Serveur OCS

Noé MIDOUX

Pierre VARDON

Sommaire :

1) **Création et utilisation d'une machine Debian 13 comme serveur OCS partie 1:**

Après avoir installé et mit à jour la machine Debian 13, nous pouvons commencer à installer et à mettre en place le serveur OCS. Sans oublier d'installer le paquet ntpsec bien évidemment.

Nous allons installer plocate, un outil de recherche de fichiers rapide, et mc (Midnight Commander), un gestionnaire de fichiers en mode texte.

Ensuite nous allons installer make qui est un outil pour automatiser les processus de compilation.

Puis nous allons installer les paquets qui sont nécessaires à la compilation de logiciels.

apt install -y plocate mc && sudo updatedb

apt install -y make

apt install -y build-essential

```
root@serveurocs:~# apt install -y make
Installation de :
  make

Paquets suggérés :
  make-doc

Sommaire :
  Mise à niveau de : 0. Installation de : 1Supprimé : 0. Non mis à jour : 0
Taille du téléchargement : 463 kB
  Espace nécessaire : 1 808 kB / 17,5 GB disponible

Réception de : 1 http://deb.debian.org/debian trixie/main amd64 make amd64 4.4.1-2 [463 kB]
463 ko réceptionnés en 0s (10,0 Mo/s)
Sélection du paquet make précédemment désélectionné.
(Lecture de la base de données... 36325 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../make_4.4.1-2_amd64.deb ...
Dépaquetage de make (4.4.1-2) ...
Paramétrage de make (4.4.1-2) ...
Traitement des actions différées (« triggers ») pour man-db (2.13.1-1) ...
```

Tout d'abord nous installons les paquets et les dépendances d'Apache2 avec les commandes suivantes.

apt install -y apache2

apt install -y apache2-doc

apt install -y apache2-dev

Après nous installons les paquets et les dépendances de MariaDB avec les commandes suivantes et en suivant leur site officiel.

https://mariadb.org/download/?t=repo-config&d=Debian+13+%22Trixie%22&v=12.1&r_m=icam

apt-get install apt-transport-https curl

Par ailleurs on allons créer un repertoire pour sauvegarder les clés.

mkdir -p /etc/apt/keyrings

curl -o /etc/apt/keyrings/mariadb-keyring.pgp https://mariadb.org/mariadb_release_signing_key.pgp'

```
root@serveurocs:~# apt-get install apt-transport-https curl
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  apt-transport-https curl
0 mis à jour, 2 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 308 kB dans les archives.
Après cette opération, 555 ko d'espace disque supplémentaires seront utilisés.
Réception de : 1 http://deb.debian.org/debian trixie/main amd64 apt-transport-https all 3.0.3 [38,6 kB]
Réception de : 2 http://deb.debian.org/debian trixie/main amd64 curl amd64 8.14.1-2 [269 kB]
308 ko réceptionnés en 0s (4 351 ko/s)
Sélection du paquet apt-transport-https précédemment désélectionné.
(Lecture de la base de données... 51098 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../apt-transport-https_3.0.3_all.deb ...
Dépaquetage de apt-transport-https (3.0.3) ...
Sélection du paquet curl précédemment désélectionné.
Préparation du dépaquetage de .../curl_8.14.1-2_amd64.deb ...
Dépaquetage de curl (8.14.1-2) ...
Paramétrage de apt-transport-https (3.0.3) ...
Paramétrage de curl (8.14.1-2) ...
Traitement des actions différées (« triggers ») pour man-db (2.13.1-1) ...
root@serveurocs:~# sudo mkdir -p /etc/apt/keyrings
-bash: sudo : commande introuvable
root@serveurocs:~# mkdir -p /etc/apt/keyrings
root@serveurocs:~# curl -o /etc/apt/keyrings/mariadb-keyring.pgp 'https://mariadb.org/mariadb_release_signing_key.pgp'
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  6575  100  6575    0     0  55804      0 --:--:-- --:--:-- --:--:-- 56196
```

Ensuite nous ajoutons ce répertoire dans le fichier **/etc/apt/sources.list.d/mariadb.sources**.

```
  GNU nano 8.4                                          /etc/apt/sources.list.d/mariadb.sources *
# MariaDB 12.1 repository list - created 2025-11-04 08:10 UTC
# https://mariadb.org/download/
X-Repolib-Name: MariaDB
Types: deb
# deb.mariadb.org is a dynamic mirror if your preferred mirror goes offline. See https://mariadb.org/mirrorbits/ for details.
# URIs: https://deb.mariadb.org/11.rc/debian
URIs: https://mirrors.ircam.fr/pub/mariadb/repo/12.1/debian
Suites: trixie
Components: main
Signed-By: /etc/apt/keyrings/mariadb-keyring.pgp
```

