une (n'oublie pas les points-virgules ; à la fin !) :

1. Créer la base de données :

```
CREATE DATABASE glpidb;
```

2. Créer l'utilisateur (remplace 'ton_mot_de_passe' par un mot de passe solide) :

```
CREATE USER 'glpi_user'@'localhost' IDENTIFIED BY 'ton_mot_de_passe';
```

3. Donner les droits :

```
GRANT ALL PRIVILEGES ON glpidb.* TO 'glpi_user'@'localhost';
```

4. Valider et quitter :

```
FLUSH PRIVILEGES;
EXIT;
```

Étape 3 : Installation des fichiers GLPI

Maintenant que l'infrastructure est prête, téléchargeons le logiciel.

3.1 Télécharger GLPI

Va dans le dossier temporaire pour télécharger proprement :

```
cd /tmp
```

"tmp = dossier temporaire"

```
wget https://github.com/glpi-project/glpi/releases/download/11.0.2/glpi-
11.0.2.tgz
```

(Note : La version 11.0.2 est un exemple, vérifie toujours la dernière version stable sur le site officiel).

```
HTTP request sent, awaiting response... 200 OK
Length: 88212945 (84M) [application/octet-stream]
Saving to: ‘glpi-11.0.2.tgz’

glpi-11.0.2.tgz             75%[=========================>
```


3.2 Décompresser et déplacer

Reste bien sur :

```
cd /tmp
```

et rentre

```
tar -xzvf glpi-11.0.2.tgz
sudo mv glpi /var/www/html/
```

Le dossier et maintenant déplacer dans ce dossier : /var/www/html/

Étape 4 : La gestion des permissions

C'est ici que 80% des débutants rencontrent des problèmes.

Par défaut, les fichiers appartiennent à ton utilisateur (toi). Mais c'est le serveur web (Apache) qui doit pouvoir lire et écrire dedans. Sous Ubuntu, l'utilisateur d'Apache s'appelle www-data.

Si tu ne fais pas cette étape, tu auras des erreurs "Access Denied" ou "Impossible d'écrire dans les dossiers".

```
sudo chown -R www-data:www-data /var/www/html/glpi
sudo chmod -R 755 /var/www/html/glpi
```

• chown : Change Owner (Change le propriétaire).
• -R : Récursif (pour tous les fichiers et dossiers contenus dedans).
• www-data:www-data : L'utilisateur et le groupe d'Apache.

Étape 5 : Configurer Apache (VirtualHost)

Avec GLPI 11, il est impératif de dire à Apache de viser le dossier /public. Nous allons créer un fichier de configuration.

1. Crée un nouveau fichier de config :

```
sudo nano /etc/apache2/sites-available/glpi.conf
```

2. Colle ce contenu dedans (c'est la carte d'identité de ton site) :

```
<VirtualHost *:80>
ServerName localhost
# C'est ici que ça change : on pointe vers /public
DocumentRoot /var/www/html/glpi/public
<Directory /var/www/html/glpi/public>
Require all granted
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php [QSA,L]
</Directory>
ErrorLog ${APACHE_LOG_DIR}/glpi_error.log
CustomLog ${APACHE_LOG_DIR}/glpi_access.log combined
</VirtualHost>
```

(Astuce : Pour enregistrer sous Nano : Ctrl+O puis Entrée, et pour quitter Ctrl+X).

Enregistrez et quittez.

3. Active ce nouveau site et le module de réécriture d'URL :

```
sudo a2ensite glpi.conf
sudo a2enmod rewrite
```

Apache possède déjà un site par défaut (000-default.conf) qui écoute aussi sur le port 80. Si tu ne désactives pas le site par défaut, il est possible qu'Apache continue d'afficher la page "It Works" au lieu de ton GLPI.

```
sudo a2dissite 000-default.conf
```

(Cela désactive la configuration par défaut pour laisser la place à la tienne).

```
sudo systemctl restart apache2
```

Étape 6 : Finalisation via le navigateur

C'est le moment de vérité !

