BTS SIO2 LM
ECOLE IRIS

Mise en Place d'une Solution de Gestion du Parc Informatique avec Synchronisation Active Directory, Notification des Tickets et Collecteurs d'E-mails

QUENTIN MASSIAS / OCEANE GONCALVES

Présentation d'OCS Inventory

Le parc informatique d'une organisation regroupe une diversité de matériels et de logiciels accumulés au fil du temps. On y trouve des équipements variés comme des téléphones, ordinateurs, imprimantes et éléments d'interconnexion, ainsi que des systèmes d'exploitation et applications dans différentes versions. Cette hétérogénéité entraîne des niveaux de sécurité inégaux et complique la gestion, notamment dans des environnements étendus où la performance et la réactivité sont essentielles.

La gestion du parc ne se limite pas à l'inventaire du matériel et des logiciels, elle couvre également le suivi et l'évolution des équipements :

- Localisation et état du matériel
- Gestion des licences et des contrats
- Télé-déploiement des logiciels et mises à jour
- Suivi financier et cycle de vie des équipements
- Documentation technique et gestion des partenaires
- Analyse statistique et anticipation des besoins

Une gestion efficace permet de répondre aux questions quotidiennes des administrateurs : localisation des équipements, état des disques durs, versions des logiciels installés, connexions réseau, garantie des matériels, valeur actuelle des actifs informatiques. Aujourd'hui, les DSI adoptent de plus en plus le référentiel ITIL pour structurer et optimiser ces processus.

OCS Inventory repose sur une architecture client-serveur et se compose de plusieurs modules :

- **Serveur de communication** : collecte, classe et archive les données des machines clientes. Il repose sur Apache, MySQL et PHP, compatible avec Linux et Windows
- **Agent client** : installé sur chaque machine, il remonte les informations vers le serveur
- **Serveur d'administration** : interface web en PHP permettant de consulter les inventaires et gérer les droits utilisateurs
- **Serveur de déploiement** : assure l'installation centralisée des logiciels et mises à jour, sécurisé via Apache SSL

Cette solution permet d'automatiser l'inventaire, d'optimiser la gestion des actifs informatiques et d'améliorer le suivi des ressources tout en garantissant une administration centralisée et sécurisée.

Mettre en place SSH

Nous allons installer SSH server sur la machine puis nous y connecter depuis notre ordinateur physique. Cela est plus pratique, puisqu'en se connectant sur la machine avec SSH, il est possible de copier/coller depuis notre terminal sur l'hôte physique, ce qui n'est pas possible directement sur la VM :

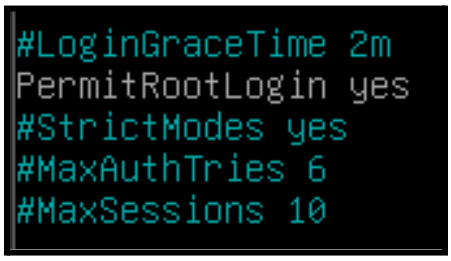
#apt install openssh-server

Autoriser le compte root à se connecter en ssh

Nous devons pour cela modifier le fichier **sshd_config** qui contient les informations de configuration SSH :

#cd /etc/ssh
#nano sshd_config

On va jusqu'à la ligne :
#PermitRootLogin_prohibit-password
On enlève le **#** et **prohibit-password** et on entre **yes, puis** l'on enregistre :



Redémarrer SSH

Dès qu'on touche au fichier de configuration de SSH, il faut redémarrer le service
#service ssh restart → Redémarrer SSH

#service ssh status → Vérifier que le service a bien redémarré

Changer l'adressage IP dynamique en statique

#nano /etc/network/interfaces → Modifier
En bas du fichier à la ligne qui commente :

#The primary network interface
Effacer la mention **dhcp** de la carte réseau et entrer **static,** et préciser l'adresse ce qui donne :

iface ens33 inet static
address 172.20.1.7/24
gateway 172.20.1.1

```
# The primary network interface
allow-hotplug ens33
iface ens33 inet static
address 172.20.1.7/24
gateway 172.20.1.1
```

Réactiver la carte

Pour appliquer les changement, on désactive puis réactive la carte :
#ifdown ens33
#ifup ens33
#ip a → vérifier la nouvelle adresse IP statique

```
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group de
fault qlen 1000
    link/ether 00:0c:29:25:26:fe brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 172.20.1.7/24 brd 172.20.1.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe25:26fe/64 scope link
       valid_lft forever preferred_lft forever
```

Vérifier la connectivité réseau

#ping 172.20.1.1 → on PING l'interface LAN-server de pfsense
Si cela ne fonctionne pas, il faut vérifier dans les paramètres machine que la carte réseau est dans le bon LAN_server ou que pfsense (Heimdall) est bien allumé !

Se connecter en SSH depuis un terminal sur l'hôte physique

On PING le serveur OCS avec le terminal puis on se connecte :
#ping 172.20.1.7
#ssh root@172.20.1.7

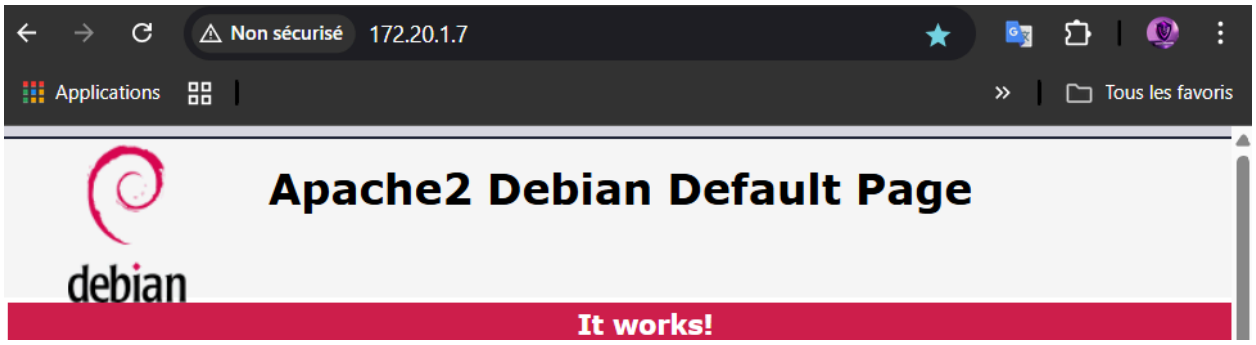
S'il y a un pb de clé, on efface l'ancienne clé dans le fichier known_host (sur l'hôte physique) :
#ssh-keygen -f "/home/user/.ssh/known_host" -R "172.20.1.7"

ou simplement
#ssh-keygen -R 172.20.1.7
On accepte la nouvelle clé et on entre les credentials
(Sur Linux, il suffit de supprimer le fichier **known_host q**ui sera automatiquement regénéré avec la clé publique à jour).
Le chemin : **/home/nom_utilisateur/.ssh/known_hosts**

Installation du serveur LAMP

#apt update && apt upgrade → Mettre à jour et installer les paquets
#hostnamectl set-hostname ocs → renommer la machine
#apt install apache2 php mariadb-server -y → Installer le serveur LAMP

Test de connexion du serveur LAMP :
Sur l'hôte physique, entrer l'adresse IP de la machine OCS sur un navigateur web. Cette page apparaît :



Vérifions les étapes suivantes :
#a2query -m mpm_prefork → Vérifie l'activation du module mpm_prefork
#a2query -m php8.2 → Vérifie l'activation du module mpm_prefork

Si les modules ne sont pas actifs, les démarrer avec ces lignes :
#a2enmod mpm_prefork
#a2enmod php8.2

#apache2ctl -t → Vérifie les syntaxes des fichiers de configuration d'apache2. Elle permet de s'assurer qu'il n'y a pas d'erreurs dans la configuration avant de redémarrer ou de recharger le serveur web

Si Apache retourne une erreur liée au **nom de domaine complet (FQDN - Fully Qualified Domain Name)**, cela signifie souvent qu'il manque une déclaration correcte du nom d'hôte. Le fichier fqdn.conf peut être créé pour corriger cela en définissant explicitement le nom d'hôte utilisé par le serveur.

#echo "ServerName stadiumcompany.local" | sudo tee /etc/apache2/conf-available/fqdn.conf → Crée le fichier fqfn.conf
#sudo a2enconf fqdn → Activer la configuration
#system reload apache2.service → Recharge la configuration sans interrompre les connexions en cours
#apache2ctl -t → Nouvelle vérification pour voir si l'erreur a disparu

Restriction de l'accès à la base de donnée :

Par défaut, MariaDB n'est pas complètement sécurisé après son installation. Nous allons utiliser **le script mysql_secure_installation** pour restreindre l'accès et appliquer des bonnes pratiques de sécurité :
#which mysql_secure_installation → permet de savoir où se trouve un exécutable installé sur le système

```
┌──(root💀ocs)-[/]
└─# which mysql_secure_installation
/usr/bin/mysql_secure_installation
```

On voit apparaître le script mysql_secure_installation.
#mysql_secure_installation → On lance ce script afin de restreindre l'accès au serveur.

Ensuite, plusieurs questions vont se suivre, auxquelles nous devrons répondre ainsi :

1. **Enter current password for root** → Appuyez sur **Entrée** si aucun mot de passe n'a été défini.
2. **Set root password? [Y/n]** → **N** (MariaDB utilise `unix_socket`, inutile de modifier l'authentification).
3. **Remove anonymous users? [Y/n]** → **Y** (supprime les comptes anonymes pour renforcer la sécurité).
4. **Disallow root login remotely? [Y/n]** → **Y** (bloque les connexions root distantes pour éviter les intrusions).
5. **Remove test database and access to it? [Y/n]** → **Y** (supprime la base de test qui peut poser des risques).
6. **Reload privilege tables now? [Y/n]** → **Y** (applique immédiatement les modifications).

Une fois ces étapes terminées, la base de données est sécurisée et prête à être utilisée. ✅

Installation et configuration d'OCS

Nous allons créer une BDD mariadb pour OCS

#mysql -u root → Entrer en mode client Mysql/Mariadb avec l'utilisateur root

MariaDB [(none)]> create database dbocs; → Je crée une base de données qui s'appelle dbocs

MariaDB [(none)]> grant all privileges on dbocs.* to userocs@'localhost' identified by 'userocs'; → Je crée un utilisateur ocsuser et je lui donne tous les privileges sur toutes la base dbocs
MariaDB [(none)]> flush privileges; → Je recharge les droits
MariaDB [(none)]> show databases; → J'affiche ma base de données
MariaDB [dbocs]> select user,host from mysql.user; → J'affiche les utilisateurs dans mariadb

```
┌──(root💀ocs)-[/]
└─# mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 37
Server version: 10.11.6-MariaDB-0+deb12u1 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database dbocs;
Query OK, 1 row affected (0,000 sec)

MariaDB [(none)]> grant all privileges on dbocs.* to userocs@'localhost' identified by 'userocs';
Query OK, 0 rows affected (0,002 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| dbocs              |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0,001 sec)

MariaDB [(none)]> select user,host from mysql.user;
+-------------+-----------+
| User        | Host      |
+-------------+-----------+
| mariadb.sys | localhost |
| mysql       | localhost |
| root        | localhost |
| userocs     | localhost |
+-------------+-----------+
4 rows in set (0,002 sec)
```

MariaDB [dbocs]> SHOW GRANTS FOR userocs@localhost; → J'affiche les droits de l'utilisateur userocs

```
MariaDB [(none)]> SHOW GRANTS FOR userocs@localhost;
+---------------------------------------------------------------------------------------------------------------+
| Grants for userocs@localhost                                                                                  |
+---------------------------------------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `userocs`@`localhost` IDENTIFIED BY PASSWORD '*300C62D5B575E2ED1CC2922CAA453A315FE05B54' |
| GRANT ALL PRIVILEGES ON `dbocs`.* TO `userocs`@`localhost`                                                    |
+---------------------------------------------------------------------------------------------------------------+
2 rows in set (0,000 sec)
```

Installation des librairies Perl et des modules PHP pour OCS Inventory

Introduction aux modules Perl et PHP
Perl utilise des modules pour organiser et simplifier le code. Ces fichiers, reconnaissables à leur extension **.pm,** contiennent des fonctionnalités spécifiques et réutilisables.

Lorsqu'un script devient trop long et difficile à maintenir, il est découpé en plusieurs modules, améliorant ainsi sa lisibilité et sa modularité.
CPAN : La bibliothèque de modules Perl
CPAN (*Comprehensive Perl Archive Network*) est une immense collection de modules prêts à l'emploi pour Perl. Certains sont directement intégrés à Perl car ils sont essentiels à son bon fonctionnement.

- CPAN est régulièrement mis à jour et la plupart des modules qu'il propose sont testés et fiables.
- Avant toute installation, il est recommandé de lire la documentation associée.
- L'utilisation de modules existants permet d'optimiser le développement en évitant de recréer des fonctionnalités déjà disponibles.

MetaCPAN – Site officiel permettant de rechercher et d'explorer tous les modules disponibles.

L'installation d'un module via CPAN se fait avec la commande suivante :
#cpan NomDuModule

Dépendances nécessaires pour OCS Inventory
OCS Inventory repose sur plusieurs modules Perl, bibliothèques PHP et dépendances système. Leur installation est indispensable au bon fonctionnement de l'application.

Documentation OCS Inventory → Liste officielle des dépendances requises

#apt install libapache2-mod-perl2 libapache2-mod-perl2-dev libxml-simple-perl \libapache-dbi-perl libarchive-zip-perl libdbd-mysql-perl libnet-ip-perl \libsoap-lite-perl make apache2-dev php-{mysql,gd,curl,mbstring,soap,xml} -y → L'installation des bibliothèques nécessaires se fait avec cette longue commande apt

Explication des modules installés :

- Modules pour l'intégration de Perl à Apache
  - libapache2-mod-perl2 et libapache2-mod-perl2-dev : Permettent à Apache d'exécuter des scripts Perl en tant que modules intégrés.
  - libapache-dbi-perl : Facilite la gestion des connexions à la base de données.
- Modules pour la manipulation des données
  - libxml-simple-perl : Gestion et manipulation de fichiers XML.
  - libnet-ip-perl : Gestion et conversion des adresses IP.
  - libarchive-zip-perl : Permet de créer, lire et extraire des fichiers ZIP.

    - libdbd-mysql-perl : Interface Perl pour communiquer avec une base de données MySQL.
    - libsoap-lite-perl : Gestion des communications SOAP pour les services web.
- Modules PHP indispensables
    - php-mysql : Connexion de PHP avec MySQL.
    - php-gd : Gestion des images (ex. graphiques, miniatures).
    - php-curl : Communication avec des services externes via HTTP/S.
    - php-mbstring : Gestion des caractères multibyte pour les langues non-ASCII.
    - php-soap : Intégration du protocole SOAP.
    - php-xml : Manipulation des documents XML.
- Autres outils nécessaires
    - apache2-dev : Outils de développement pour Apache.
    - make : Utilisé pour compiler et installer des programmes depuis le code source.

Installation des modules supplémentaires via CPAN
Certains modules spécifiques ne sont pas disponibles via apt et doivent être installés avec CPAN directement. Voici la liste des modules supplémentaires à installer :

#cpan SOAP::Lite
#cpan Mojolicious::Lite
#cpan Switch
#cpan Plack::Handler
#cpan Apache2::SOAP

Explication des modules CPAN installés :

- **SOAP::Lite** : Gestion des échanges de données via SOAP.
- **Mojolicious::Lite** : Framework minimaliste pour le développement d'applications web en Perl.
- **Switch** : Ajoute une structure switch au langage Perl (équivalent aux case en C).
- **Plack::Handler** : Permet de gérer des requêtes HTTP dans une application Perl.
- **Apache2::SOAP** : Intégration avancée du protocole SOAP à Apache.

Certaines dépendances comme **XML::Entities** peuvent nécessiter une installation spécifique :
#perl -MCPAN -e "install XML::Entities"

Remarque : Lorsqu'une question apparaît pendant l'installation d'un module CPAN, il suffit de taper y (ou appuyer sur Entrée) pour valider.

Nous avons installé toutes les dépendances nécessaires au fonctionnement de OCS Inventory.