Pour continuer nous mettons les paquets à jour et nous installons les paquets et les dépendances de MariaDB serveur.

apt-get update

apt-get install mariadb-server

```
root@serveurocs:/etc/apt/sources.list.d# apt-get update
Atteint : 1 http://deb.debian.org/debian trixie InRelease
Atteint : 2 http://deb.debian.org/debian trixie-updates InRelease
Atteint : 3 http://security.debian.org/debian-security trixie-security InRelease
Réception de : 4 https://mirrors.ircam.fr/pub/mariadb/repo/12.1/debian trixie InRelease [4 620 B]
Réception de : 5 https://mirrors.ircam.fr/pub/mariadb/repo/12.1/debian trixie/main amd64 Packages [19,0 kB]
23,7 ko réceptionnés en 0s (163 ko/s)
Lecture des listes de paquets... Fait
root@serveurocs:/etc/apt/sources.list.d# apt-get install mariadb-server
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  galera-4 gawk libaio1t64 libcgi-fast-perl libcgi-pm-perl libclone-perl libconfig-inifiles-perl libdbd-mariadb-perl libdbi-perl libencode-locale-perl
  libfcgi-bin libfcgi-perl libfcgi0t64 libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl
  libio-compress-brotli-perl libio-html-perl liblwp-mediatypes-perl libmariadb3 libncurses6 libsigsegv2 libterm-readkey-perl libtimedate-perl liburi-perl
  mariadb-client mariadb-client-compat mariadb-client-core mariadb-common mariadb-server-compat mariadb-server-core mysql-common psmisc pv rsync socat
Paquets suggérés :
  gawk-doc libmldbm-perl libnet-daemon-perl libsql-statement-perl libdata-dump-perl libipc-sharedcache-perl libbusiness-isbn-perl libmime-base32-perl
  libregexp-ipv6-perl libwww-perl mailx mariadb-test netcat-openbsd doc-base python3-braceexpand
Les NOUVEAUX paquets suivants seront installés :
  galera-4 gawk libaio1t64 libcgi-fast-perl libcgi-pm-perl libclone-perl libconfig-inifiles-perl libdbd-mariadb-perl libdbi-perl libencode-locale-perl
  libfcgi-bin libfcgi-perl libfcgi0t64 libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl libhttp-date-perl libhttp-message-perl
  libio-compress-brotli-perl libio-html-perl liblwp-mediatypes-perl libmariadb3 libncurses6 libsigsegv2 libterm-readkey-perl libtimedate-perl liburi-perl
  mariadb-client mariadb-client-compat mariadb-client-core mariadb-common mariadb-server mariadb-server-compat mariadb-server-core mysql-common psmisc pv
  rsync socat
0 mis à jour, 39 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 32,3 MB dans les archives.
Après cette opération, 250 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n]
```

Une fois l'installation des paquets et les dépendances de MariaDB serveur faite nous allons faire la même chose pour ceux de PHP.

apt install -y php

apt install -y php-pear

apt install -y php-mbstring

apt install -y php-soap

apt install -y php-mysql

apt install -y php-curl

apt install -y php-zip

apt install -y php-gd

Ensuite pour que tout cela fonctionne correctement il est nécessaire d'installer les librairies Perl qui sont nécessaires au bon fonctionnement de OCS.

apt install -y libgd-tools

apt install -y libnet-ip-perl

apt install -y libxml-simple-perl

apt install -y libarchive-zip-perl

apt install -y libapache-dbi-perl

apt install -y libsoap-lite-perl

apt install -y libapache2-mod-perl2

apt install -y libdigest-hmac-perl

apt install -y libgssapi-perl

apt install -y libgd-dev

apt install -y libcrypt-ssleay-perl

apt install -y uuid

apt install -y libmime-lite-perl

sudo apt install -y libnet-jabber-perl

apt install -y libauthen-ntlm-perl

apt install -y libxml-sax-expatxs-perl

apt install -y libmojolicious-perl

apt install -y libdbd-mysql-perl

Après avoir tout installer nous faisons la commande cpan et nous choisissons yes pour configurer nos modules installer précédemment.

```
root@serveurocs:/# cpan
Loading internal logger. Log::Log4perl recommended for better logging

CPAN.pm requires configuration, but most of it can be done automatically.
If you answer 'no' below, you will enter an interactive dialog for each
configuration option instead.

Would you like to configure as much as possible automatically? [yes] yes

We initialized your 'urllist' to https://cpan.org/. Type 'o conf init urllist' to change it.

Autoconfiguration complete.

commit: wrote '/root/.cpan/CPAN/MyConfig.pm'

You can re-run configuration any time with 'o conf init' in the CPAN shell
Terminal does not support AddHistory.

To fix that, maybe try>  install Term::ReadLine::Perl


cpan shell -- CPAN exploration and modules installation (v2.36)
Enter 'h' for help.

cpan[1]>
```