1. Ouvre ton navigateur web.
2. Tape l'adresse : http://localhost (IP du serveur) "http://192.168.147.128/"
3. Tu devrais voir l'assistant d'installation de GLPI.

La procédure à l'écran :

1. Choisis la langue (Français).
2. Accepte la licence.
3. Choisis "Installer".
4. Vérification de l'environnement : Si tu as bien suivi l'étape 1.3, tout devrait être vert (ou orange pour des avertissements mineurs). Si tu as du rouge, note l'extension manquante et installe-la (sudo apt install php-nom_extension).

   Requis **bcmath extension**
   Requis pour la prise en charge des QR codes.
   L'extension bcmath est absente

5. Connexion BDD :
   o Serveur SQL : localhost

o Utilisateur : glpi_user
o Mot de passe : celui que tu as défini à l'étape 2.

6. Sélectionne la base glpidb.

Une fois terminé, GLPI te donnera les identifiants par défaut.
• Identifiant : glpi
• Mot de passe : glpi

Erreur

Guide de dépannage pour les erreurs fréquentes

Voici les petits cailloux dans la chaussure que tu pourrais rencontrer :

"Je vois une page blanche ou le code PHP s'affiche en texte brut"

• Cause : Apache ne "comprend" pas le PHP. Il manque souvent le module ou il n'est pas activé.
• Solution : Assure-toi d'avoir installé libapache2-mod-php et redémarre Apache :

```
sudo systemctl restart apache2
```

"Erreur : Impossible d'accéder à la base de données"

• Cause : Souvent une faute de frappe dans le mot de passe ou le nom d'utilisateur lors de l'assistant web.
• Solution : Si tu es bloqué, tu peux supprimer l'utilisateur et le refaire dans MariaDB, ou vérifier que le service tourne :

```
sudo systemctl status mariadb
```

"Alerte de sécurité : Dossier install non supprimé"

• Cause : Une fois installé, GLPI te demande de supprimer le dossier d'installation pour éviter qu'un pirate ne réinstalle ton site.
• Solution : Retourne dans le terminal :

```
sudo rm -rf /var/www/html/glpi/install/
```

Sécuriser son serveur GLPI (HTTPS & Pare-feu)

🖊 Introduction : pourquoi cette étape est critique ?

Dans le monde professionnel, ton serveur actuel est dangereux.

Pourquoi ? Parce que le protocole HTTP (HyperText Transfer Protocol) fait circuler les informations "en clair" sur le réseau.

Imagine la situation :

Tu es l'administrateur. Tu te connectes à GLPI avec ton mot de passe super complexe.

Si un pirate (ou un curieux) est connecté sur le même réseau que toi et lance un logiciel d'écoute comme Wireshark, il verra passer ton mot de passe tel quel, comme s'il lisait une carte postale sans enveloppe.

Notre mission du jour :

1. Chiffrer les échanges (HTTPS) pour que les données deviennent illisibles pour les pirates.
2. Filtrer les accès (Pare-feu) pour n'ouvrir que les portes strictement nécessaires.

Partie 1 : Comprendre le HTTPS (La théorie simple)

Avant de taper des commandes, comprenons le concept.

• HTTP (Port 80) : C'est comme envoyer une carte postale. Le facteur (le réseau) et tous ceux qui manipulent le courrier peuvent lire le message écrit au dos.
• HTTPS (Port 443) : C'est comme envoyer une lettre dans un coffre-fort blindé. Seul le destinataire (le serveur) possède la clé pour ouvrir le coffre et lire le message.

Pour que cela fonctionne, nous avons besoin de deux choses :

1. Une Clé privée (conservée secrètement sur le serveur).
2. Un Certificat public (distribué aux visiteurs pour qu'ils puissent chiffrer les messages qu'ils t'envoient).

Partie 2 : Mise en place du HTTPS (Certificat Auto-signé)

Dans une vraie entreprise, on achète un certificat validé par une autorité reconnue (comme Let's Encrypt ou Digicert). Pour ce TP, nous allons créer notre propre certificat : on dit qu'il est auto-signé.
C'est un peu comme fabriquer sa propre carte d'identité : elle est techniquement valide pour

chiffrer, mais la police (le navigateur web) te dira qu'elle ne connait pas l'autorité qui l'a délivrée. C'est normal !