Installation d'OCS

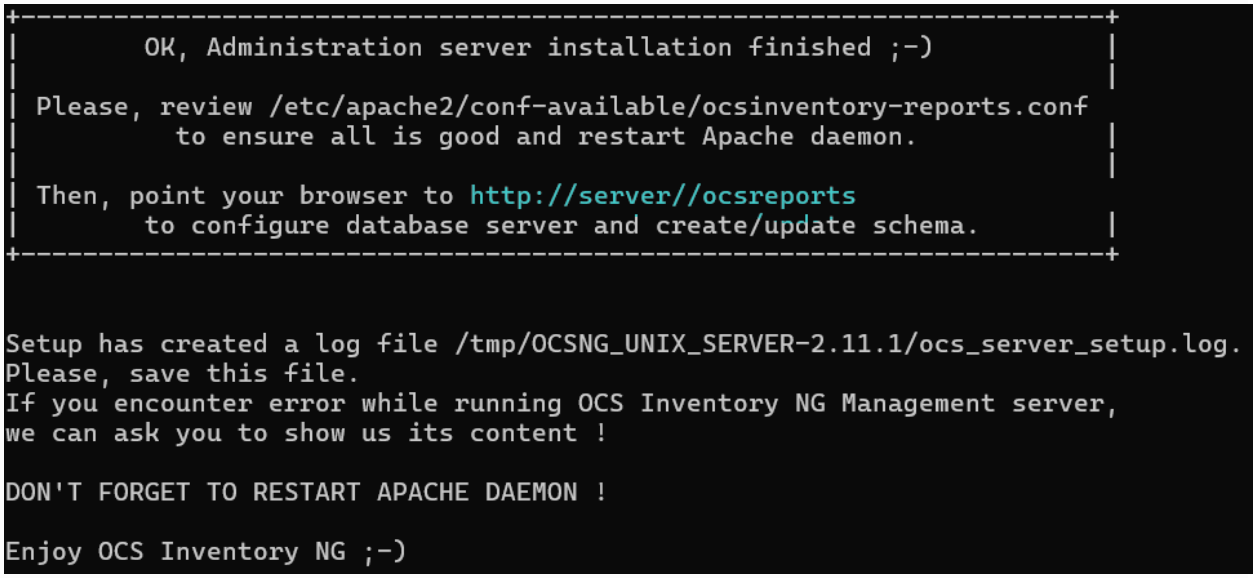
Récupération du lien de téléchargement de la dernière version d'OCS sur le site

https://ocsinventory-ng.org/?page_id=1235&lang=fr
#cd /tmp/ → Se rendre dans le dossier tmp pour décompresser notre fichier
#wget https://github.com/OCSInventory-NG/OCSInventory-ocsreports/releases/download/2.11.1/OCSNG_UNIX_SERVER-2.11.1.tar.gz → Récupère la version d'OCS directement sur github pour simplifier l'installation
#tar xzf OCSNG_UNIX_SERVER-2.11.1.tar.gz → Décompresser le fichier avec l'utilitaire gzip
#cd xzf OCSNG_UNIX_SERVER-2.11.1.tar.gz → Se déplacer dans le dossier décompressé
#./setup.sh → Après un ls, on voit ce fichier que l'on exécute

```
┌──(root💀ocs)-[/tmp]
└─# tar xzf OCSNG_UNIX_SERVER-2.11.1.tar.gz

┌──(root💀ocs)-[/tmp]
└─# ls
cpan_install_HzT_.txt
cpan_install_nwm0.txt
hx7PV
OCSNG_UNIX_SERVER-2.11.1
OCSNG_UNIX_SERVER-2.11.1.tar.gz
systemd-private-96445681939949aaa90ed4dd52675124-apache2.service-ugkeWQ
systemd-private-96445681939949aaa90ed4dd52675124-systemd-logind.service-iSfLMx
systemd-private-96445681939949aaa90ed4dd52675124-systemd-timesyncd.service-mqT6U1
vmware-root_460-834905685

┌──(root💀ocs)-[/tmp]
└─# cd OCSNG_UNIX_SERVER-2.11.1

┌──(root💀ocs)-[/tmp/OCSNG_UNIX_SERVER-2.11.1]
└─# ls
Apache  binutils  dtd  INSTALL  ocsreports  setup.sh
Api     cpanfile  etc  LICENSE  README.md

┌──(root💀ocs)-[/tmp/OCSNG_UNIX_SERVER-2.11.1]
└─# ./setup.sh
```

Ce script, une fois lancé, va demander plusieurs paramètres ou confirmations,avec des options par défaut **pré-sélectionnées**, et il suffit de valider chaque question en appuyant sur **Entrée**.

La fin de l'installation nous indique comment nous connecter au serveur :



Mise à jour des paramètres de la base de données et configuration d'OCS Inventory

Pour garantir le bon fonctionnement d'OCS Inventory, nous devons mettre à jour les fichiers de configuration avec le nom de la base de données **(dbocs)** et le nom de l'utilisateur **(ocsuser).** OCS Inventory doit savoir où et comment se connecter à la base de données. Nous allons donc modifier les fichiers suivants :

→ **zz-ocsinventory-restapi.conf**
→ **z-ocsinventory-server.conf**
→ **dbconfig.inc.php**

Afin d'y renseigner les données suivantes :
- Le nom de la base de données **(dbocs)**
- Le nom d'utilisateur **(ocsuser)**

#nano /etc/apache2/conf-available/zz-ocsinventory-restapi.conf → Modifier comme suit :

```
PerlOptions +Parent

<Perl>
  $ENV{PLACK_ENV} = 'production';
  $ENV{MOJO_HOME} = '/usr/local/share/perl/5.36.0';
  $ENV{MOJO_MODE} = 'deployment';
  $ENV{OCS_DB_HOST} = 'localhost';
  $ENV{OCS_DB_PORT} = '3306';
  $ENV{OCS_DB_LOCAL} = 'dbocs';
  $ENV{OCS_DB_USER} = 'userocs';
  $ENV{OCS_DB_PWD} = 'userocs';
  $ENV{OCS_DB_SSL_ENABLED} = 0;
#  $ENV{OCS_DB_SSL_CLIENT_KEY} = '';
#  $ENV{OCS_DB_SSL_CLIENT_CERT} = '';
#  $ENV{OCS_DB_SSL_CA_CERT} = '';
  $ENV{OCS_DB_SSL_MODE} = 'SSL_MODE_PREFERRED';
</Perl>
```

#nano /etc/apache2/conf-available/z-ocsinventory-server.conf → Modifier comme suit :

```
<IfModule mod_perl.c>

  # Which version of mod_perl we are using
  # For mod_perl <= 1.999_21, replace 2 by 1
  # For mod_perl > 1.999_21, replace 2 by 2
  PerlSetEnv OCS_MODPERL_VERSION 2

  # Master Database settings
  # Replace localhost by hostname or ip of MySQL server for WRITE
  PerlSetEnv OCS_DB_HOST localhost
  # Replace 3306 by port where running MySQL server, generally 3306
  PerlSetEnv OCS_DB_PORT 3306
  # Name of database
  PerlSetEnv OCS_DB_NAME dbocs
  PerlSetEnv OCS_DB_LOCAL dbocs
  # User allowed to connect to database
  PerlSetEnv OCS_DB_USER userocs
  # Password for user
  PerlSetVar OCS_DB_PWD userocs
  # SSL Configuration
```

#nano /usr/share/ocsinventory-reports/ocsreports/dbconfig.inc.php → Modifier comme suit

```
<?php
$_SESSION["SERVEUR_SQL"]="localhost";
$_SESSION["COMPTE_BASE"]="userocs";
$_SESSION["PSWD_BASE"]="userocs";
?>
```

OCS Inventory utilise plusieurs fichiers de configuration pour s'intégrer à Apache. Nous devons les activer avec la commande a2enconf :

#a2enconf z-ocsinventory-server.confa
#a2enconf zz-ocsinventory-restapi.conf
#a2enconf ocsinventory-reports.conf

Ces fichiers permettent à Apache de gérer :
- Le serveur OCS Inventory (z-ocsinventory-server.conf)
- L'API REST (zz-ocsinventory-restapi.conf)
- L'interface web d'OCS Inventory (ocsinventory-reports.conf)

Apache gère ses sites et services via deux répertoires :
/etc/apache2/sites-available/ → Répertoire où sont stockées les configurations des sites mais non activées.
/etc/apache2/sites-enabled/ → Répertoire des sites activés.

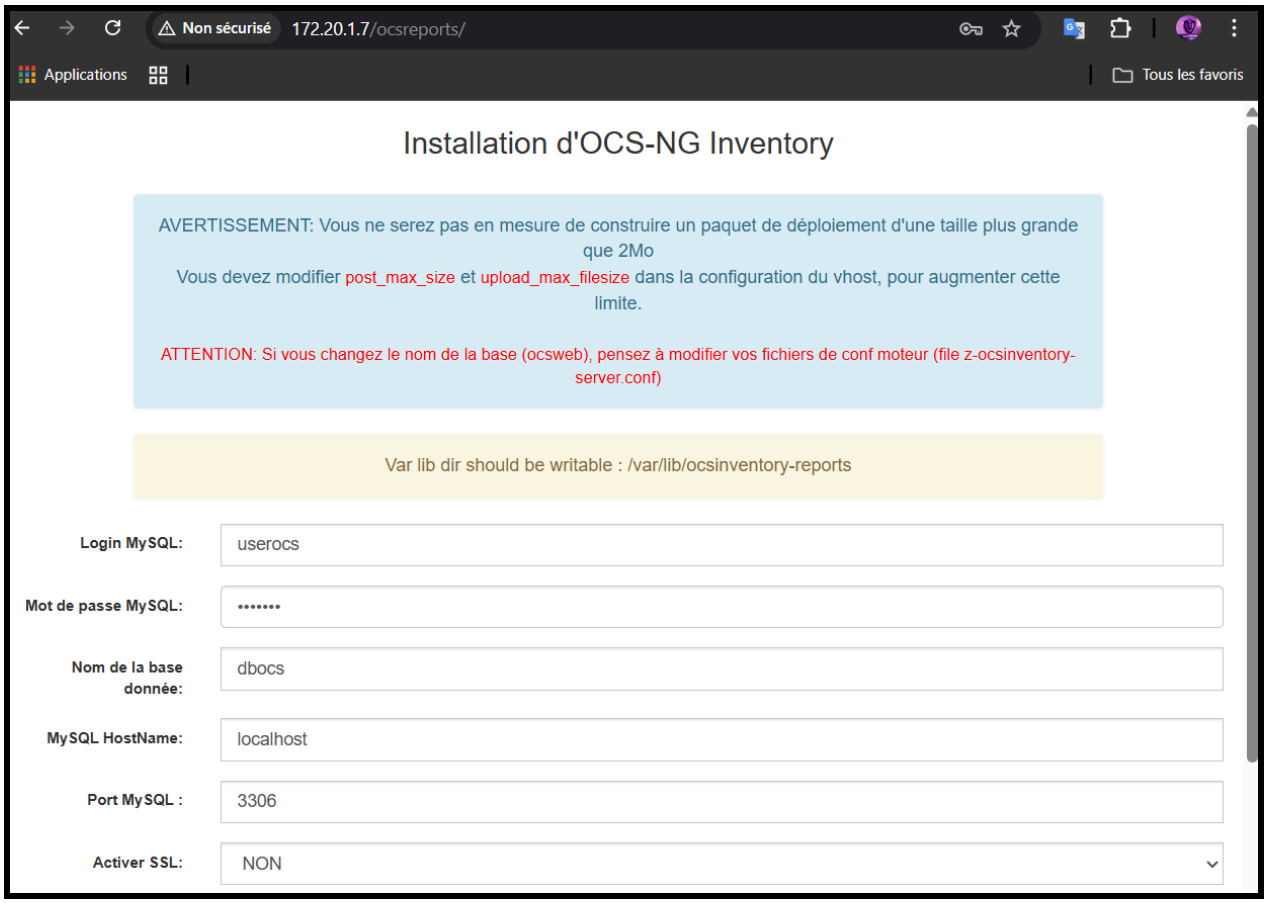
Nous allons donc déplacer le fichier **ocsinventory-reports.conf** dans le dossier **sites-available** puis l'activer :

#mv /etc/apache2/conf-available/ocsinventory-reports.conf /etc/apache2/sites-a vailable/ → Déplacement du fichier vers sites-available
#a2ensite ocsinventory-reports.conf --> Activation du fichier de configuration avec a2ensite :
#systemctl restart apache2 --> Redémarrer pour appliquer les changements

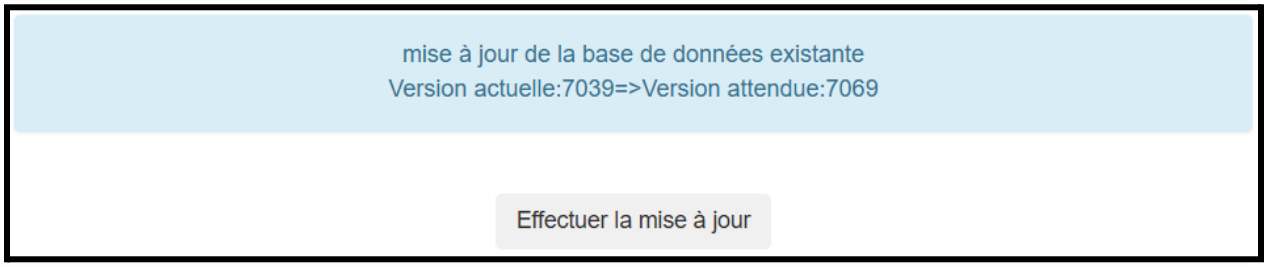
Installation d'OCS sur l'interface web

Sur le navigateur, entrer l'adresse donnée par la console ocs, en ajoutant l'adresse IP :
http://172.20.1.7/ocsreports/

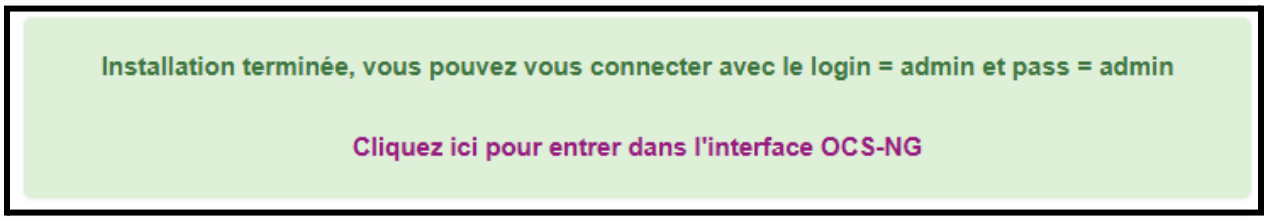
Remplir les champs de cette manière :



Effectuer la mise à jour si elle est affichée :



La base de données est validée et l'installation terminée, nous pouvons nous connecter :



Cette alerte apparaît :

Pour cette alerte de sécurité, renommer le fichier install.php en .install.php :

```
┌──(root💀ocs)-[/etc/apache2/conf-available]
└─# cd /usr/share/ocsinventory-reports/ocsreports/

┌──(root💀ocs)-[/usr/share/ocsinventory-reports/ocsreports]
└─# ls
ajax            config            favicon.ico         js          templates
ajax.php        Contributors      files               libraries   themes
backend         crontab           image               LICENSE     tools
Changes         css               index.php           plugins     update.php
composer.json   dbconfig.inc.php  install.php         README.md   var.php
composer.lock   extensions        ipdiscover-util.pl  require     vendor

┌──(root💀ocs)-[/usr/share/ocsinventory-reports/ocsreports]
└─# mv install.php .install.php

┌──(root💀ocs)-[/usr/share/ocsinventory-reports/ocsreports]
└─# ls
ajax            config            favicon.ico         libraries   themes
ajax.php        Contributors      files               LICENSE     tools
backend         crontab           image               plugins     update.php
Changes         css               index.php           README.md   var.php
composer.json   dbconfig.inc.php  ipdiscover-util.pl  require     vendor
composer.lock   extensions        js                  templates
```

Accéder à OCS directement via le nom de domaine

Le nom de domaine stadiumcompany.local va remplacer l'adresse IP dans l'URL du navigateur pour accéder à OCS. Pour cela nous devons créer un enregistrement OCS sur notre serveur DNS Hermès :

Une fois l'enregistrement créé, nous allons modifier le fichier ocsinventory-reports.conf et y ajouter un bloc Virtual host :

```
<virtualhost *:80>
Servername ocs.stadiumcompany.local
DocumentRoot /usr/share/ocsinventory-reports/ocsreports/
</virtualhost>
```

a2ensite ocsinventory-reports.conf → On active le fichier

systemctl restart apache2 → On redémarre

Test de l'inventaire des machines avec OCS Inventory

Le but est d'installer et configurer l'agent OCS Inventory sur différentes plateformes (Linux, Windows, Android) et de s'assurer que les machines remontent bien dans l'interface web d'OCS.

Installation de l'agent OCS Inventory

Sur une machine Linux (Debian, Ubuntu, CentOS…), on doit installer l'agent ocsinventory-agent qui va collecter et envoyer les informations système au serveur OCS.

SUR LINUX :

#apt update && sudo apt upgrade -y → on met à jour
 # install ocsinventory-agent -y → Installer l'agent OCS inventory
Configurer l'agent pour communiquer avec OCS

Lors de l'installation, il sera demandé de choisir la méthode de communication.
Choisir http et entrer l'URL de notre serveur OCS :

http://172.20.1.7/ocsinventory

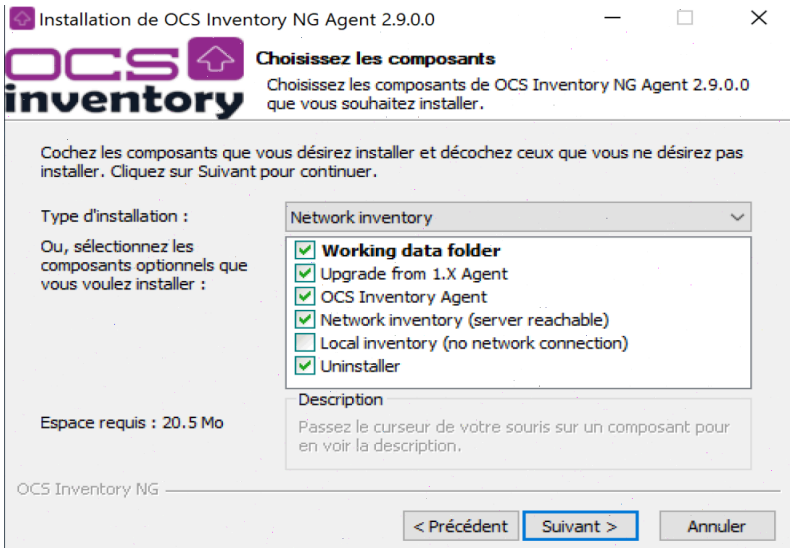
#ocsinventory-agent → Exécuter l'agent manuellement, pour votre machine Linux apparaîtra dans l'interface web d'OCS Inventory.
#dpkg-reconfigure ocsinventory-agent → Pour configurer l'agent si rien ne remonte

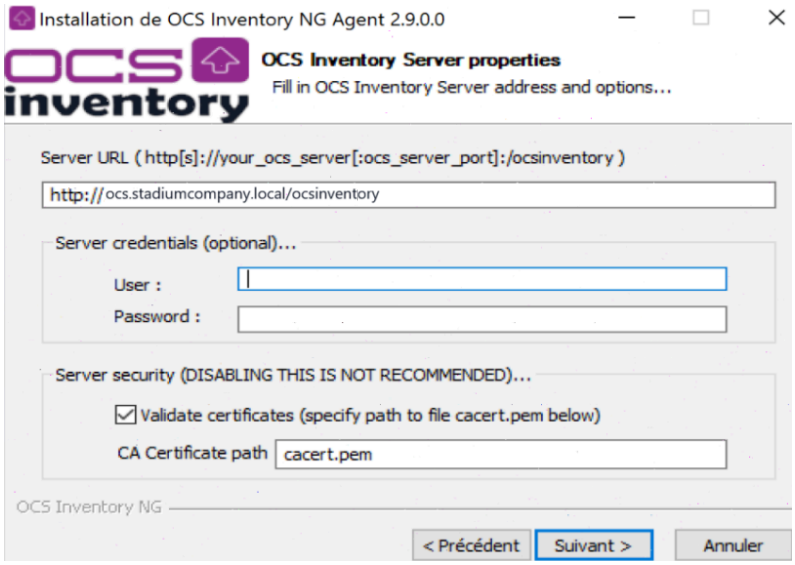
SUR WINDOWS :
1Téléchargez l'agent OCS Inventory pour Windows avec ce lien :

https://github.com/OCSInventory-NG/WindowsAgent/releases/download/2.9.0.0/OCS-Windows-Agent-2.9.0.0_x64.zip