Pour faciliter la configuration de nos modules nous faisons ces commandes-là.

i /cpan/

reload cpan

exit

Ensuite nous installons des modules Perl spécifiques via le gestionnaire cpan.

cpan YAML

perl -MCPAN -e 'install XML::Entities'

perl -MCPAN -e 'install Apache2::SOAP'

perl -MCPAN -e 'install Net::IP'

perl -MCPAN -e 'install Apache::DBI'

perl - MCPAN -e 'install Mojolicious::Lite'

perl -MCPAN -e 'install Switch'

perl -MCPAN -e 'install Plack::Handler'

Pour continuer nous installons les modules Perl utiles pour le traitement de fichiers ZIP les communications sécurisées et les services Web.

cpan -f Archive::Zip

perl -MCPAN -e 'install Crypt::SSLeay'

perl -MCPAN -e 'install SOAP::Lite'

Après nous allons sécuriser l'installation de notre base de données en faisant la commande suivante.

mysql_secure_installation

Une fois la commande exécuter nous faisons entrer puis n, n, Y, n, Y, Y.

Ensuite nous redémarrons la machine pour appliquer les changements effectuer.

reboot

Une fois la machine redémarrer nous accédons à notre base de données.

mysql -u root –p

Nous créeons la base de données ocsweb avec un utilisateur ocs puis nous lui mettons les privilèges sur celle-ci et nous appliquons ces changements.

create database ocsweb;

CREATE USER 'ocs'@'localhost' IDENTIFIED BY 'root';

GRANT ALL PRIVILEGES ON ocsweb.* TO 'ocs'@'localhost';

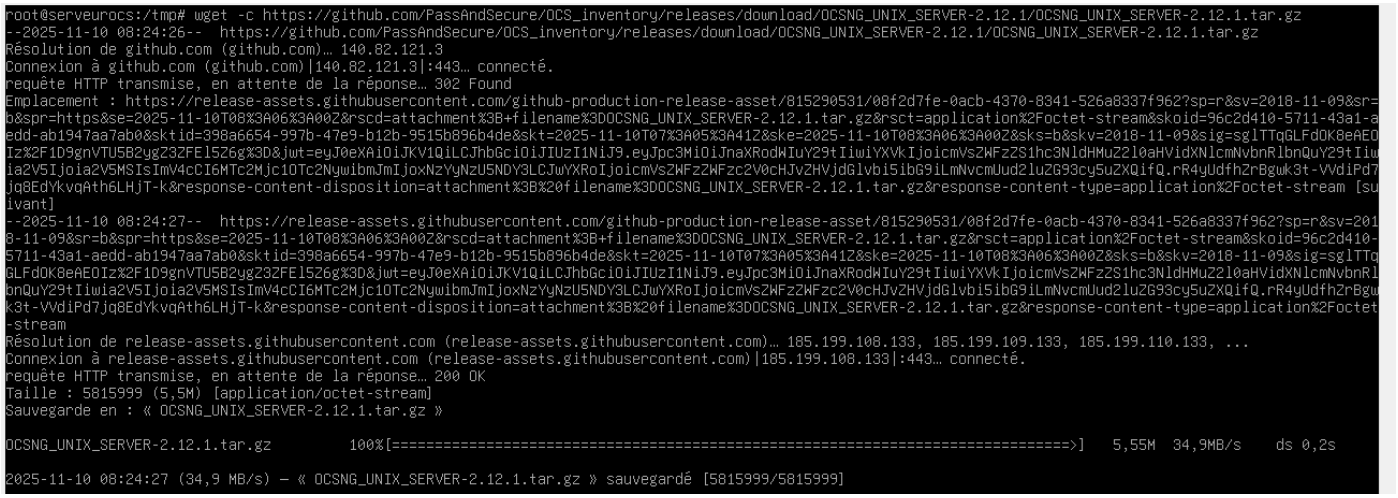
flush privileges;

exit;

```
root@serveurocs:~# mysql -u root -p
mysql: Deprecated program name. It will be removed in a future release, use '/usr/bin/mariadb' instead
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 12.1.1-MariaDB-deb13 mariadb.org binary distribution

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 ocsweb ;
Query OK, 1 row affected (0,002 sec)

MariaDB [(none)]> create user ocs identified by 'ocs';
Query OK, 0 rows affected (0,012 sec)

MariaDB [(none)]> grant all privileges on ocsweb.* to ocs ;
Query OK, 0 rows affected (0,002 sec)

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

MariaDB [(none)]> exit;
Bye
```