Étape 2.1 : Activer le module SSL d'Apache

Apache possède un module pour gérer le chiffrement, mais il n'est pas activé par défaut.

1. Ouvre ton terminal (Ctrl+Alt+T).
2. Active le module SSL :

```
sudo a2enmod ssl
```

3. Pour que le changement soit pris en compte, redémarre Apache :

```
sudo systemctl restart apache2
```

Étape 2.2 : Générer les clés et le certificat

Nous allons utiliser l'outil OpenSSL (« le couteau suisse de la cryptographie »).

Crée un dossier pour ranger proprement tes certificats :

```
sudo mkdir /etc/apache2/ssl
```

Génère le certificat et la clé en une seule commande (copie-colle tout ceci) :

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/apache2/ssl/glpi.key -out /etc/apache2/ssl/glpi.crt
```

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:Moselle
Locality Name (eg, city) []:Metz
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Mewo
Organizational Unit Name (eg, section) []:Mewo
Common Name (e.g. server FQDN or YOUR name) []:
Email Address []:
```

Note

Déchiffrons cette commande :

• req -x509 : On veut créer un certificat au standard X.509.
• -days 365 : Le certificat sera valide 1 an.

• -newkey rsa:2048 : On crée une clé de chiffrement RSA très robuste (2048 bits).
• -keyout et -out : Où ranger la clé (le secret) et le certificat (le public).

L'outil va te poser des questions (Pays, Ville, etc.). Comme c'est un TP local, tu peux appuyer sur Entrée pour tout passer, ou remplir "FR" pour le pays.

Étape 2.3 : Configurer le VirtualHost Apache

Rappelle-toi, dans le TP précédent, nous avons configuré GLPI pour répondre sur le port 801. Nous devons maintenant dire à Apache d'écouter aussi sur le port 443 (le port standard du HTTPS) et d'utiliser nos clés.

1. Crée un nouveau fichier de configuration pour la version sécurisée :

```
sudo nano /etc/apache2/sites-available/glpi-ssl.conf
```

2. Colle le contenu suivant (c'est une adaptation de ta configuration précédente, avec la couche sécurité en plus) :

```
<VirtualHost *:443>
ServerName localhost
DocumentRoot /var/www/html/glpi/public
SSLEngine on
SSLCertificateFile /etc/apache2/ssl/glpi.crt
SSLCertificateKeyFile /etc/apache2/ssl/glpi.key
<Directory /var/www/html/glpi/public>
Require all granted
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php [QSA,L]
</Directory>
ErrorLog ${APACHE_LOG_DIR}/glpi_ssl_error.log
CustomLog ${APACHE_LOG_DIR}/glpi_ssl_access.log combined
</VirtualHost>
```

Note

Ctrl O pour enregistrer
Ctrl X pour quitter

Étape 2.4 : Activer le site et tester

1. Active ce nouveau site sécurisé :

```
sudo a2ensite glpi-ssl.conf
```

2. Recharge Apache :

```
sudo systemctl reload apache2
```

Le moment de vérité :

Ouvre Firefox et tape : https://localhost (Note bien le s à la fin de http). 192.168.147.128