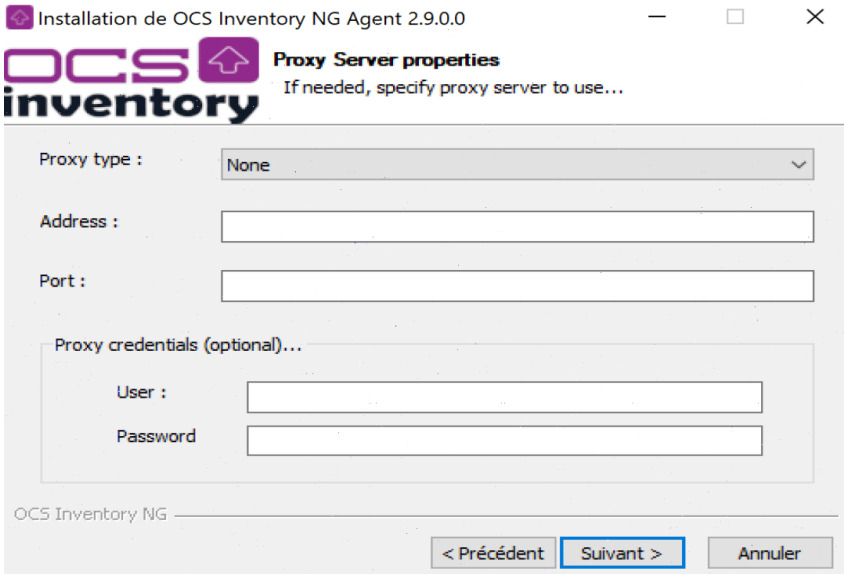
Décompresser le fichier téléchargé → Lancer l'installation en exécutant setup.exe

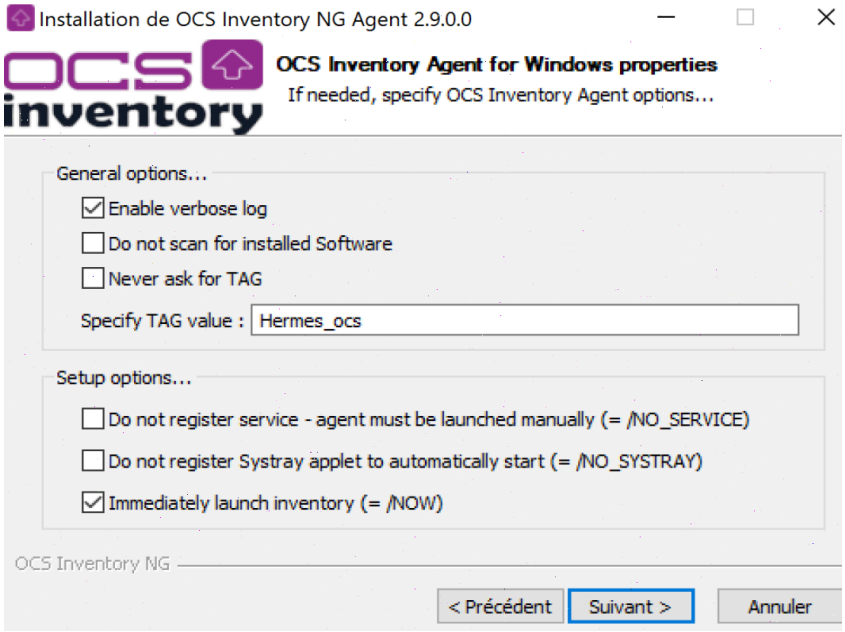
Laisser tout par défaut :

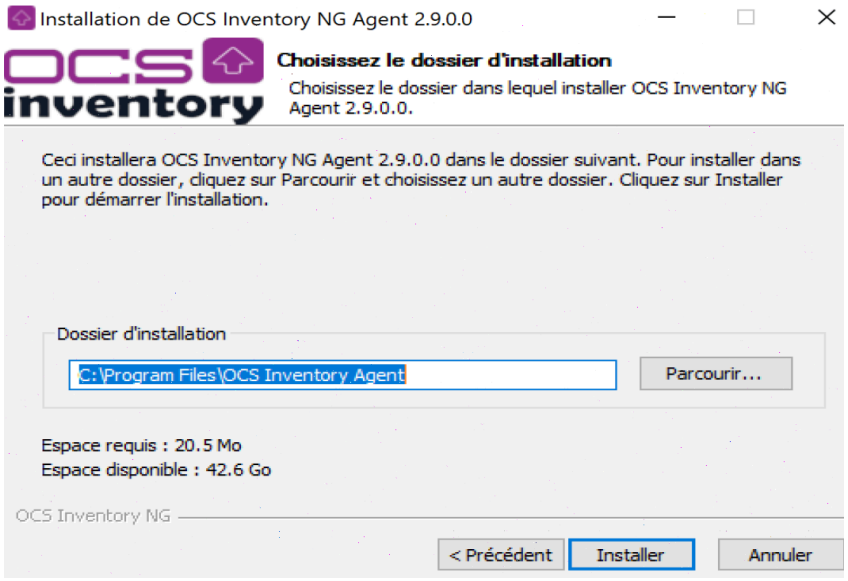




Nous n’avons aucun proxy :


Installation de OCS Inventory NG Agent 2.9.0.0
Proxy Server properties
If needed, specify proxy server to use...
Proxy type :
None
Address :
Port :
Proxy credentials (optional)...
User :
Password
OCS Inventory NG
< Précédent
Suivant >
Annuler


Installation de OCS Inventory NG Agent 2.9.0.0
OCS Inventory Agent for Windows properties
If needed, specify OCS Inventory Agent options...
General options...
Enable verbose log
Do not scan for installed Software
Never ask for TAG
Specify TAG value :
Hermes_ocs
Setup options...
Do not register service - agent must be launched manually (= /NO_SERVICE)
Do not register Systray applet to automatically start (= /NO_SYSTRAY)
Immediately launch inventory (= /NOW)
OCS Inventory NG
< Précédent
Suivant >
Annuler



faites un clic droit sur l'icône OCS Inventory (dans la barre des tâches) et sélectionner :
Exécuter l'agent OCS Inventory maintenant

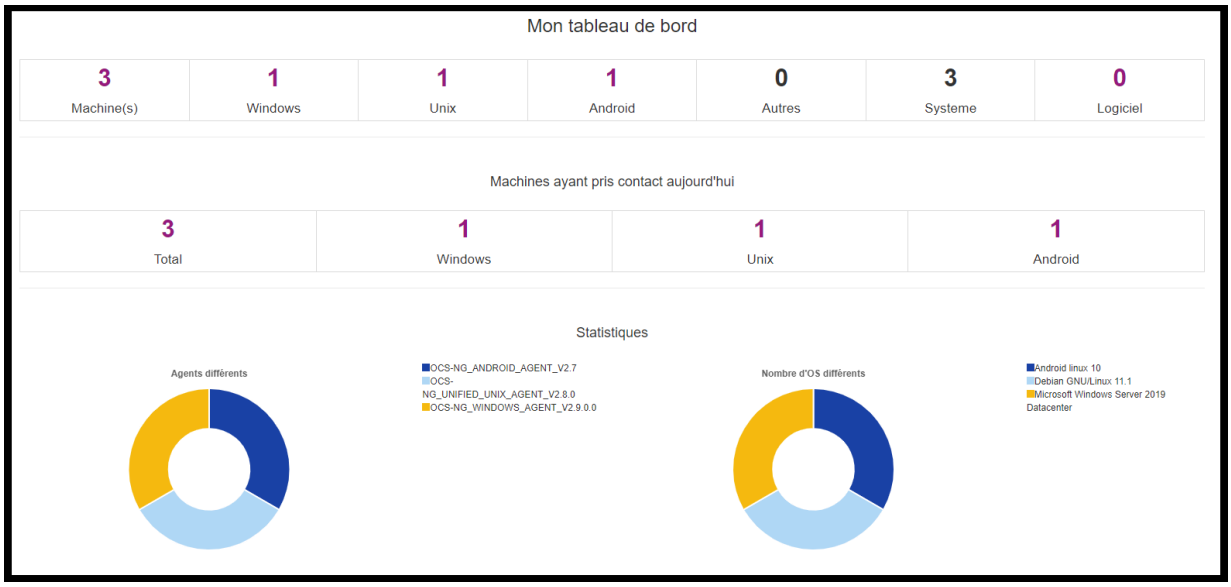
SUR ANDROID :

- Pour que le smartphone puisse accéder au serveur OCS, il faut s'assurer qu'il est bien sur le même réseau.
- Ajouter une deuxième carte réseau en mode "Bridge" à la machine virtuelle OCS : cela permettra d'y accéder depuis un périphérique externe comme un téléphone.
- Connecter le smartphone au même réseau Wi-Fi que le serveur OCS

Puis, sur l'appareil android :

- Téléchargez l'agent OCS Inventory sur le **Play Store**
- Recherchez **"OCS Inventory Agent"** et installez l'application.
- Ouvrir l'application et entrer l'adresse de votre serveur OCS :
http://172.20.1.7/ocsinventory
- Envoyez l'inventaire en cliquant sur **"Envoyer"**

Après installation, l'inventaire remonte les endpoints concernés :



Présentation de GLPI

Solution open-source de gestion de parc informatique et de servicedesk, GLPI est une application Full Web pour gérer l'ensemble des problématiques de gestion de parc informatique : de la gestion de l'inventaire des composantes matérielles ou logicielles d'un parc informatique à la gestion de l'assistance aux utilisateurs.

GLPI possède des fonctionnalités à forte valeurs ajoutées :

- Gestion et suivi des ressources informatiques
- Gestion et suivi des licences
- Gestion et suivi des consommables
- Base de connaissances
- Gestion des réservations
- Service Desk (helpdesk, SLA..)
- Inventaire automatisé
- Télé déploiement

L'utilisation conjointe de la solution d'inventaire OCS Inventory NG (ou de la suite de plugins FusionInventory) permet des avantages importants pour la structure de Stadium, notamment :

- Réduction des coûts
- Optimisation des ressources
- Gestion rigoureuse des licences

- Démarche qualité
- Satisfaction utilisateur
- Sécurité

Diffusé sous licence libre GPL, GLPI est disponible gratuitement, et se révèle être une solution rapide à déployer et simple d'utilisation :

- Les prérequis techniques sont minimes
- La mise en production est immédiate
- Accessible depuis un simple navigateur Web
- Interface paramétrable
- Utilisation intuitive
- Ajout aisé de fonctionnalité grâce à un système de plugins
- Communication possible avec des annuaires existants

La VM GLPI sera configurée sur Debian 12 (Bookworm) en CLI, sans l'environnement graphique. De cette manière, la surface d'attaque est réduite et les performances machines sont améliorées.

Contexte d'utilisation

Stadium possède de nombreux postes et équipements réseaux dans sa nouvelle infrastructure qui en plus d'être complexe, se répartit sur plusieurs sites. L'inventaire automatisé des équipements permettra de recenser efficacement les ressources matérielles, tout en enregistrant les configurations actuelles des équipements, ce qui favorise la conformité entre les sites.

Avec son réseau complexe répartis à travers les différents VLANs déjà mis en place et les différents sites, de nombreux incidents sont susceptibles de se produire (problème de bande passante, interruption de service), nécessitant une prise en charge rapide et efficace. GLPI se révèle particulièrement efficace dans ce contexte en centralisant les demandes des utilisateurs, via un signalement sur le portail utilisateur de GLPI. Cette gestion de tickets permet un suivi simplifié, une priorisation des incidents et évite ainsi des interruptions prolongées. La rzpports d'incidents permettent d'identifier les problématiques majeures ou récurrentes, pour une maintenance améliorée dans la durée.

La gestion documentaire proposée par GLPI, permettra de répondre aux exigences du cahier des charges qui insiste sur la documentation des solutions retenues. La planification des tâches s'opère également avec un module de gestion de projet intégré.

L'environnement de Stadium étant homogénéisé avec des équipements CISCO, il devient crucial de suivre les licences des équipements ainsi que les contrats de

maintenance avec les fournisseurs, afin de maintenir l'ensemble de l'infrastructure à jour et sécurisée.

Ainsi, avec ses 3 sites distincts, GLPI permet une vue centralisée des ressources mais aussi des incidents sur l'ensemble des sites. En favorisant une communication fluide entre les équipes techniques de chaque site grâce à son interface collaborative, GLPI se révèle un outil puissant et adaptable.

Pré-installation de GLPI

Vérifier la connexion internet

Avant toute chose, on vérifie la connectivité réseau de notre VM :
#ping 1.1.1.1 → Ping le DNS public de Cloudflare en envoyant des paquets ICMP (ping) à ce serveur pour vérifier si la VM peut atteindre Internet.

Vérifier la résolution DNS

#ping www.google.com → Vérifie la résolution de nom de domaines
#dig → Sans arguments, effectue une requête par défaut, permettant de vérifier quels serveurs DNS root sont utilisés pour la résolution
#dig www.google.com → interroge les serveurs DNS pour obtenir les enregistrements DNS associés au domaine spécifié, comme les adresses IP
#nslookup www.google.com → Interroge les serveurs DNS afin d'obtenir des informations sur la résolution de noms de domaines (adresse IP)outil de diagnostic DNS qui permet d'interroger les serveurs DNS pour obtenir des informations sur la résolution des noms de domaine

Mettre à jour la machine

apt update → si tout est à jour pas besoin de faire apt upgrade, si non on installe les paquets avec apt upgrade

Changer le hostname

#hostnamectl set-hostname glpi → Renommer la VM

Mettre en place SSH

Nous allons installer SSH server sur la machine puis nous y connecter depuis notre ordinateur physique. Cela est plus pratique, puisqu'en se connectant sur la machine avec SSH, il est possible de copier/coller depuis notre terminal sur l'hôte physique, ce qui n'est pas possible directement sur la VM :

#apt install openssh-server

Autoriser le compte root à se connecter en ssh

Nous devons pour cela modifier le fichier **sshd_config** qui contient les informations de configuration SSH :

#cd /etc/ssh
#nano sshd_config

On va jusqu'à la ligne :
#PermitRootLogin_prohibit-password
On enlève le **#** et **prohibit-password** et on entre **yes, puis** l'on enregistre :

```
#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Redémarrer SSH

Dès qu'on touche au fichier de configuration de SSH, il faut redémarrer le service
#service ssh restart → Redémarrer SSH
#service ssh status → Vérifier que le service a bien redémarré

Changer l'adressage IP dynamique en statique

#nano /etc/network/interfaces → Modifier
En bas du fichier à la ligne qui commente :

#The primary network interface
Effacer la mention **dhcp** de la carte réseau et entrer **static,** et préciser l'adresse ce qui donne :

iface ens33 inet static
address 172.20.1.8/24
gateway 172.20.1.1

```
# The primary network interface
allow-hotplug ens33
iface ens33 inet static
address 172.20.1.8/24
gateway 172.20.1.1
```

Réactiver la carte

Pour appliquer les changement, on désactive puis réactive la carte :
#ifdown ens33
#ifup ens33
#ip a → vérifier la nouvelle adresse IP statique

Vérifier la connectivité réseau

#ping 172.20.1.1 → on PING l'interface LAN-server de pfsense
Si cela ne fonctionne pas, il faut vérifier dans les paramètres machine que la carte réseau est dans le bon LAN_server ou que pfsense (Heimdall) est bien allumé !

Se connecter en SSH depuis un terminal sur l'hôte physique

On PING le serveur GLPI avec le terminal puis on se connecte :
#ping 172.20.1.8
#ssh root@172.20.1.8

S'il y a un pb de clé, on efface l'ancienne clé dans le fichier known_host (sur l'hôte physique) :
#ssh-keygen -f "/home/user/.ssh/known_host" -R "172.20.1.8"
ou simplement
#ssh-keygen -R 172.20.1.8
On accepte la nouvelle clé et on entre les credentials
(Sur Linux, il suffit de supprimer le fichier known_host qui sera automatiquement regénéré avec la clé publique à jour).
Le chemin : **/home/nom_utilisateur/.ssh/known_hosts**

Installation de GLPI

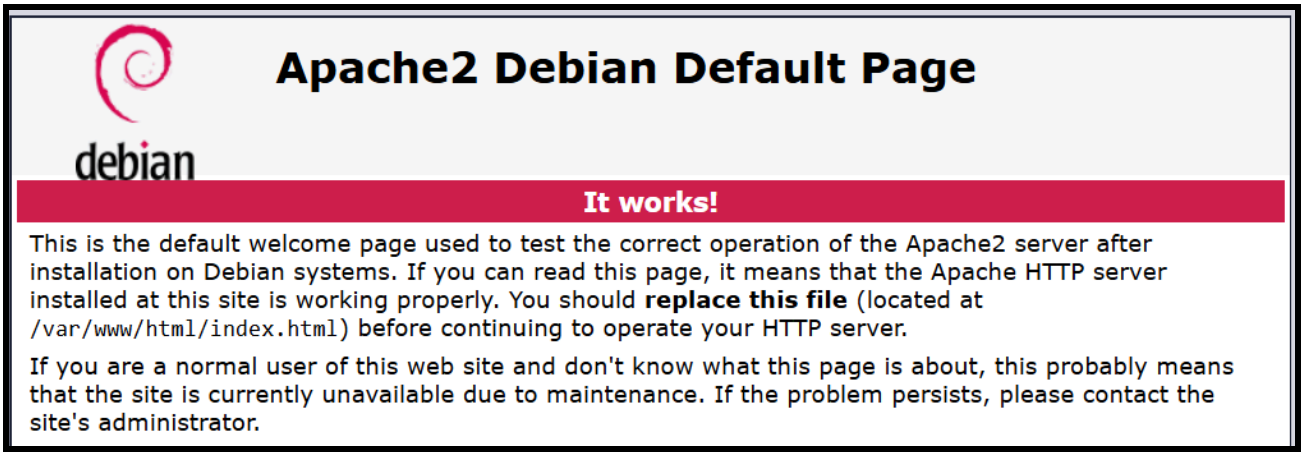
Installer le serveur LAMP (Linux Apache2 MariaDB et Php)

#apt install apache2 php mariadb-server -y
Vérifier le module démarré (si php est démarré en même temps alors prefork est démarré)

Vérifier Apache2

#service apache2 status → Vérifier le fonctionnement du service apache2

On entre l'adresse d'apache (hébergé sur glpi et donc avec l'adresse glpi) dans un navigateur web pour voir si la page fonctionne aussi
http://172.20.1.8



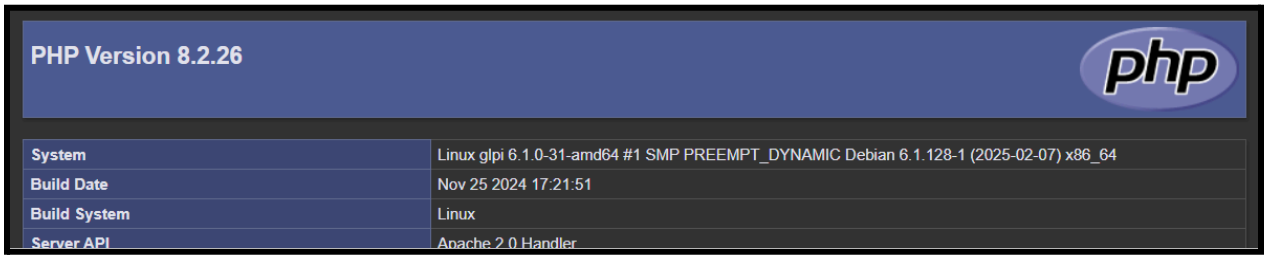
Vérifier php

On va créer une page nommée index.php.
Pour cela 2 manières :

#cd /var/www/html
On créer un fichier index.php puis on y ajoute une commande :
#touch index.php
#nano index.php
<?php phpinfo(); ?> →
#cat → Afficher ce qu'on vient d'écrire
ou
echo "<?php phpinfo(); ?>" > /var/www/html/index.php → On redirige la commande dans un nvx fichier