Alerte securite

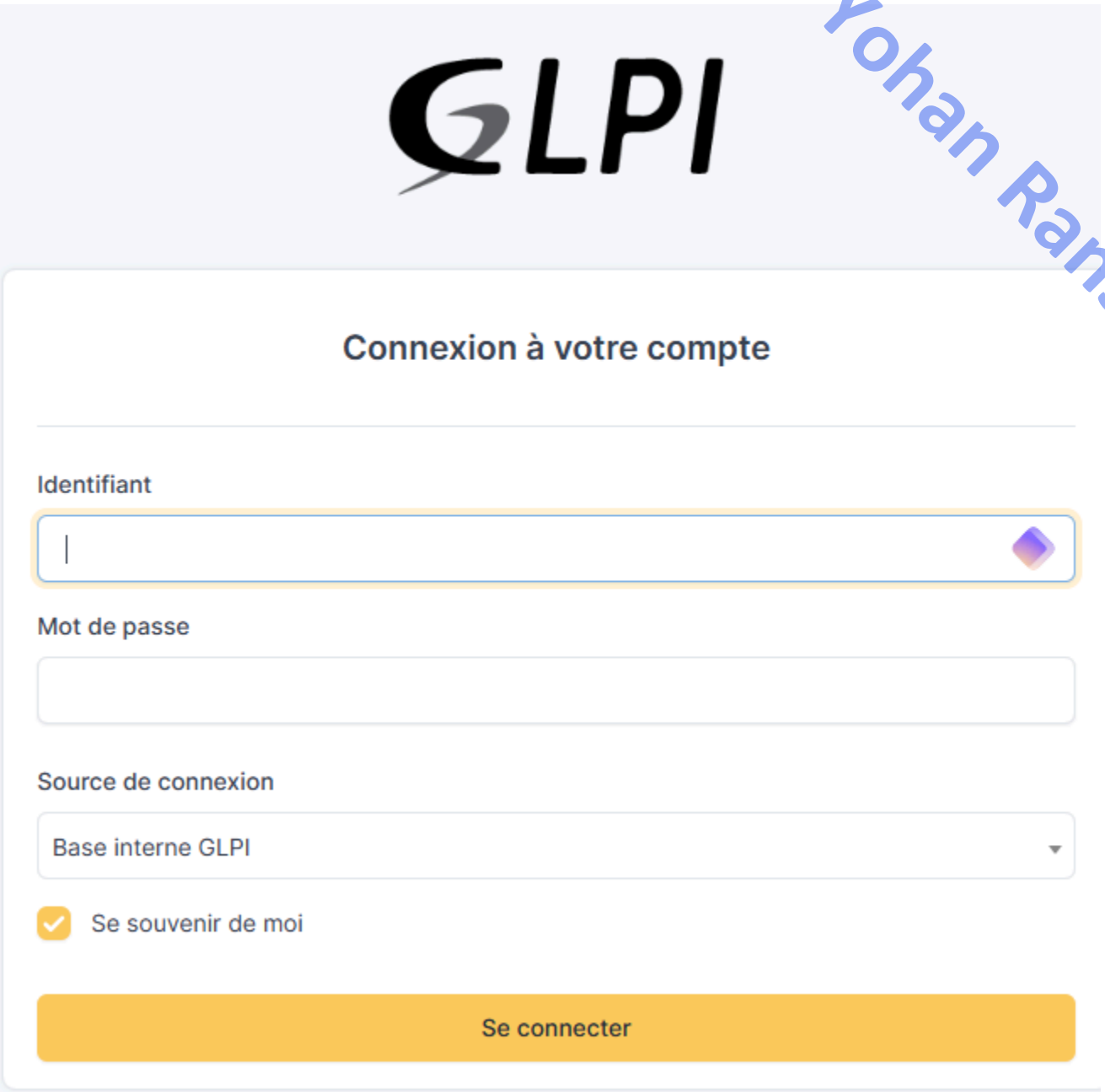
L'alerte de sécurité !

Tu vas voir un écran effrayant : "Attention : risque probable de sécurité".

C'est normal ! Firefox te prévient : "Je chiffre bien la connexion, MAIS je ne connais pas l'organisme qui a signé ce certificat" (puisque c'est toi !).

• Clique sur Avancé.
• Clique sur Accepter le risque et poursuivre.

Tu es maintenant connecté à GLPI avec un petit cadenas (parfois barré d'un triangle jaune selon les navigateurs car auto-signé), mais tes données sont chiffrées !



Partie 3 (Bonus) : Le garde du corps (Pare-feu UFW)

Sécuriser le transport des données (HTTPS), c'est bien. Empêcher les intrus d'entrer, c'est mieux.

Nous allons utiliser UFW (Uncomplicated Firewall). Imagine-le comme un videur à l'entrée d'une boîte de nuit.

La stratégie du "Tout refuser, sauf..."

En sécurité, la meilleure politique est de tout fermer par défaut, et d'ouvrir seulement ce qui est nécessaire.

1. Vérifie l'état actuel du pare-feu :

```
sudo ufw status
```

(Il devrait être "inactive").

2. Ouvre les ports vitaux AVANT d'activer le pare-feu (sinon tu risques de te bloquer toi-même, surtout si tu travaillais en SSH à distance !) :

o Port 22 (SSH) : Pour l'administration à distance (si tu l'utilises).

```
sudo ufw allow ssh
```

o Port 80 (HTTP) : Pour rediriger les gens vers le site sécurisé.

```
sudo ufw allow http
```

o Port 443 (HTTPS) : Pour l'accès sécurisé à GLPI.

```
sudo ufw allow https
```

3. Active le pare-feu :

```
sudo ufw enable
```

(Réponds 'y' pour confirmer).

4. Vérifie que le videur fait son travail :

```
sudo ufw status verbose
```

Tu devrais voir une liste où seuls les ports 22, 80 et 443 sont en "ALLOW". Tout le reste est bloqué.

L'inventaire automatique avec l'Agent GLPI

1. Le concept : Client-Serveur et Automatisation

Avant de taper des commandes, comprenons ce que nous allons construire.
Nous allons installer un petit espion (l'Agent) sur un ordinateur client. Cet agent va :

1. Scanner le matériel (CPU, RAM, Disque) et les logiciels installés.
2. Contacter le serveur GLPI.

3. Lui transmettre le rapport complet.
   • Le Client (L'Agent) : C'est le journaliste sur le terrain. Il récolte l'info.
   • Le Serveur (GLPI) : C'est le rédacteur en chef. Il reçoit l'info et la classe dans la base de données.

2. Préparation de l'environnement

Pour ce TP, tu as besoin de :

• Ton serveur GLPI (celui du TP précédent, qui tourne en HTTPS).
• Une deuxième machine Linux (une autre VM Ubuntu par exemple) qui jouera le rôle du "Client à inventorier".

> **Important**
>
> Note importante : Dans ce TP, comme nous utilisons un certificat HTTPS auto-signé (créé nous-mêmes et non par une autorité officielle), nous devrons dire à l'Agent de ne pas être trop regardant sur la sécurité du certificat. En production réelle, on ne ferait pas ça !

3. Installation de l'Agent GLPI (Côté Client)

Connecte-toi sur ta machine cliente (pas sur le serveur GLPI !).

Étape 3.1 : Installer les outils de base

L'agent a besoin de quelques outils pour fouiller dans le matériel, notamment dmidecode.

```
sudo apt update
sudo apt install dmidecode libxml2-utils -y
```

Étape 3.2 : Télécharger le script d'installation

Les développeurs de GLPI fournissent un script "magique" qui télécharge et installe la bonne version de l'agent. Nous allons le récupérer.

```
wget https://github.com/glpi-project/glpi-agent/releases/download/1.15/glpi-agent-1.15-linux-installer.pl
```

> **Note**
>
> (Note : La version peut changer.)

Étape 3.3 : Lancer l'installation

C'est ici que nous allons configurer l'agent pour qu'il parle à ton serveur.
Il nous faut deux informations cruciales :

1. L'URL du serveur : https://ADRESSE_IP_DE_TON_SERVEUR/front/inventory.php
2. L'option "No SSL Check" : Pour accepter ton certificat auto-signé.

Lance la commande suivante (remplace 192.168.X.X par l'IP réelle de ton serveur GLPI) :

```
sudo perl glpi-agent-1.15-linux-installer.pl --
server=https://192.168.X.X/front/inventory.php --no-ssl-check --install
```

> **Note**
>
> Déchiffrons cette commande :
>
> • perl ...installer.pl : On exécute le script d'installation.
> • --url=... : On indique l'adresse de livraison des inventaires. Attention, l'adresse se termine bien par /front/inventory.php.
> • --no-ssl-check : On dit à l'agent "Fais confiance au serveur même si le certificat n'est pas reconnu officiellement".
> • --install : On confirme qu'on veut installer le service.