```
root@glpi:/# cd /var/www/html
root@glpi:/var/www/html# ls
root@glpi:/var/www/html# echo "<?php phpinfo(); ?>" > /var/www/html/index.php
root@glpi:/var/www/html# ls
index.php
root@glpi:/var/www/html# cat index.php
<?php phpinfo(); ?>
root@glpi:/var/www/html#
```

172.20.1.8/index.php → On vérifie sur le navigateur que la page s'affiche :



Si les pages ne s'affichent pas, il s'agit d'un pb de module. Pour les vérifier :

#a2query -m → affiche tous les modules et leurs états
#a2enmod php7.4 → démarre le module s'il n'est pas activé

Sécuriser la BDD SQL

#cd /usr/bin → Tous les exécutables MySQL sont dans un répertoire /bin/
#ls *mysql* → Affiche toutes les commandes qui contiennent mysql
#mysql_secure_installation → On entre cette ligne de commande à repérer pour lancer le processus de sécurisation MySQL

```
root@glpi:/usr/bin# ls *mysql*
msql2mysql                  mysqld_safe_helper    mysqlrepair
mysql                       mysqldump             mysqlreport
mysqlaccess                 mysqldumpslow         mysql_secure_installation
mysqladmin                  mysql_find_rows       mysql_setpermission
mysqlanalyze                mysql_fix_extensions  mysqlshow
mysqlbinlog                 mysqlhotcopy          mysqlslap
mysqlcheck                  mysqlimport           mysql_tzinfo_to_sql
mysql_convert_table_format  mysql_install_db      mysql_upgrade
mysqld_multi                mysqloptimize         mysql_waitpid
mysqld_safe                 mysql_plugin          wsrep_sst_mysqldump
root@glpi:/usr/bin#
```

On répond aux questions de configuration :
Un mdp est demandé mais il n'y en n'a pas donc **entrer** directement
On ne **veut pas** switch vers un socket d'authentification
On va changer le mdp : **root**
On **supprime** l'utilisateur anonyme
On **désactive** les connexions root à distance
On **retire** la BDD de test
On **recharge** les privilèges des tables

Installation des extensions php

Des extensions PHP sont nécessaires afin que GLPI fonctionne correctement.
#apt search ^php- → Liste toutes les extensions déjà existantes

Explication de chaque module :

- **ldap** → Utilisé pour l'authentification avec un annuaire LDAP, notamment Active Directory.
- **apcu** → Active un cache interne pour améliorer les performances de GLPI.
- **mysql** → Permet à GLPI de se connecter et d'interagir avec la base de données MariaDB/MySQL.
- **mbstring** → Gère correctement les caractères multi-octets (accents, caractères spéciaux).
- **curl** → Utilisé pour l'authentification CAS, les mises à jour de GLPI et les requêtes HTTP externes.
- **gd** → Génération et manipulation d'images, utile pour les graphiques et les avatars.

- **xml** → Gestion avancée du XML, utilisé pour l'import/export de données.
- **intl** → Prise en charge des langues, formats de date et monnaies internationales.
- **bz2** → Utilisé pour la compression et la sauvegarde des fichiers.
- **zip** → Gère les fichiers ZIP, utile pour certaines fonctionnalités d'import/export.
- **json** → Indispensable pour le support du format JSON, notamment dans les API de GLPI.
- **cli** → Permet l'exécution de PHP en ligne de commande, nécessaire pour les scripts et tâches automatiques.

#apt install php-{ldap,apcu,mysql,mbstring,curl,gd,xml,intl,bz2,zip,json,cli} -y → Installe toutes les extensions essentielles pour le bon fonctionnement de GLPI, en utilisant l'expansion {} pour installer plusieurs paquets PHP d'un seul coup, ce qui simplifie l'installation.
#php -m | grep -E 'ldap|apcu|mysql|mbstring|curl|gd|xml|intl|bz2|zip|json|cli' → Verifier l'installation des extensions

Il faut aussi adapter certaines valeurs dans le fichier de configuration PHP (php.ini).
#nano /etc/php/8.2/apache2/php.ini → Accéder au fichier php.ini
Le fichier étant long, nous cherchons les termes suivants avec CTRL+ W et les modifier ainsi :

memory_limit = 64M
upload_max_filesize = 100M
file_uploads = On
max_execution_time = 600
session.auto_start = off
session.use_trans_sid =

#systemctl restart apache2 → Redémarre pour appliquer les changements

Créer une BDD et un utilisateur

Les extensions installées, on va créer une BDD et un utilisateur pour y accéder :
#mysql -u root → Nous fait entrer dans l'environnement de création de BDD (MariaDB monitor)
#CREATE DATABASE dbglpi; → Créer la bdd avec le nom spécifié

On va créer un utilisateur nommé **userglpi** et lui donner les privilèges sur cette BDD :

#CREATE USER 'userglpi'@'localhost' IDENTIFIED BY 'userglpi'; → Crée userglpi avec le mot de passe userglpi.
#GRANT ALL PRIVILEGES ON dbglpi.* TO 'userglpi'@'localhost'; → Donne tous les droits sur la base de données dbglpi à userglpi.
FLUSH PRIVILEGES; → Recharge les droits pour prendre en compte les modifications.

```
MariaDB [(none)]> CREATE USER 'userglpi'@'localhost' IDENTIFIED BY 'userglpi';
Query OK, 0 rows affected (0,003 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON dbglpi.* TO 'userglpi'@'localhost';
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,000 sec)
```

Vérifications :

#show databases; → Affiche les tables

#select user,host from mysql.user; → Affiche les utilisateurs (et on voit le nôtre (userglpi))

#exit → Sortir de l'invite de commande de MariaDB pour revenir dans l'environnement Linux

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| dbglpi             |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0,005 sec)

MariaDB [(none)]> select user,host from mysql.user;
+-------------+-----------+
| User        | Host      |
+-------------+-----------+
| mariadb.sys | localhost |
| mysql       | localhost |
| root        | localhost |
| userglpi    | localhost |
+-------------+-----------+
4 rows in set (0,001 sec)
```

Télécharger GLPI

On se rend sur le site GLPI pour le télécharger sur https://glpi-project.org/downloads/ → dans le site, clic droit sur le bouton download en bas de page → copier l'adresse du lien → le coller sur le terminal avec get :

#wget https://github.com/glpi-project/glpi/releases/download/10.0.18/glpi-10.0.18.tgz

#ls → Vérifier le dossier compressé **glpi-10.0.18.tgz**

#tar xzfv glpi-10.0.18.tgz -C /var/www/html → Décompresse le dossier dans /html/

ou

#cp <le dossier>

#cd /var/www/html

#tar xzfv <le fichier>

Changer les droits par défaut

Aller dans le répertoire **/var/www/html** qui contient le dossier décompressé glpi :
 # ls -l → Affiche les droits
Il est indiqué que le compte root et le groupe root ont les autorisations sur le dossier glpi :

```
root@glpi:/var/www/html# ls -l
total 8
drwxr-xr-x 24 user user 4096 12 févr. 11:41 glpi
-rw-r--r--  1 root root   20 12 mars  18:36 index.php
```

Il faut changer cela :
Par sécurité, l'on doit donner la priorité au compte système nommé **www-data**, ainsi qu'au groupe qui se nomme de la même manière. En tant que compte système, il n'autorise aucune connexion utilisateur autre que le compte système paramétré.

On peut vérifier l'existence de ce compte ainsi :
#cd /etc → Le répertoire /etc contient le fichier **passwd** qui contient tous les comptes créés
#nano passwd → Le compte **www-data** apparaît avec la mention **"nologin"** indiquant qu'il est impossible de se connecter avec ce compte, car c'est un compte système.

A l'inverse, tous les comptes finissant par **Bash** autorisent une connexion utilisateur.La commande **chown** sert à changer le propriétaire et le groupe d'un fichier ou dossier.
Nous allons donc donner le contrôle total du dossier **/var/www/html/glpi/** à l'utilisateur et au groupe www-data :

#chown -R www-data:www-data /var/www/html/glpi/ → définit les autorisations d'accès du dossier, de façon récursive pour le compte www-data et le groupe www-data
#ls -l /var/www/html → On voit les changements de permissions effectués

```
root@glpi:/etc# ls -l /var/www/html
total 8
drwxr-xr-x 24 www-data www-data 4096 12 févr. 11:41 glpi
-rw-r--r--  1 root     root       20 12 mars  18:36 index.php
```

#chmod -R 775 /var/www/html/glpi → Définis les droits pour tous les types

Astuce : on calcule les permissions en chiffre en puissances de 2 :
2*2 → 2^1 → 2^0
read write execute
On répète cela pour chaque catégorie…

#systemctl restart apache2 → Redémarrer pour appliquer les changements

Sécuriser le fichier php.ini

Il est recommandé de changer un paramètre dans le fichier **php.ini** concernant les coquilles :
#cd /etc/php/
#ls → Affiche le dossier dans lequel se déplacer

#cd 8.2/
#ls → Le dossier apache2 apparaît et on entre dedans
#cd apache2
#ls → Un dernier ls affiche le fichier qui nous intéresse
#nano php.ini

Dans le fichier avec la recherche CTRL + W :
session.cookie_httponly → On fait une recherche de caractères qui apparaîtra en surbrillance
On modifie la ligne ainsi :
session.cookie_httponly = on

#system restart apache2 → Redémarrer à nouveau après avoir enregistré

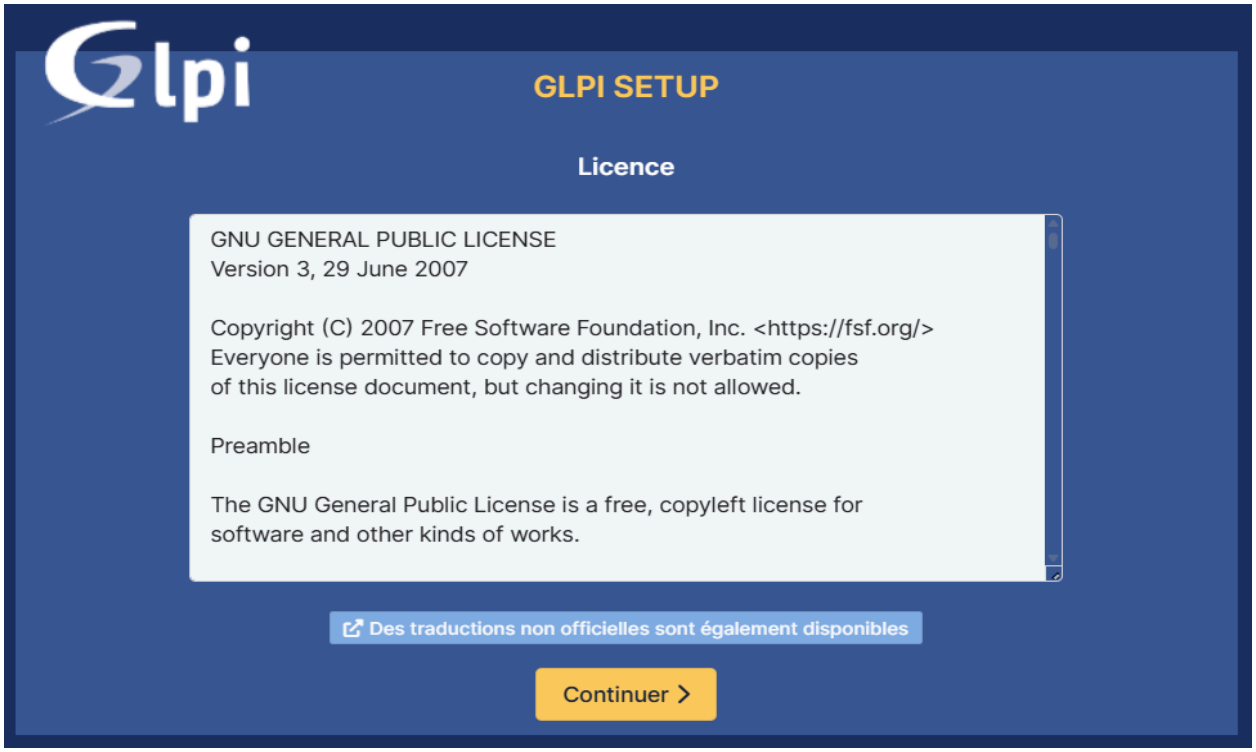
Finaliser l'installation GLPI sur l'interface web

Entrer cette adresse sur un navigateur web http://172.20.1.8/glpi

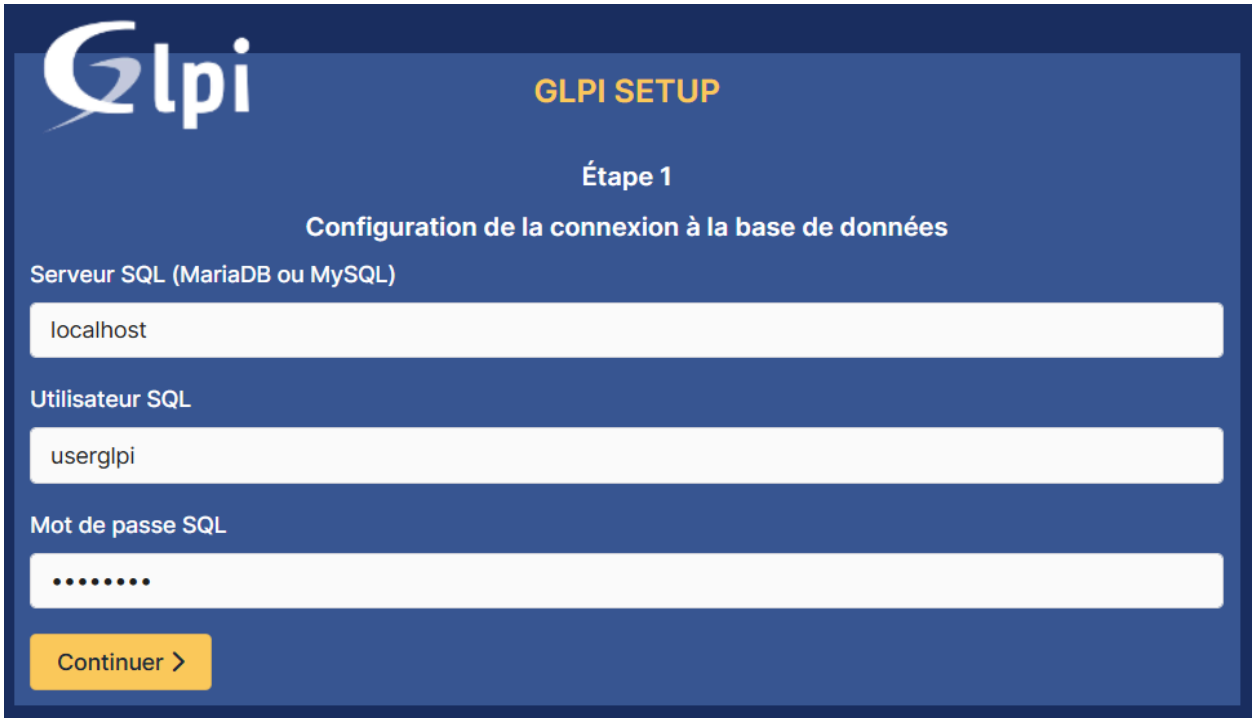
Voici les étapes à suivres :

OK → Continuer → installer → Continuer → On entre le MDP de la BDD sql configuré (userglpi et userglpi) → sélectionner la BDD dbglpi (on peut aussi la créer ici si ce n'est pas encore fait) → Continuer → Choisir de ne pas envoyer les données de statistiques en décochant la case → Continuer → On peut se connecter avec les credentials de base de GLPI (ID: glpi et MDP: glpi)