Si tout se passe bien, le script te dira "GLPI Agent is installed".

4. Vérification et premier Inventaire

> Une fois installé, l'agent tourne en tâche de fond (comme un démon/service). Mais nous n'allons pas attendre qu'il se réveille. Nous allons le forcer à travailler tout de suite pour voir si ça marche. Mais d'abord, vérifie que l'option Activer l'inventaire est bien sur Oui côté Serveur dans le menu Administration > Inventaire.
>
> 
>

Étape 4.1 : Forcer l'envoi de l'inventaire

Ensuite, de retour sur le PC Client, tape cette commande :

```
sudo glpi-agent --force
```

Observe bien le résultat dans le terminal.

• Si tu vois : [info] target server0: server https://... response: 200 OK
o C'est GAGNÉ ! Le code 200 signifie que le serveur a bien reçu et accepté le colis.
• Si tu vois : 500 Internal Server Error ou 400 Bad Request
o Vérifie l'URL que tu as renseignée.
• Si tu vois : Can't connect...
o Vérifie ton pare-feu (UFW) sur le serveur : as-tu bien autorisé le port 443 ? (sudo ufw allow 443).

5. Le résultat (Côté Serveur)

Retourne maintenant sur ton navigateur web, sur l'interface de GLPI.

1. Connecte-toi (login: glpi / pass: glpi ou celui que tu as changé).
2. Va dans le menu Parc (Assets) > Ordinateurs.

NOM	STATUT	FABRICANT	NUMÉRO DE SÉRIE	TYPE	MODÈLE	SYSTÈME D'EXPLOITATION - I
yohan		VMware, Inc.	VMware-56 4d c0 3e 2c 1a 56 6a-2d b4 74 81 06 be 2d 66	VMware	VMware Virtual Platform	Ubuntu 24.04.3 LTS

Surprise ! Une ligne est apparue.

Tu n'as rien saisi, et pourtant GLPI connait maintenant :

• Le nom de la machine.
• Son système d'exploitation (Ubuntu, Debian...).
• Son numéro de série.
• La quantité de RAM et le type de processeur.

Clique sur le nom de l'ordinateur pour voir la richesse des informations remontées.

Automatiser la sauvegarde de GLPI

Note

1. Comprendre : Que doit-on sauvegarder ?

Beaucoup de débutants pensent qu'il suffit de copier le dossier du logiciel pour le sauvegarder. Pour une application Web comme GLPI, c'est faux. Il y a deux éléments vitaux et distincts à protéger.

1. Le cerveau (La base de données SQL) :

o Elle contient tout ce qui est écrit : la liste des ordinateurs, les utilisateurs, les tickets, les notes, la configuration.
o Si tu perds ça, GLPI redevient vide.
o Outil de sauvegarde : mysqldump.

2. Le corps (Les fichiers) :

o Ce sont les fichiers physiques : le code PHP de GLPI, tes plugins installés, et surtout les documents joints aux tickets (captures d'écran, factures PDF).
o Outil de sauvegarde : tar.

2. Préparation du terrain

Nous allons créer un dossier dédié pour ranger nos sauvegardes proprement.

1. Connecte-toi à ton terminal sur le serveur.
2. Crée le dossier de stockage :

```
sudo mkdir -p /var/backups/glpi
```

3. Pour ce TP, nous allons travailler en tant que root (super-utilisateur) pour éviter les problèmes de droits, car nous devons lire des fichiers système protégés. Passe en mode root :

```
sudo -i
```

> (Ton invite de commande doit passer de $ à #).

3. Création du script de sauvegarde

Au lieu de taper les commandes tous les jours, nous allons les écrire dans un script. C'est un simple fichier texte contenant une suite d'instructions que Linux va exécuter.

Étape 3.1 : Créer le fichier

```
nano /usr/local/bin/backup_glpi.sh
```

Étape 3.2 : Rédiger le script (Copier-Coller)

Copie ce code dans le fichier.