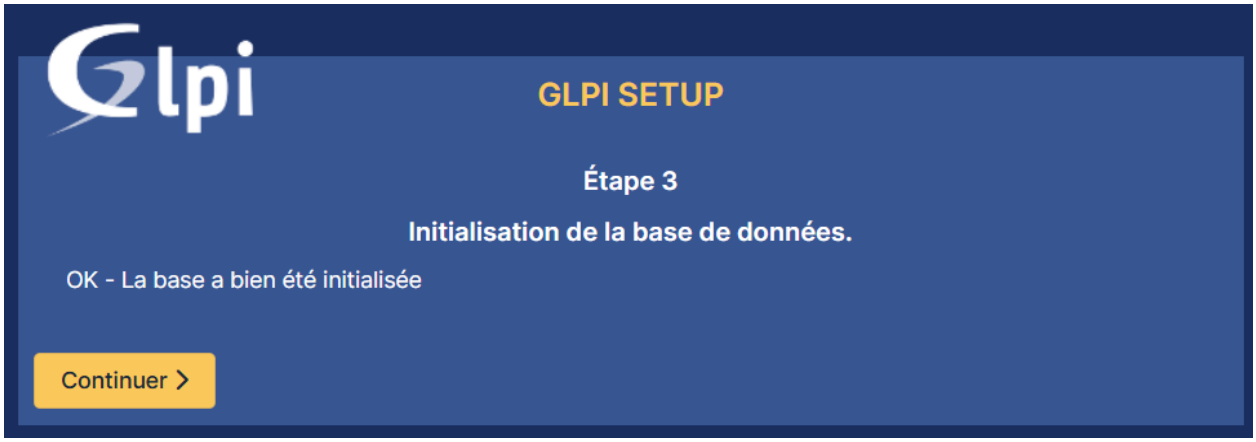
Captures d'écran de l'installation ci-dessous :


GLPI SETUP
Licence
GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007
Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The GNU General Public License is a free, copyleft license for
software and other kinds of works.
Des traductions non officielles sont également disponibles
Continuer


GLPI SETUP
Début de l'installation
Installation ou mise à jour de GLPI
Choisissez 'Installation' pour une nouvelle installation de GLPI.
Choisissez 'Mise à jour' pour lancer la mise à jour de votre version de GLPI à partir d'une version antérieure.
Installer
Mettre à jour


GLPI SETUP
Étape 1
Configuration de la connexion à la base de données
Serveur SQL (MariaDB ou MySQL)
localhost
Utilisateur SQL
userglpi
Mot de passe SQL
••••••••
Continuer >


GLPI SETUP
Étape 2
Test de connexion à la base de données
Connexion à la base de données réussie
Veuillez sélectionner une base de données :
Créer une nouvelle base ou utiliser une base existante :
dbglpi
Continuer >


GLPI SETUP
Étape 3
Initialisation de la base de données.
OK - La base a bien été initialisée
Continuer >

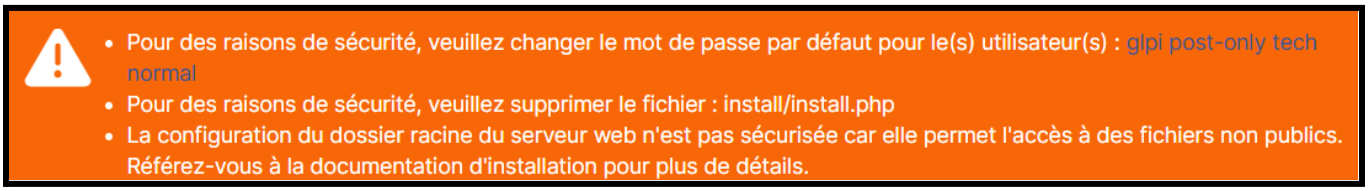

GLPI SETUP
Étape 4
Récolter des données
Envoyer "statistiques d'usage"
Nous avons besoin de vous pour améliorer GLPI et son écosystème de plugins !


GLPI SETUP
Étape 6
L'installation est terminée
Les identifiants et mots de passe par défaut sont :
glpi/glpi pour le compte administrateur
tech/tech pour le compte technicien
normal/normal pour le compte normal
post-only/postonly pour le compte postonly
Vous pouvez supprimer ou modifier ces comptes ainsi que les données initiales.
Utiliser GLPI

Supprimer les avertissements

2 avertissement apparaissent :

Ils demandent de supprimer le fichier **install** et de changer les **mdp par défaut :**



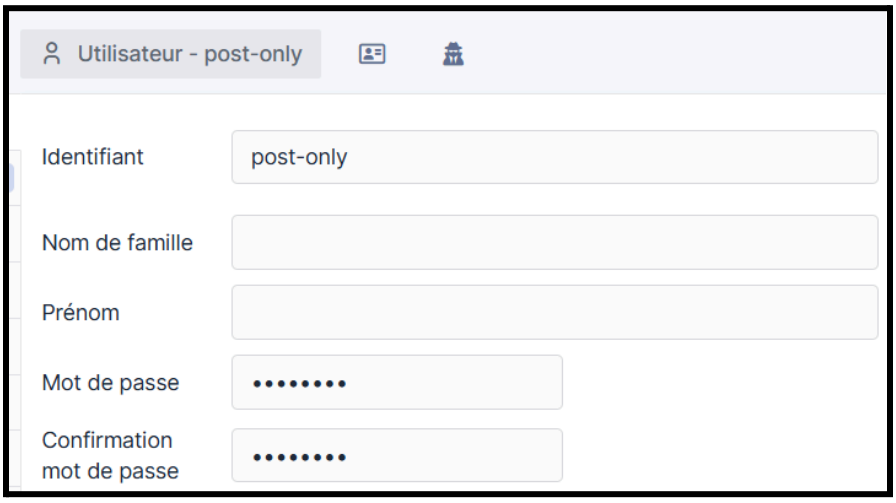
Changer les mdp par défaut :

Lors de l'installation de GLPI, **trois comptes par défaut sont créés** avec un mot de passe générique. Pour des raisons de sécurité, il est **obligatoire** de les modifier.

📌 Comptes par défaut et leur utilité

Nom d'utilisateur	Rôle	Utilisation	Mot de passe par défaut
glpi	**Administrateur**	Accès total à toutes les fonctionnalités (configuration, gestion des tickets, utilisateurs, plugins, etc.).	`glpi`
tech	**Technicien**	Peut voir et gérer les tickets, résoudre des problèmes, accéder aux informations techniques.	`tech`
normal	**Utilisateur standard**	Peut créer des tickets, voir ses propres demandes, accéder à son historique.	`normal`
post-only	**Utilisateur limité**	Peut seulement soumettre des tickets, mais ne voit pas leur suivi ni les réponses.	`postonly`

Par simplicité, le mot de passe sera systématiquement changé en **Bts2024@** pour tous les utilisateurs :



Sauvegarder en bas de page et réitérer l'opération sur les 2 autres comptes.

Changer le fichier install :
Il faut renommer le fichier pour que la plateforme GLPI ne le détecte pas.
En effet, en y accédant, il est possible d'effacer toutes les configurations et de casser le système mis en place. Il est aussi possible de simplement le supprimer.
Pour cela, retournons sur le terminal glpi :

#cd /var/www/html/glpi/install
#ls
#rm install.php → Supprime le fichier
ou
#mv install.php install.phpold → Renomme le fichier, le rendant indétectable.

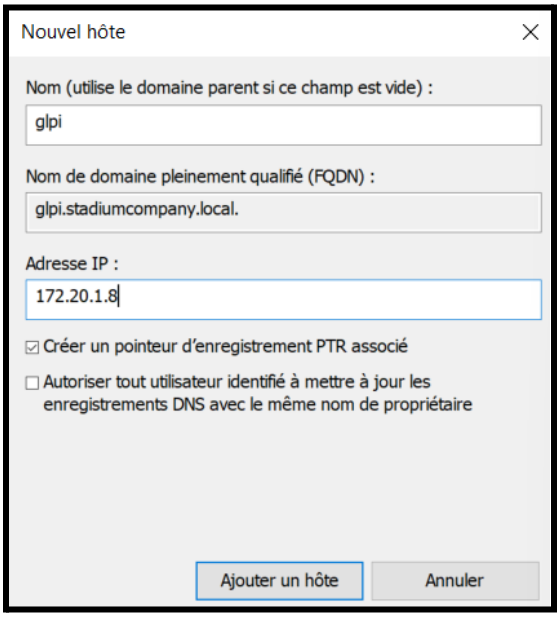
Exploitation de GLPI

L'exploitation de GLPI consiste à le coupler avec l'AD, faire l'identification par mail avec Zimbra, puis créer des comptes en local.

Accéder à GLPI avec un nom de domaine

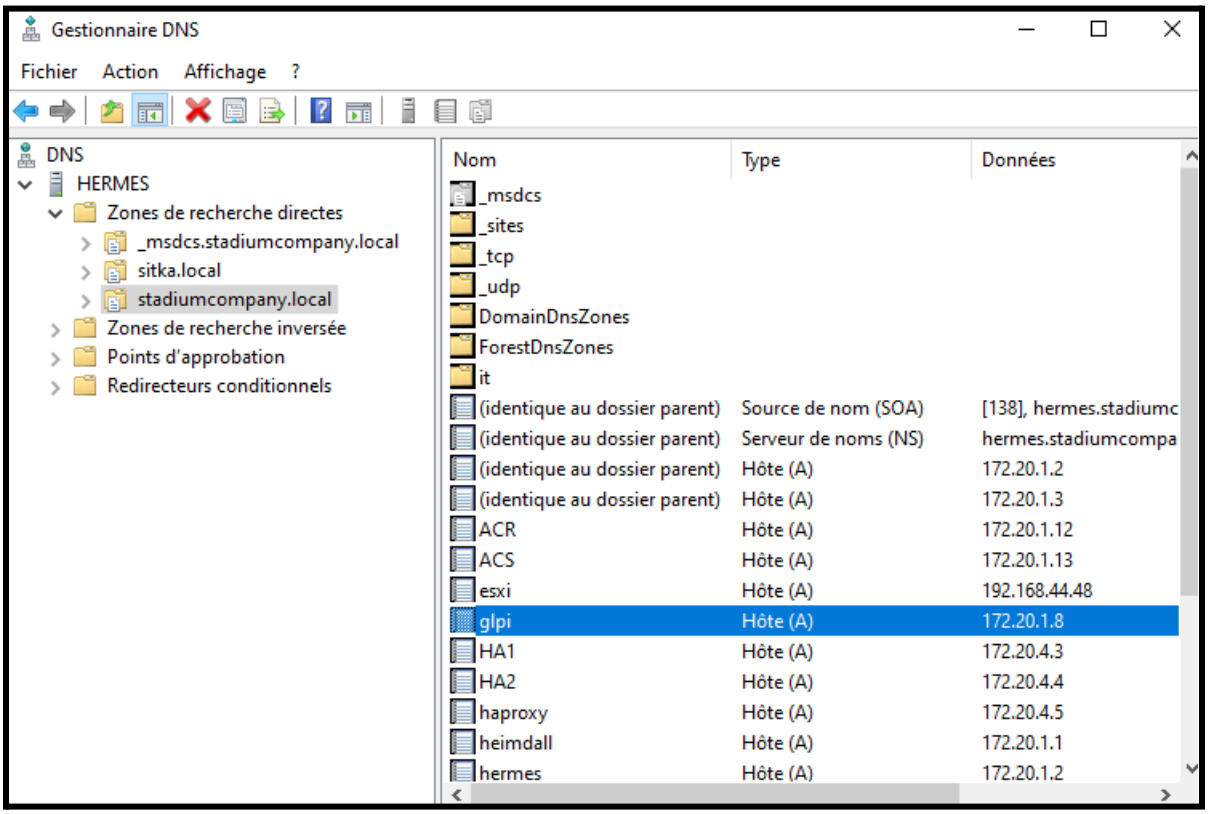
Pour avoir accès à l'interface web glpi avec le nom de domaine, il faut créer un enregistrement de type A sur le serveur DNS sur Hermès :
L'objectif est donc d'ajouter un **enregistrement A** qui associe un nom de domaine (glpi.stadiumcompany.local) à l'adresse IP du serveur GLPI.

Dans Gestionnaire de serveur → Outils → DNS → Zones de recherche directes → Sélectionner la zone DNS : stadiumcompany.local → Clic droit dessus → "Nouvel hôte (A ou AAAA)" → Entrer les informations ainsi :



Vérifications :

Actualiser la zone DNS avec F5



#nslookup glpi.stadiumcompany.local → Vérifier le retour d'IP sur le terminal

```
root@glpi:~# nslookup glpi.stadiumcompany.local
Server:         172.20.1.2
Address:        172.20.1.2#53

Name:   glpi.stadiumcompany.local
Address: 172.20.1.8
```

Configuration du Virtual host sur GLPI

On crée le fichier **glpi.conf** dans le répertoire **/etc/apache2/sites-available**

```
root@glpi:/etc/apache2/sites-available# cd /etc/apache2/sites-available/
root@glpi:/etc/apache2/sites-available# touch glpi.conf
root@glpi:/etc/apache2/sites-available# nano glpi.conf
```

Une fois dans le fichier, on le configure de la sorte :

```
<IfModule mod_ssl.c>
    <VirtualHost *:443>
   ServerName glpi.stadiumcompany.local
   DocumentRoot /var/www/glpi/public

  SSLEngine on
  SSLCertificateFile /etc/ssl/private/stadiumcompany.pem

<Directory /var/www/glpi/public>
      Require all granted
      RewriteEngine On

      RewriteCond %{REQUEST_FILENAME} !-f
      RewriteRule ^(.*)$ index.php [QSA,L]
   </Directory>
</VirtualHost>
</ifmodule>
```

Démarrage d'apache2 et du mode rewrite

#a2enmod rewrite
#systemctl restart apache2

```
root@glpi:/etc/apache2/sites-available# a2enmod rewrite
Enabling module rewrite.
To activate the new configuration, you need to run:
  systemctl restart apache2
root@glpi:/etc/apache2/sites-available# systemctl restart apache2
```

#mv /var/www/html/glpi/ /var/www/ → Nous déplaçons ensuite le répertoire **glpi** vers le dossier **/var/www** :

Sécurisation de l'accès à l'interface glpi avec SSL

D'abord, on doit créer un certificat SSL. Pour ça, vérifions la présence du paquet ssl-cert :

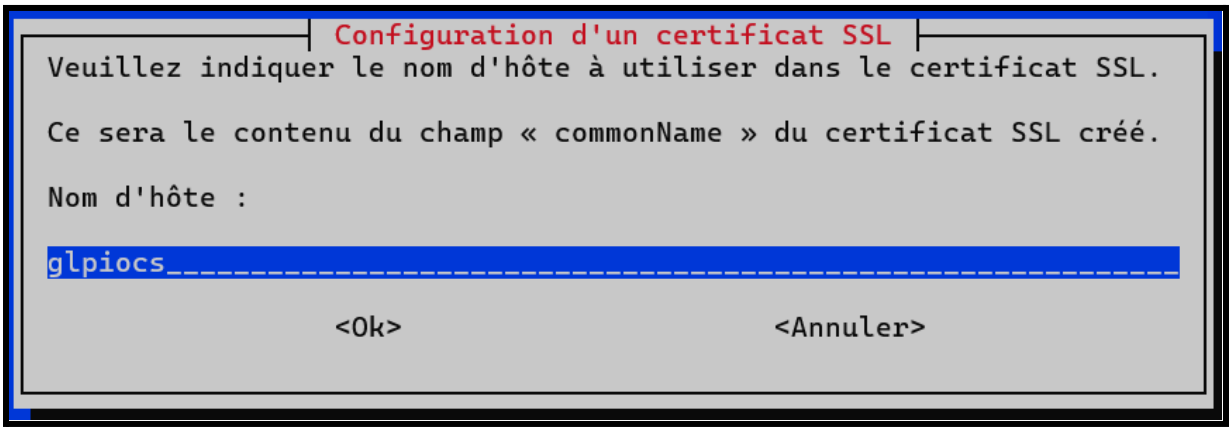
```
root@glpi:~# dpkg -l ssl-cert
Souhait=inconnU/Installé/suppRimé/Purgé/H=à garder
| État=Non/Installé/fichier-Config/dépaqUeté/échec-conFig/H=semi-installé/W=attend-tr>
|/ Err?=(aucune)/besoin Réinstallation (État,Err: majuscule=mauvais)
||/ Nom           Version      Architecture Description
+++-==============-============-============-=================================
ii  ssl-cert       1.1.2        all          simple debconf wrapper for OpenSSL
```

Puis, nous créons un fichier **PEM (Privacy Enhanced Mail)** qui contient un certificat autosigné avec une clée privée :

#make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/ssl/private/stadiumcompany.pem →
#make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/ssl/private/stadiumcompany.pem →

Cette commande génère un **certificat autosigné** avec une **clé privée** en utilisant le fichier de configuration ssleay.cnf. Nous entrons ainsi dans le mode configuration :

Le nom d'hôte → glpiocs



Nom supplémentaire →
DNS:glpi.stadiumcompany.local, DNS:ocs.stadiumcompany.local, IP:172.20.1.8

Dans **/etc/ssl/private**, on vérifie l'existence du fichier **stadiumcompany.pem** avec un **ls.**

Avec un **cat, stadiumcompany.pem** s'affiche et on voit qu'il possède bien un certificat et une clé privé, avec les mentions "**BEGIN PRIVATE KEY**" et "**BEGIN CERTIFICATE**"

```
root@glpi:/etc/ssl/private# cat stadiumcompany.pem
-----BEGIN PRIVATE KEY-----
MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQC6LjNkpSvGWjYo
E6bXbOGvsFQlKC+Hs4SU2YpPqz6sbAoBgijBvzE+qHLn9v7TZP/ASkiKR3fG6WWv
ztxx4YILiipMFxdGh+VcTxbUQOJhGQKjeS8vvQNa49cLQLXuqGOj00wK4ri1mih4
-----END PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIDLTCCAhWgAwIBAgIUEGuP21ZLExRk3fNTrg8QCjlk/DYwDQYJKoZIhvcNAQEL
BQAwEjEQMA4GA1UEAwwHZ2xwaW9jczAeFw0yNTAzMTIyMTE2NTJaFw0zNTAzMTAy
MTE2NTJaMBIxEDAOBgNVBAMMB2dscGlvY3MwggEiMA0GCSqGSIb3DQEBAQUAA4IB
```

On active maintenant le mode SSL

```
root@glpi:/etc/ssl/private# a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and crea
te self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
root@glpi:/etc/ssl/private# systemctl restart apache2
```

On fait pareil pour activer le fichier glpi.conf :

```
root@glpi:/etc/ssl/private# a2ensite glpi.conf
Enabling site glpi.
To activate the new configuration, you need to run:
  systemctl reload apache2
root@glpi:/etc/ssl/private# systemctl restart apache2
root@glpi:/etc/ssl/private#
```

Maintenant, testons notre accès sécurisé à GLPI :