> **Note**
>
> Attention : Remplace TON_MOT_DE_PASSE_SQL par celui défini lors de l'installation de GLPI.

> **Note**
>
> #!/bin/bash
>
> **--- CONFIGURATION ---**
>
> DATE=$(date +%Y-%m-%d_%Hh%M)
> BACKUP_DIR="/var/backups/glpi"
>
> **Identifiants Base de Données (voir ton installation)**
>
> DB_USER="glpi_user"
> DB_PASS="TON_MOT_DE_PASSE_SQL"
> DB_NAME="glpidb"
>
> **--- 1. SAUVEGARDE DE LA BASE DE DONNÉES (Le Cerveau) ---**
>
> echo "Sauvegarde de la base de données en cours..."
> mysqldump -u $DB_USER -p$DB_PASS $DB_NAME > $BACKUP_DIR/glpi_db$DATE.sql
>
> **--- 2. SAUVEGARDE DES FICHIERS (Le Corps) ---**
>
> echo "Sauvegarde des fichiers en cours..."
> On compresse le dossier /var/www/html/glpi
> tar -czf BACKUP_DIR/glpi_files_DATE.tar.gz /var/www/html/glpi
>
> **--- 3. NETTOYAGE (Garder les 7 derniers jours) ---**

Supprime les fichiers vieux de plus de 7 jours pour ne pas saturer le disque

find $BACKUP_DIR -type f -mtime +7 -name "*.gz" -delete*
find $BACKUP_DIR -type f -mtime +7 -name ".sql" -delete

echo "Sauvegarde terminée avec succès dans $BACKUP_DIR"

Déchiffrons le script :

• mysqldump ... > fichier.sql : On extrait tout le texte de la base de données vers un fichier.
• tar -czf ... : C'est l'équivalent de "WinZip" ou "WinRAR". c (create), z (zip/compress), f (file).
• find ... -delete : Très important ! Sans ça, ton disque dur finira par être plein. On garde une semaine d'historique.

Sauvegarde avec Ctrl+O puis Entrée, et quitte avec Ctrl+X.

Étape 3.3 : Rendre le script exécutable et sécurisé

Pour l'instant, c'est juste du texte. Il faut dire à Linux que c'est un programme.
De plus, comme il contient ton mot de passe, personne d'autre que root ne doit pouvoir le lire.

```
chmod 700 /usr/local/bin/backup_glpi.sh
```

Note

(700 signifie : Moi (root) je fais tout, les autres n'ont aucun droit).

Étape 3.4 : Le test manuel (Obligatoire !)

> Ne jamais automatiser quelque chose qui ne marche pas manuellement.

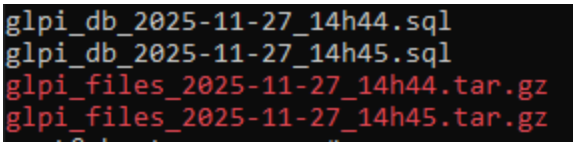
1. Lance ton script :

```
/usr/local/bin/backup_glpi.sh
```

2. Vérifie que les fichiers sont bien créés :

```
ls -lh /var/backups/glpi/
```

Tu dois voir deux fichiers (un .sql et un .tar.gz) avec la date d'aujourd'hui. Si oui, bravo, ton script fonctionne !



4. L'automatisation avec CRON

Cron est le réveil-matin de Linux. Il sert à planifier des tâches.

Nous allons dire à Cron : "Tous les jours, à 3h00 du matin (quand personne ne travaille), lance le script de sauvegarde".

1. Ouvre l'éditeur de tâches planifiées :

```
crontab -e
```

(Si on te demande de choisir un éditeur, choisis nano, c'est le plus simple).

2. Va tout en bas du fichier et ajoute cette ligne :

> 0 3 * * * /usr/local/bin/backup_glpi.sh

Comment lire le cron ?

o 0 : À la minute 0
o 3 : De la 3ème heure (3h00 du matin)
o *: Tous les jours du mois*
o : Tous les mois
o * : Tous les jours de la semaine

3. Enregistre et quitte.

Le système te répondra : crontab: installing new crontab. C'est validé !