ATTENTION : bien penser à rajouter le DNS d'Hermès sur le PC physique ! En effet, le PC physique n'utilise pa**s Hermès (172.20.1.2)** comme serveur DNS, d'où l'erreur **DNS_PROBE_FINISHED_NXDOMAIN**

Afin qu'Hermès continue de servir uniquement aux VM, sans perturber l'accès internet normal, il faut faire en sorte que le PC puisse utiliser Hermès pour résoudre **glpi.stadiumcompany.local** tout en gardant son accès Internet de base :

ipconfig /all → Sur le PC physique, vérifier la carte réseau à modifier et confirmer le serveur DNS qu'elle indique utiliser :

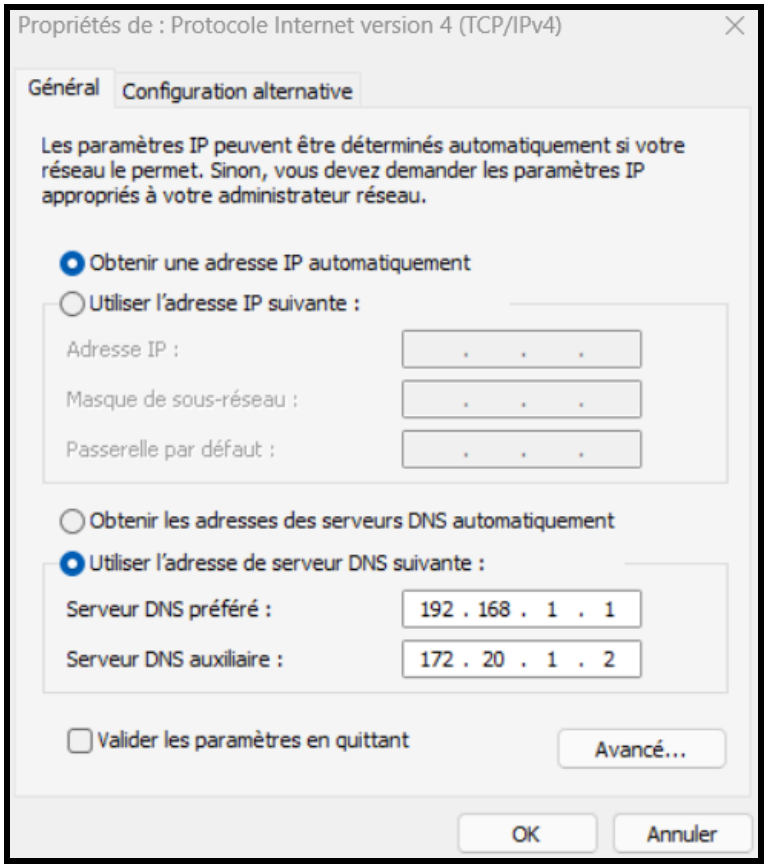
```
Serveurs DNS. . .  . . . . . . . . . . . : 2a01:cb08:a17:3900:12e9:92ff:fe02:4010
                                           fe80::12e9:92ff:fe02:4010%21
                                           192.168.1.1
```

Cela veut dire que le PC envoie toutes ses requêtes DNS à la box internet (192.168.1.1) du réseau, et non à Hermès (172.20.1.2) sur le réseau virtuel.

Nous allons donc ajouter Hermès comme DNS secondaire (sans impacter l'Internet) pour qu'il puisse résoudre glpi.stadiumcompany.local uniquement lorsque nécessaire.

Ouvrir les paramètres réseau → Panneau de configuration → Centre Réseau et partage → Modifier les paramètres de la carte → Clic droit sur la carte Ethernet → Propriétés → Sélectionner Protocole Internet version 4 (TCP/IPv4) → Propriétés :

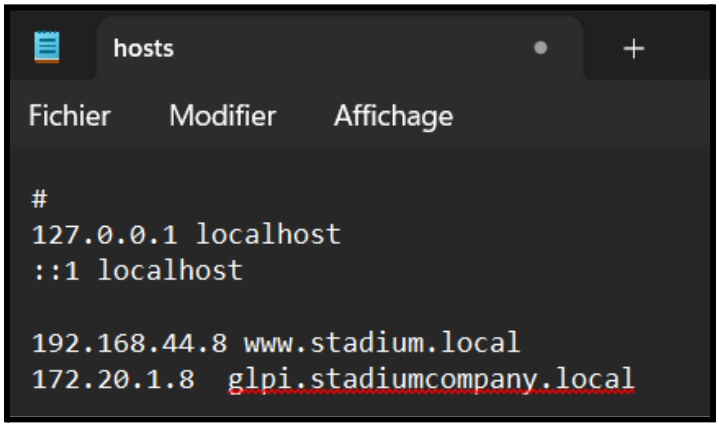
Cocher **"Utiliser les adresses de serveur DNS suivantes"**
Serveur DNS préféré : **192.168.1.1** (la box)
Serveur DNS auxiliaire : **172.20.1.2** (Hermès)



Enfin, nous allons ajouter une résolution locale avec le fichier hosts. Cela permet de forcer la résolution DNS pour **glpi.stadiumcompany.local** en l'ajoutant dans le fichier hosts, et le PC n'aura même plus besoin de passer par un DNS mais ira directement sur **172.20.1.8** :

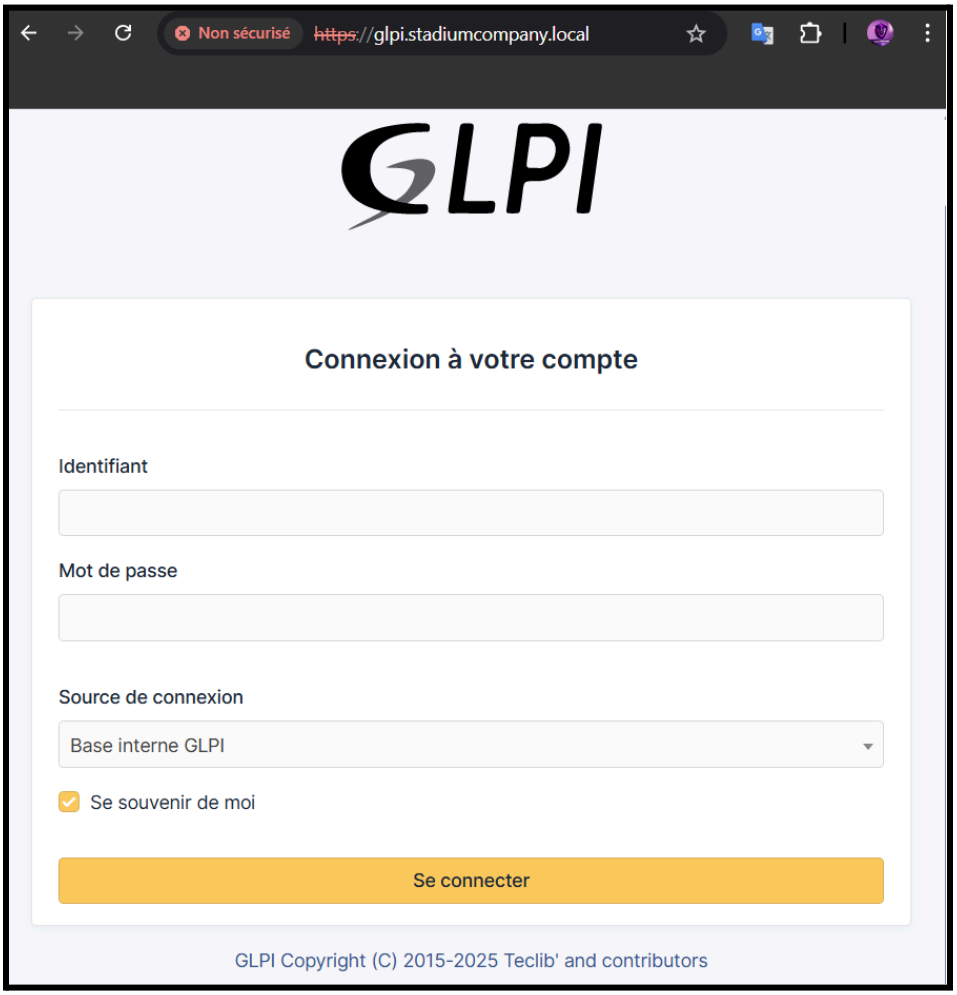
notepad C:\Windows\System32\drivers\etc\hosts → Éditer le fichier hosts

Ajoutons cette ligne à la fin du fichier :
172.20.1.8 glpi.stadiumcompany.local



Enregistrer et fermer.

Nous pouvons à présent nous connecter via https et avec le nom de domaine directement, en entrant l'adresse → **https://glpi.stadiumcompany.local**



Sécurisation de glpi en masquant sa version et l'OS utilisé

Lorsqu'un serveur Apache **envoie ses en-têtes HTTP**, il inclut par défaut des informations **critiques** comme :

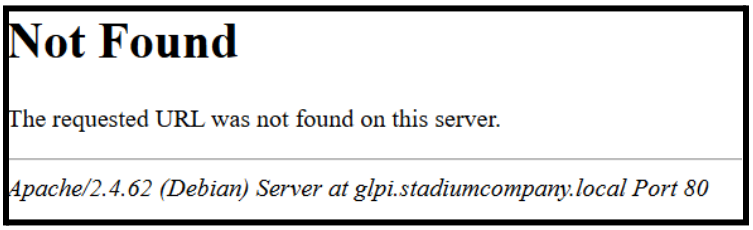
- **Le nom du serveur Web** (Apache)
- **Sa version exacte** (Apache/2.4.58, par exemple)
- **Le système d'exploitation** utilisé (Debian, Ubuntu, Windows, etc.)

Cela représente une faille majeure. En effet, u**n attaquant peut identifier les failles connues** associées à **cette version précise d'Apache et du système**, pour **lancer une attaque ciblée** en exploitant une vulnérabilité **déjà documentée** dans cette version. Cela facilite **les attaques automatisées**, où des scripts scannent des serveurs à la recherche d'anciennes versions vulnérables.

Le problème ? En connectant à distance de n'importe quelle machine sur Linux, il est possible d'afficher ces informations avec la commande curl :
#curl -I 172.20.1.8 → Affiche les informations

```
HTTP/1.1 200 OK
Date: Wed, 12 Mar 2025 21:10:00 GMT
Server: Apache/2.4.58 (Debian)
Content-Type: text/html
```

Dans notre cas, la version d'apache apparaît de cette manière sur le navigateur web :



Il est possible d'afficher ces informations en local :
#apt-cache policy apache2

```
root@glpi:~# apt-cache policy apache2
apache2:
  Installé : 2.4.62-1~deb12u2
  Candidat : 2.4.62-1~deb12u2
 Table de version :
 *** 2.4.62-1~deb12u2 500
        500 http://deb.debian.org/debian bookworm/main amd64 Packages
        500 http://security.debian.org/debian-security bookworm-security/main amd64 Packages
        100 /var/lib/dpkg/status
```

Apache affiche ces infos via ses **fichiers de configuration.** Pour cacher cela, il faut modifier les paramètres dans le fichier **security.conf** :

#nano /etc/apache2/conf-available/security.conf

On désactive la ligne **ServerTokens OS** en rajoutant au début de la ligne un **#**
On désactive la ligne **ServerSignature On** en rajoutant au début de la ligne u**n #**

On redémarre apache pour appliquer les changements :
#systemctl restart apache2

Après test, les informations n'apparaissent plus sur le navigateur web :

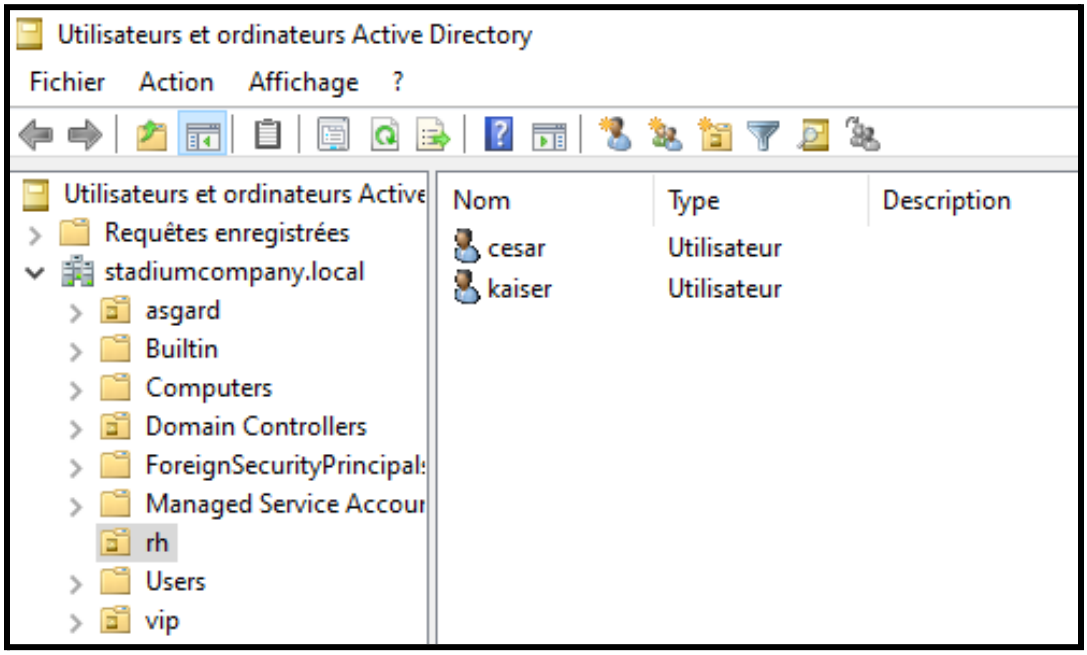
Liaison de glpi avec l'AD

Il faut d'abord créer les UO et les utilisateurs sur le contrôleur de domaine.
Ici, nous allons créer l'UO rh dans laquelle nous allons créer 2 utilisateurs : kaiser et cesar.

Nous devons accéder à **"Utilisateurs et ordinateurs Active Directory"** sur Hermès :

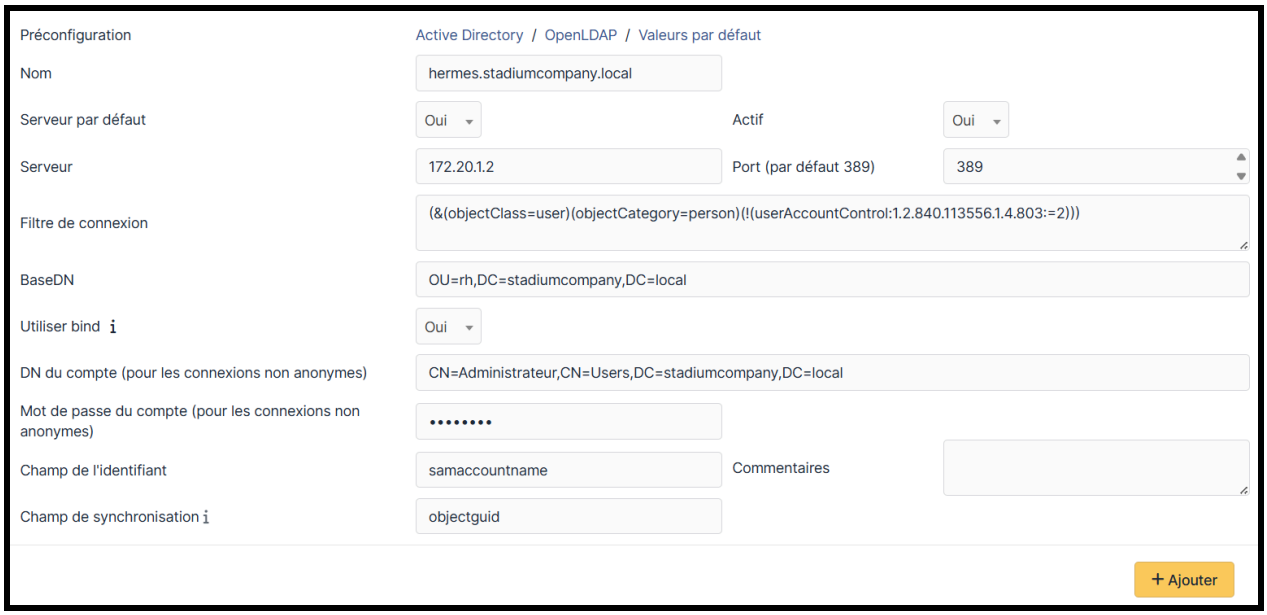
- Ouvrir le Menu Démarrer
- Taper **"Utilisateurs et ordinateurs Active Directory"** et cliquer dessus.
- Dans la console, naviguer jusqu'au domaine **stadiumcompany.local**
- **-** Faire un clic droit sur le domaine pour créer une nouvelle OU.
- Pour créer des **utilisateurs**, faire un clic droit sur l'OU
- Sélectionner **"Nouveau"** → **"Utilisateur"** et remplir les informations.