5. Comment restaurer en cas de catastrophe ?

Une sauvegarde ne sert à rien si on ne sait pas la remettre en place. Voici les commandes pour "réparer" ton serveur (à conserver précieusement) :

1.
    1. Restaurer les fichiers (Le corps)

```
tar -xzf /var/backups/glpi/glpi_files_DATE.tar.gz -C /
```

> **Note**
>
> Remplacer "DATE" par la date et l'heure du fichier `glpi_files_2025-11-27_14h44.tar.gz`

2. Restaurer la base de données (Le cerveau)
   Attention, cela écrase les données actuelles !

```
mysql -u glpi_user -p glpidb < /var/backups/glpi/glpi_db_DATE.sql
```

> **Note**
>
> Remplacer "DATE" par la date et l'heure du fichier `glpi_db_2025-11-27_14h44.sql`

Externaliser les sauvegardes (SSH & Rsync)

Partie 1 : L'authentification sans mot de passe

C'est le défi principal.

Si tu lances une copie manuelle, le serveur B va te demander un mot de passe.

Mais notre script tourne la nuit, tout seul. Il ne peut pas taper de mot de passe !

Nous allons utiliser le principe des Clés SSH.

L'analogie de la clé et du cadenas :

• Nous allons fabriquer une clé unique (Clé Privée) que nous gardons sur le serveur GLPI.
• Nous allons fabriquer un cadenas correspondant (Clé Publique) que nous allons installer sur le serveur de Sauvegarde.
• Ainsi, le serveur GLPI pourra entrer sans taper de code, juste en montrant sa clé.

Étape 1.1 : Générer les clés (Sur le serveur GLPI)

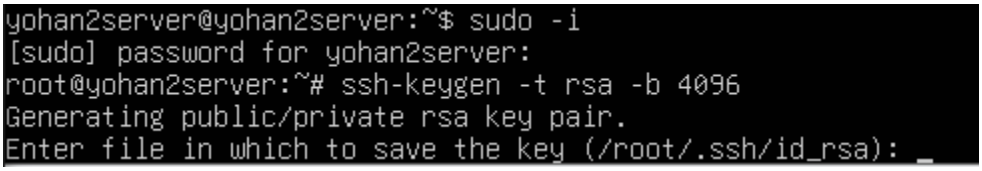
Attention : Comme notre script de sauvegarde tourne en root (via Cron), nous devons générer les clés pour l'utilisateur root.

1. Connecte-toi en root sur le serveur GLPI :

```
sudo -i
```

2. Génère la paire de clés (appuie sur Entrée à chaque question, ne mets PAS de passphrase, sinon l'automatisation est impossible) :

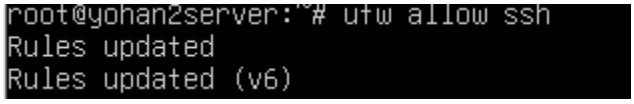
```
ssh-keygen -t rsa -b 4096
```



Le système a créé deux fichiers dans /root/.ssh/ : id_rsa (la clé) et id_rsa.pub (le cadenas).

Libérez les ports sur yohan2server et le client

```
ufw allow ssh
```



Étape 1.2 : Envoyer le cadenas sur le serveur de sauvegarde

Toujours depuis le serveur GLPI, nous allons envoyer la clé publique vers le serveur B.

Remplace utilisateur par le nom de ton utilisateur sur le serveur B (ex: etudiant ou admin) et IP_SERVEUR_B par l'IP réelle.

```
ssh-copy-id yohan2server@192.168.147.130
```

(Tu devras taper le mot de passe de l'utilisateur distant une dernière fois pour installer la clé).

Étape 1.3 : Le test de vérité

C'est le moment critique. Toujours depuis le serveur GLPI (en root), tente de te connecter au serveur B :

```
ssh yohan2server@192.168.147.130
```

• Si tu entres directement sans mot de passe : C'est gagné ! Tape exit pour revenir sur GLPI.
• Si on te demande un mot de passe : Quelque chose a raté, recommence l'étape 1.2.