Une fois fait, ils pourront être utilisés sur GLPI :



Création de la liaison avec l'annuaire LDAP

Sur l'interface web glpi, cliquer sur : Configuration → Authentification → Annuaire LDAP → Ajouter → Il faut remplir les informations de cette manières :

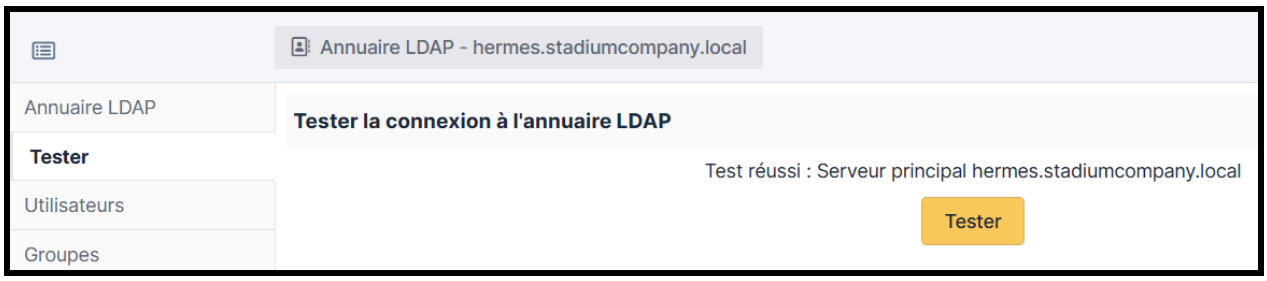


➔ Dans filtre de connexion, appliquer le filtre suivant : **(&(objectClass=user)(objectCategory=person)(!(userAccountControl:1.2.840.113556.1.4.803:=2)))**
C'est un filtre classique qui permet de ne récupérer que les utilisateurs actifs

➔ Dans mot de passe du compte : ajouter le MDP de l'admin du contrôleur de domaine **Bts2024@**

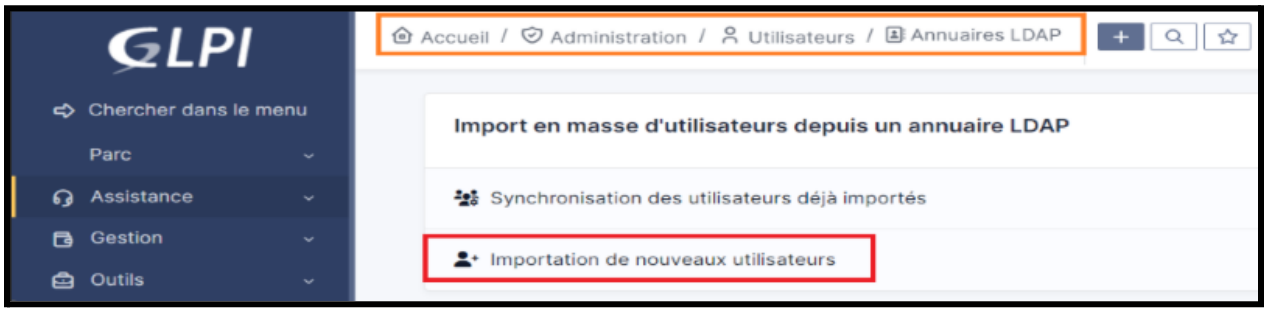
Dans la page suivante, cliquer sur le lien hermes.stadiumcompany.local afin de **tester** la liaison avec l'AD :

Le test est réussit :



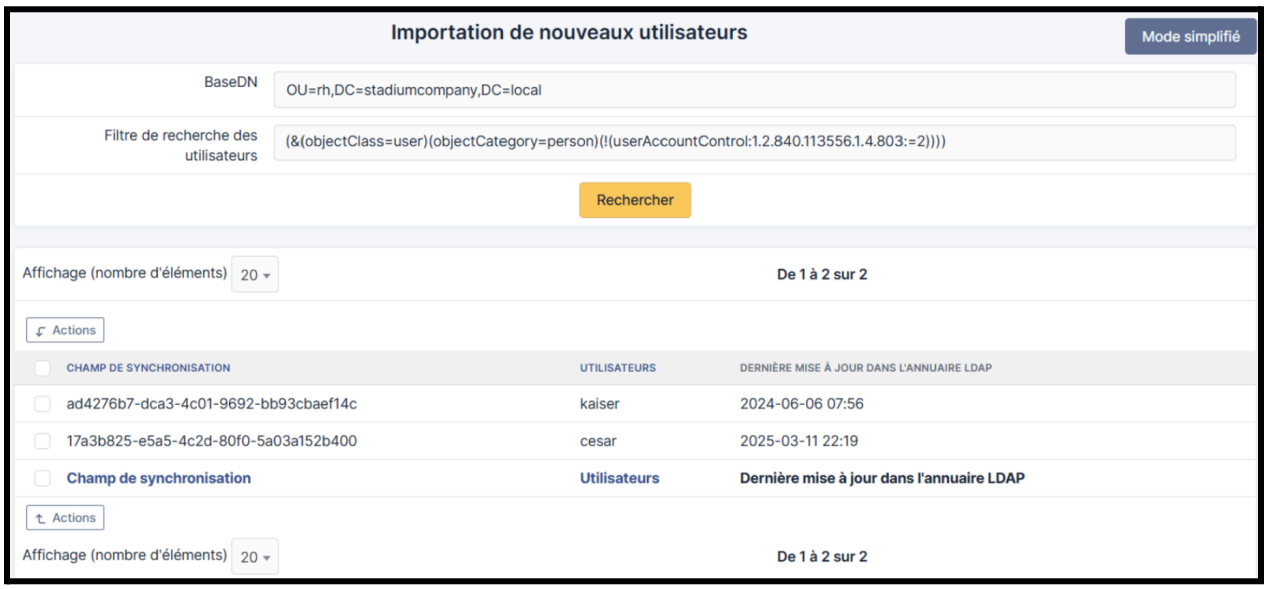
Importation des utilisateurs à partir de la base d'annuaire LDAP

Toujours sur glpi, se rendre sur **: Administration → Utilisateur → Liaison annuaire LDAP → Importation de nouveaux utilisateurs**



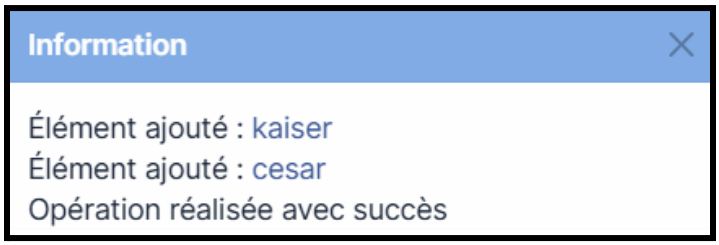
ATTENTION : la gestion des filtres LDAP a été modifiée avec la nouvelle version 10.0.18 de GLPI. Il faut d'abord passer en **Mode Expert** (en haut à droite), puis cliquer sur **Rechercher** :

Les 2 utilisateurs de l'UO rh crée sur l'AD d'Hermès, cesar et kaiser, apparaissent bien. Il faut les **cocher → Actions → Importer → Envoyer**



Action → Importer → Envoyer :

Vérifier la confirmation d'importation :



Pour vérifier la présence des utilisateurs importés dans le menu, se rendre dans **:**
Administration → Utilisateur

Test de connexion LDAP avec glpi

Maintenant que les utilisateurs ont été importés, on va tenter une connexion avec eux via LDAP.

ID : **cesar**
MDP : **Bts2024@**
Source de connexion : **hermes.stadiumcompany.local**

Création de Tickets GLPI

Nous allons mettre en place une fonctionnalité d'alerte en configurant les notifications sur notre serveur glpi, afin d'envoyer des mail à l'administrateur dès qu'un ticket sera créé.
Dans un premier temps, nous allons tester l'envoi de mails par telnet, de notre serveur glpi vers la messagerie Zimbra.

Sur la console glpi entrer :
#telnet xmail.stadiumcompany.local 25 → Envoie un mail de test via Telnet, de GLPI vers ZImbra

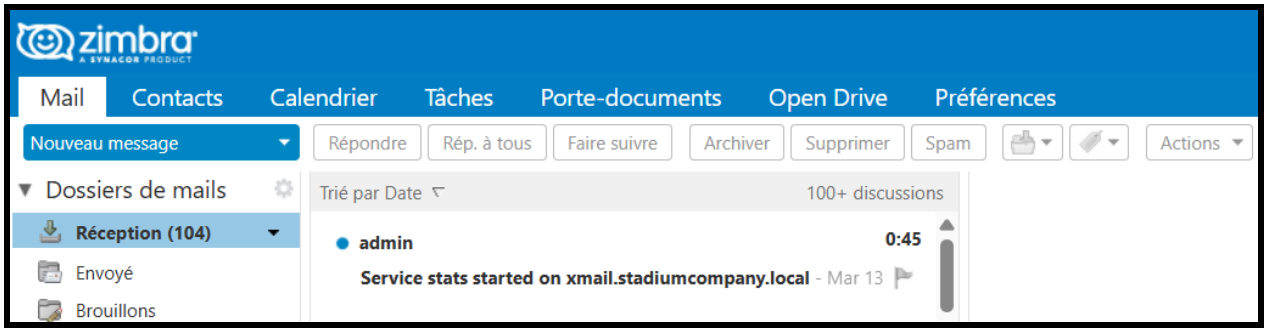
On obtient ce prompt confirmant que le serveur mail est joignable depuis glpi depuis le port 25 :

```
Trying 172.20.4.2...
Connected to xmail.stadiumcompany.local.
Escape character is '^]'.
220-xmail.stadiumcompany.local ESMTP Postfix
220 xmail.stadiumcompany.local ESMTP Postfix
```

Afin de nous assurer du bon fonctionnement de la notification glpi par mail, nous allons directement vérifier sur Zimbra la réception de ce mail :

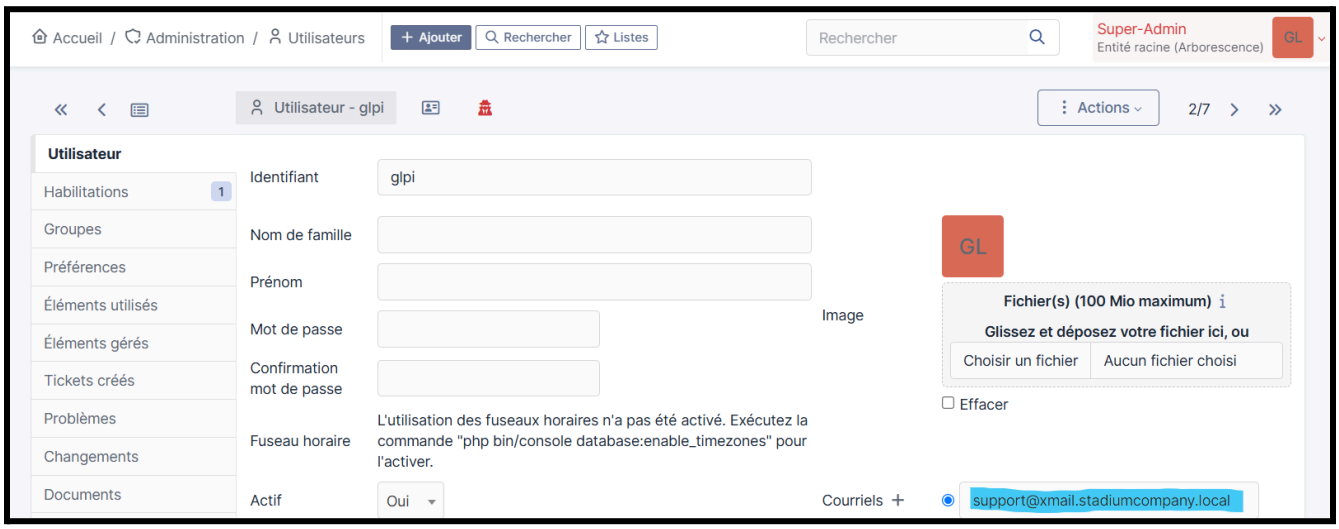
On se rend sur l'interface Zimbra avec → https://172.20.4.2/
On entre les ID et MDP du compte admin de Zimbra :

Le mail est bien réceptionné :

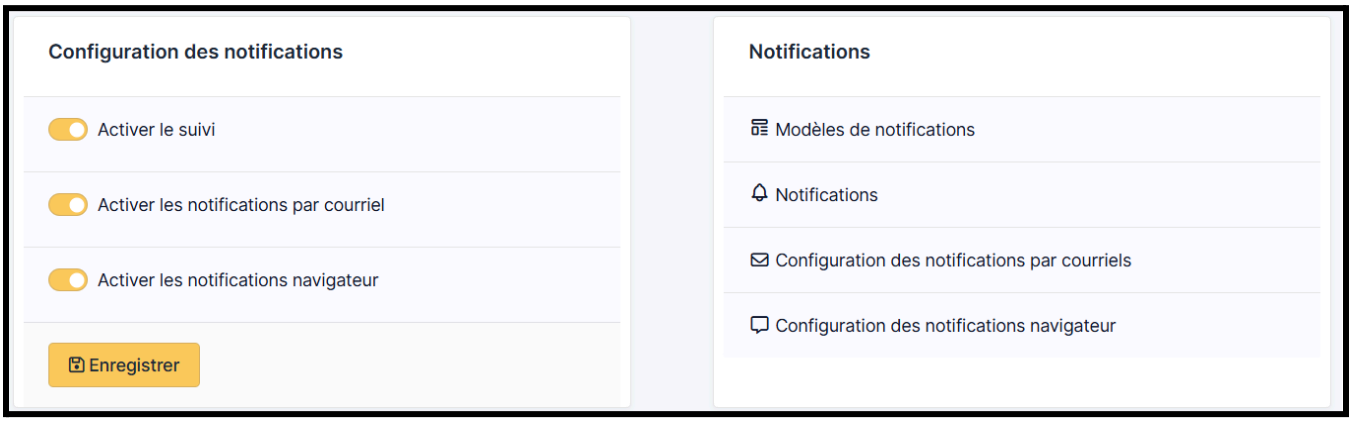


Nous pouvons désormais configurer GLPI pour envoyer les notifications automatiquement, et pour cela GLPI à besoin d'une adresse mail configurée bien sûr :

Se reconnecter sur le compte initial nommé glpi (du localhost bdglpi et pas LDAP) qui est le seul compte administrateur → Administration → Utilisateurs → Sélectionner notre compte administrateur "glpi" :



Puis dans **Configuration → Notification → Activer tous les suivis → Enregistrer →** Puis cliquer sur **"Configuration des notifications par courriels" :**



Remplir les paramètres comme suit :

Notifications courriel

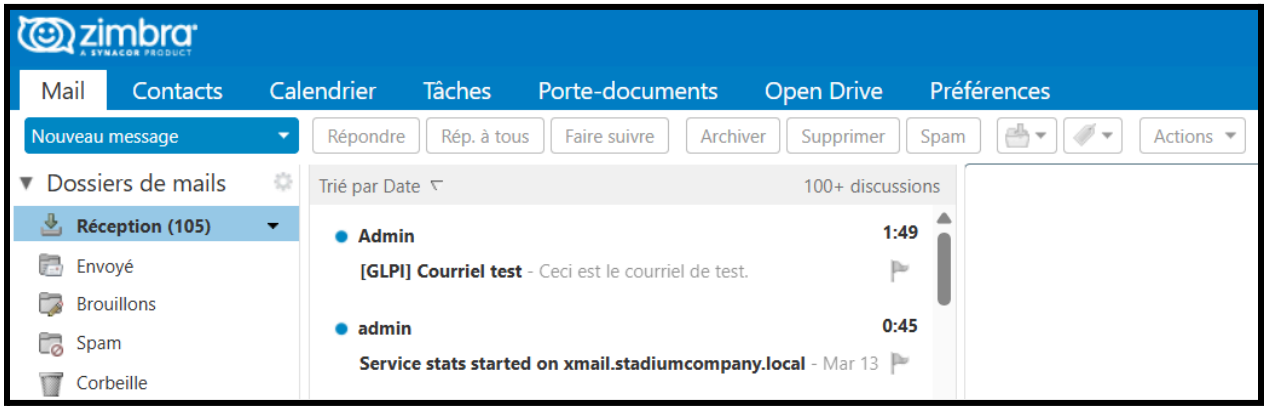
Courriel de l'administrateur	admin@xmail.stadiumcomp	Nom de l'administrateur	Admin
Courriel de l'expéditeur	admin@xmail.stadiumcomp	Nom de l'expéditeur du message	Admin
Adresse de réponse	admin@xmail.stadiumcomp	Nom de réponse	
Adresse de non réponse		Nom de non réponse	
Ajouter des documents dans les notifications de ticket	Oui		
Signature des courriels	Notification envoyée par le centre helpdesk de l'organisation Stadiumcompagny		
Mode d'envoi des courriels	SMTP	Tentatives d'envoi max.	5
Tenter d'envoyer de nouveau dans (minutes)	5		

Serveur de messagerie

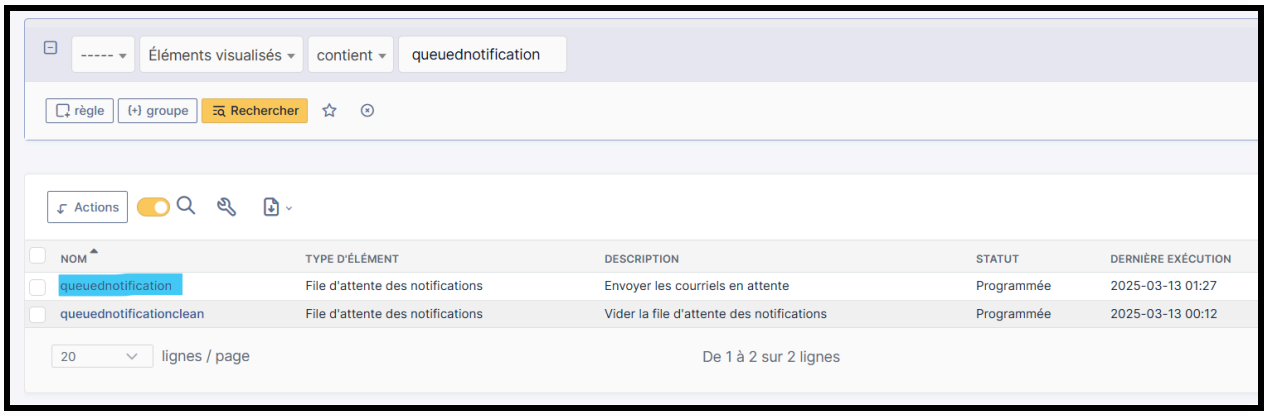
Vérifier le certificat	Non		
Hôte SMTP	xmail.stadiumcompany.local	Port	25
Identifiant SMTP (optionnel)		Mot de passe SMTP (optionnel)	
		Effacer	
Expéditeur du message	admin@xmail.stadiumcompany.local		

Envoyer un courriel de test à l'administrateur | Sauvegarder

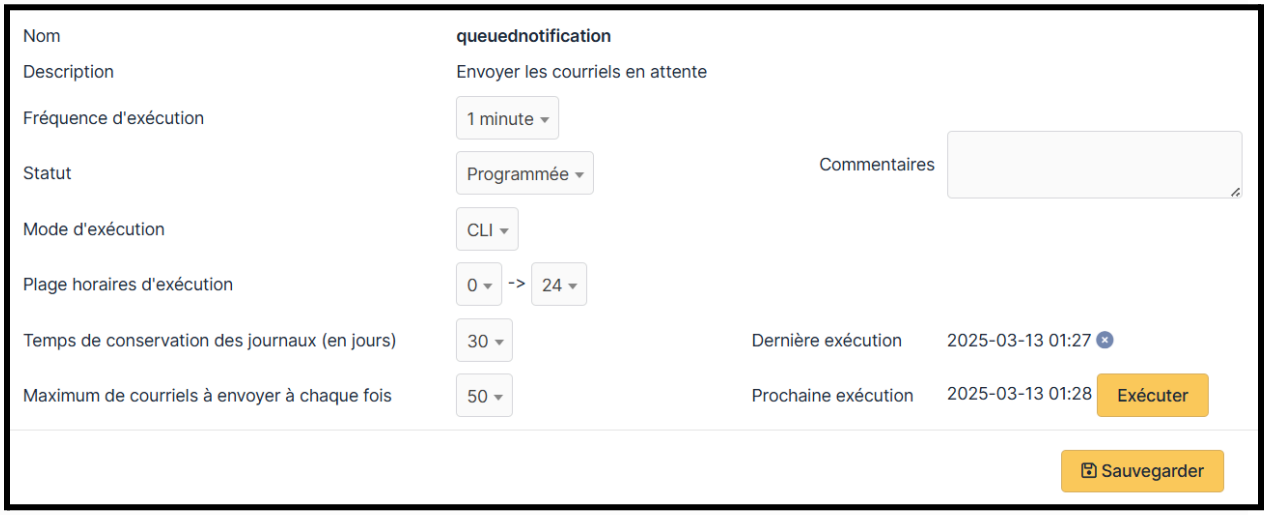
L'envoie du courrier test est réussi :



Attention il faut vérifier la fréquence d'envoi d'alerte dans le menu Configuration → Action automatique → Rechercher queuednotification :



Fréquence d'exécution sur 1 minute, le mode en CLI → Sauvegarder :



Relier OCS Inventory et GLPI

Nous voulons connecter **OCS Inventory (172.20.1.7)** à **GLPI (172.20.1.8)** pour que GLPI récupère automatiquement les inventaires de machines gérés par OCS. Cela permet d'avoir **une gestion centralisée du parc informatique** et d'éviter d'avoir deux outils séparés. De cette manière, il est possible de mettre à jour en temps réel les configurations matérielles, ainsi que d'améliorer le suivi et des interventions IT.

Par défaut, GLPI ne supporte pas OCS, donc il faut un plugin.
Téléchargeons et installons le plugin OCS dans GLPI depuis GitHub :

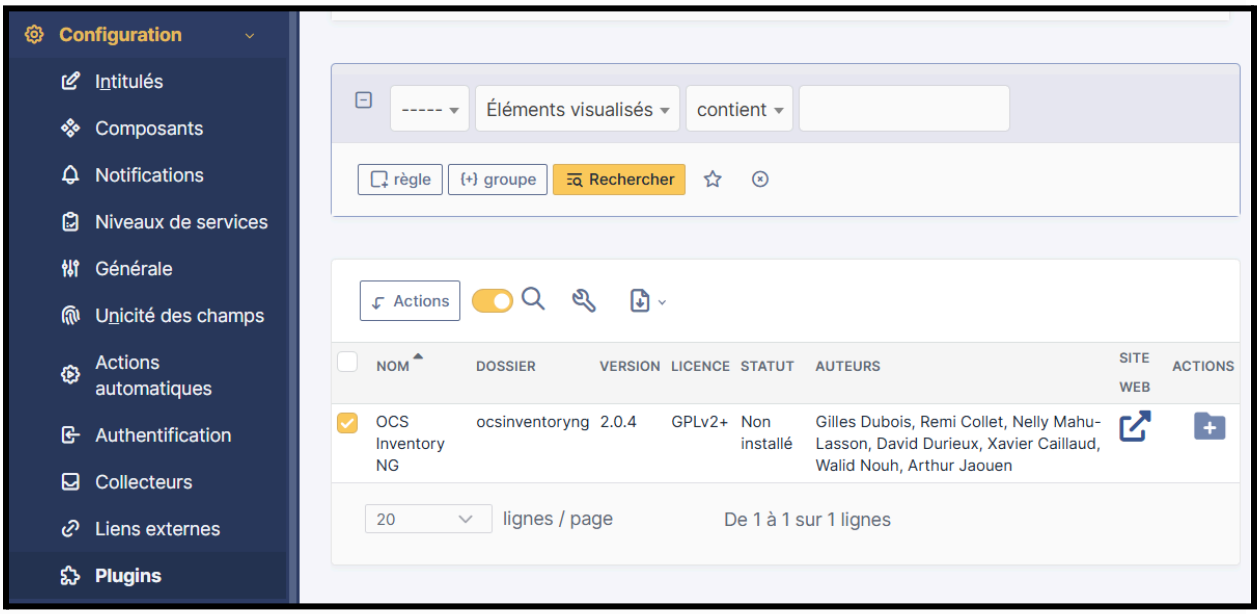
#cd /var/www/glpi/plugins/ → Se déplacer dans le dossier plugins
#wget https://github.com/pluginsGLPI/ocsinventoryng/releases/download/2.11.1/ocsinventoryng-2.11.1.#tar.gz →Télécharge la dernière version du plugin OCS Inventory NG

```
root@glpi:~# cd /var/www/glpi/plugins/
root@glpi:/var/www/glpi/plugins# wget https://github.com/pluginsGLPI/oc
sinventoryng/releases/download/2.0.4/glpi-ocsinventoryng-2.0.4.tar.bz2
```

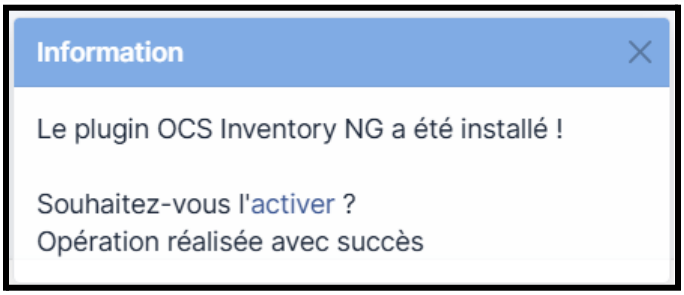
#tar -xjf glpi-ocsinventoryng-2.0.4.tar.bz2 → Décompresse l'archive téléchargée
#rm glpi-ocsinventoryng-2.0.4.tar.bz2 → Supprime l'archive pour économiser de l'espace :
#chown -R www-data:www-data ocsinventoryng/ → Assure-toi que les permissions sont correctement définies
#chmod -R 755 ocsinventoryng/ → Assure-toi que les permissions sont correctement définies

De retour sur l'interface GLPI, nous allons activer le plugin :

Configuration > Plugins. → Localiser le plugin "OCS Inventory NG" dans la liste → Installer → Activer :

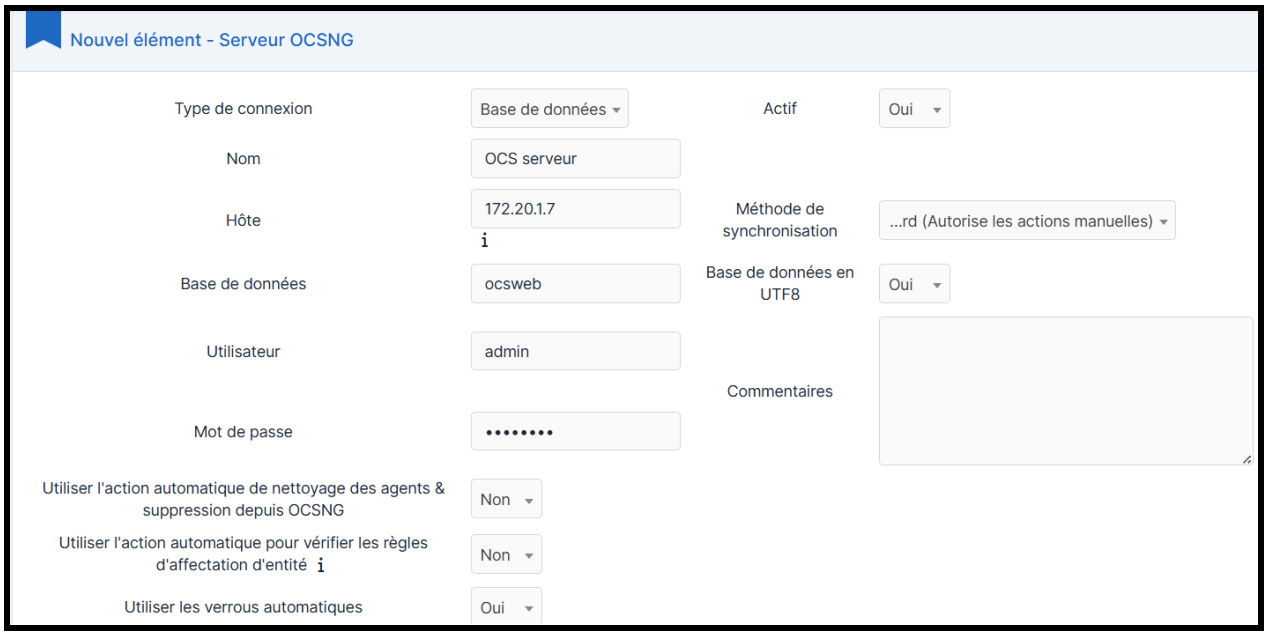


Activer le plugin :



Se rendre dans Outils --> OCS Inventory NG, et cliquer sur Ajouter un serveur OCSNG :

Remplir les données ainsi :



Ensuite, testons la connexion

Si la connexion échoue, cela peut être relié à plusieurs raisons :

1) MySQL/MariaDB n'est pas actif sur OCS
2) MySQL n'écoute pas sur l'IP d'OCS (mais seulement en local)
3) Les permissions de l'utilisateur admin ne sont pas bonnes

1) Activer MySQL/MariaDB sur OCS

Avant de connecter OCS à GLPI, il faut s'assurer que la BDD est bien active sur OCS
Depuis la console **OCS (172.20.1.7)** on vérifie le service :

#systemctl status mariadb → Vérifie si MariaDB tourne
#systemctl start mariadb → Démarre MariaDB si ce n'est pas le cas
#systemctl enable mariadb → Active MariaDB au démarrage

2) Vérifier si MySQL/MariaDB écoute bien sur l'IP de OCS (grâce au port 3306 ouvert)

ss -tlnp | grep 3306 → Vérifie si MySQL/MariaDB écoute sur le réseau

```
┌──(root💀ocs)-[~]
└─# ss -tlnp | grep 3306
LISTEN 0      80         127.0.0.1:3306      0.0.0.0:*    users:(("mariadbd",pid=594,fd=22))
```

La seule adresse IP qui apparait est 127.0.0.1:3306, ce qui indique que MySQL/MariaDB écoute seulement sur cette IP (qui est celle de l'hôte lui-même). La liaison ne se fait pas avec GLPI car MySQL/MariaDB est uniquement accessible en local.
Il faut modifier le fichier 50-server.cnf, pour indiquer d'écouter sur toutes les interfaces :

#nano /etc/mysql/mariadb.conf.d/50-server.cnf → Modifier la configuration pour indiquer d'écouter sur toutes les interfaces.

Trouver la ligne → **bind-address = 127.0.0.1** → et la modifier en → **bind-address = 0.0.0.0**

```
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 0.0.0.0
```

systemctl restart mariadb → Redémarrer
ss -tlnp | grep 3306 → On vérifie le changement effectué

```
┌──(root💀ocs)-[~]
└─# ss -tlnp | grep 3306
LISTEN 0      80           0.0.0.0:3306      0.0.0.0:*    users:(("mariadbd",pid=1727,fd=22))
```

3) Vérifier les permissions MySQL de l'utilisateur admin sur OCS

Sur la console GLPI je tente d'accéder à la BDD d'OCS :

mysql -u admin -p -h 172.20.1.7 -D dbocs -e "SHOW TABLES;" vérifier si GLPI peut bien accéder à la base de données d'OCS :

```
root@glpi:~# mysql -u admin -p -h 172.20.1.7 -D dbocs -e "SHOW TABLES;"
Enter password:
ERROR 1044 (42000): Access denied for user 'admin'@'%' to database 'dbocs'
```

J'ai un refus. Le user **admin** doit manquer de permissions…

On doit s'assurer que l'utilisateur admin a les permissions nécessaires pour accéder à la base **dbocs** depuis GLPI.

Dans MySQL sur la console OCS, vérifions l'existence de la table avec **SHOW DATABASES;**

```
┌──(root💀ocs)-[~]
└─# mysql -u root -p -e "SHOW DATABASES;"
Enter password:
+--------------------+
| Database           |
+--------------------+
| dbocs              |
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
```

#mysql -u root -p -e "SELECT user, host FROM mysql.user;" → Liste les utilisateurs MySQL/MariaDB sur OCS

```
┌──(root💀ocs)-[~]
└─# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 30
Server version: 10.5.18-MariaDB-0+deb11u1 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SELECT user, host FROM mysql.user;
+-------------+-----------+
| User        | Host      |
+-------------+-----------+
| mariadb.sys | localhost |
| mysql       | localhost |
| root        | localhost |
| userocs     | localhost |
+-------------+-----------+
4 rows in set (0,001 sec)
```

L'utilisateur admin n'apparaît pas. Il n'est pas configuré pour être utilisé à distance.

mysql -u root -p -e "SHOW GRANTS FOR 'admin'@'%';" → Vérifie les permissions de admin

```
MariaDB [(none)]> SHOW GRANTS FOR 'admin'@'%';
+-----------------------------------------------------------------------------------------------------------------+
| Grants for admin@%                                                                                              |
+-----------------------------------------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `admin`@`%` IDENTIFIED BY PASSWORD '*63DFF4CDDBC08C9376E7A8FD9DD12A6351D6C3D4' |
| GRANT ALL PRIVILEGES ON `ocsweb`.* TO `admin`@`%`                                                               |
+-----------------------------------------------------------------------------------------------------------------+
2 rows in set (0,000 sec)
```

On comprend que le user admin a des permissions sur ocsweb, mais pas sur dbocs !
Changeons cela avec ces commandes :

GRANT ALL PRIVILEGES ON dbocs.* TO 'admin'@'%' IDENTIFIED BY 'Bts2024@';
FLUSH PRIVILEGES;

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON dbocs.* TO 'admin'@'%' IDENTIFIED BY 'Bts2024@';
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,001 sec)
```

mysql -u root -p -e "SHOW GRANTS FOR 'admin'@'%';" → Montre les permissions

```
+-----------------------------------------------------------------------------------------------------------+
| Grants for admin@%                                                                                        |
+-----------------------------------------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `admin`@`%` IDENTIFIED BY PASSWORD '*63DFF4CDDBC08C9376E7A8FD9DD12A6351D6C3D4'      |
| GRANT ALL PRIVILEGES ON `ocsweb`.* TO `admin`@`%`                                                         |
| GRANT ALL PRIVILEGES ON `dbocs`.* TO `admin`@`%`                                                          |
+-----------------------------------------------------------------------------------------------------------+
```

Désormais, **admin** apparaît également sur la BDD, indiquant qu'il a été ajouté à la liste des utilisateurs avec accès distant (@'%').

```
MariaDB [(none)]> SELECT user, host FROM mysql.user;
+-------------+-----------+
| User        | Host      |
+-------------+-----------+
| admin       | %         |
| mariadb.sys | localhost |
| mysql       | localhost |
| root        | localhost |
| userocs     | localhost |
+-------------+-----------+
5 rows in set (0,001 sec)
```

Vérifions à nouveau la connexion depuis GLPI vers OCS :
mysql -u admin -p -h 172.20.1.7 -D dbocs -e "SHOW TABLES;"

```
root@glpi:~# mysql -u admin -p -h 172.20.1.7 -D dbocs -e "SHOW TABLES;"
Enter password:
+---------------------------+
| Tables_in_dbocs           |
+---------------------------+
| accesslog                 |
| accountinfo               |
| accountinfo_config        |
| archive                   |
```

admin a bien accès à la BDD d'OCS depuis GLPI.

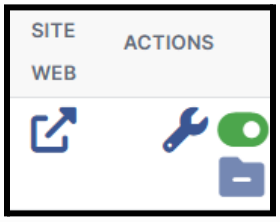
De retour sur l'interface web GLPI et après un nouveau test échoué, ce message apparait :

Signifie que **l'option TRACE_DELETED n'est pas activée dans la configuration d'OCS Inventory, donc allons l'activer :**

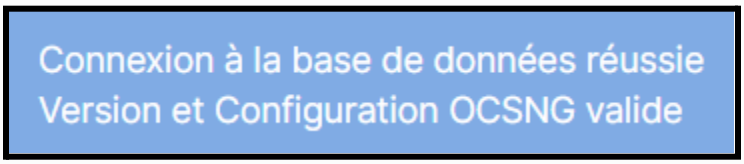
Sur l'interface web d'OCS (172.20.1.7) → Configuration → Configuration générale → Serveur → Activer la fonctionnalité → Cliquer sur "Mettre à jour" :

Retourner sur la console OCS et redémarrer tous les services :
#systemctl restart apache2
#systemctl restart mariadb
#systemctl restart ocsinventory

Bien penser à désactiver/réactiver le service également dans configuration → Plugins :



La connexion est valide :



Importer les machines sur OCS et synchronisation avec GLPI

Actuellement, **OCS est vide** car aucun agent n'a encore remonté d'inventaire.
Pour que les machines remontent automatiquement, elles doivent **avoir l'agent OCS installé**.

https://github.com/OCSInventory-NG/WindowsAgent/releases → L'agent OCS pour Windows/Linux/Mac

Effectuer une installation standard en précisant le serveur http://172.20.1.7

Vérifions la synchronisation automatique :
Sur l'interface web → Configuration > Actions automatique → Trouver ocsng_fullsync et ocsng_cleanup → Cliquer sur chaque action et les mettre en "Exécution CLI" → Cliquer sur "Exécuter" pour lancer immédiatement la synchronisation

Vérifions que les remontées d'OCS apparaissent dans GLPI :
Outils > OCS Inventory NG → Cliquer sur "Synchroniser" → Vérifier si les machines remontent bien dans GLPI (si l'agent est bien installé et configuré sur les machines).

En option, configurons la synchronisation automatique :
Afin que les nouvelles machines soient automatiquement ajoutées, aller dans :
Outils > OCS Inventory NG > Configuration → Activer "Importer automatiquement les nouveaux ordinateurs" → Activer "Mettre à jour les ordinateurs existants" → Sauvegarder