Ensuite nous installons les fichiers nécessaires à l'installation ou à la configuration de OCS Inventory depuis leur dépôt GitHub officiel.

wget -c
https://github.com/PassAndSecure/OCS_inventory/releases/download/OCSNG_UNIX_SERVER-2.12.1/OCSNG_UNIX_SERVER-2.12.1.tar.gz



Après avoir installé les fichiers de OCS Inventory nous les décompressons et nous allons dans le répertoire où se trouve les fichiers décompresser.

tar xfvz OCSNG_UNIX_SERVER-2.12.1.tar.gz && cd OCSNG_UNIX_SERVER-2.12.1

Pour continuer nous exécutons le script d'installation **setup.sh** présent dans le répertoire actuel puis nous enregistrons la sortie dans le fichier **/tmp/installOCS.txt** tout en l'affichant dans le terminal.

sh setup.sh | tee /tmp/installOCS.txt

```
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ProxiedService/Imap.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ProxiedService/Http/
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ProxiedService/Http/Abstract.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ProxiedService/Http/Get.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ProxiedService/Http/Post.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/TypeMismatchException.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/PGTStorage/
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/PGTStorage/AbstractStorage.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/PGTStorage/File.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/PGTStorage/Db.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/Client.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ServiceBaseUrl/
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ServiceBaseUrl/Base.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ServiceBaseUrl/AllowedListDiscovery.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ServiceBaseUrl/Interface.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/ServiceBaseUrl/Static.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS/OutOfSequenceBeforeProxyException.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/source/CAS.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/CAS.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/composer.json
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/jasig/phpcas/NOTICE
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/autoload_namespaces.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/autoload_real.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/LICENSE
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/ClassLoader.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/autoload_psr4.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/autoload_static.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/autoload_classmap.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/autoload_files.php
OCSNG_UNIX_SERVER-2.12.1/ocsreports/vendor/composer/installed.json
OCSNG_UNIX_SERVER-2.12.1/setup.sh
root@serveurocs:/tmp/OCSNG_UNIX_SERVER-2.12.1# sh setup.sh  tee /tmp/installOCS.txt
setup.sh: 15: [[: not found

+-----------------------------------------------------------+
|                                                           |
|  Welcome to OCS Inventory NG Management server setup !    |
|                                                           |
+-----------------------------------------------------------+

Trying to determine which OS or Linux distribution you use
+-----------------------------------------------------------+
| Checking for Apache web server binaries !                 |
+-----------------------------------------------------------+

CAUTION: If upgrading Communication server from OCS Inventory NG 1.0 RC2 and
previous, please remove any Apache configuration for Communication Server!

Do you wish to continue ([y]/n)?
```

Ensuite nous mettons les permissions nécessaires à OCS Inventory puis nous relançons le service apache2 et la machine pour appliquer ces changements.

**chown -R www-data:www-data /var/lib/ocsinventory-reports/ && **

**a2enconf ocsinventory-reports && **

**a2enconf ocsinventory-server && **

**a2enconf ocsinventory-restapi && **

systemctl reload apache2

reboot

Après nous allons ajouter les informations de notre base de données dans les fichiers de configuration de OCS Inventory qui sont **/etc/apache2/conf-available/z-ocsinventory-server.conf** et **/usr/share/ocsinventory-reports/ocsreports/dbconfig.inc.php**.

Tout d’abord nous faisons les modifications dans le premier fichier de configuration.

nano /etc/apache2/conf-available/z-ocsinventory-server.conf

PerlSetEnv OCS_DB_NAME ocsweb

PerlSetEnv OCS_DB_LOCAL localhost

PerlSetEnv OCS_DB_USER ocs

PerlSetVar OCS_DB_PWD sio

Puis nous faisons les modifications dans le second fichier de configuration.

nano /usr/share/ocsinventory-reports/ocsreports/dbconfig.inc.php

```
  GNU nano 8.4                                  /usr/share/ocsinventory-reports/ocsreports/dbconfig.inc.php *
<?php
//$_SESSION["SERVEUR_SQL"]="localhost";
//$_SESSION["COMPTE_BASE"]="ocs";
//$_SESSION["PSWD_BASE"]="ocs";



define("DB_NAME", "ocsweb");
define("SERVER_READ", "localhost");
define("SERVER_WRITE", "localhost");
define("SERVER_PORT", "3306");
define("COMPT_BASE", "ocs");
define("PSWD_BASE", "sio");
define("ENABLE_SSL", "");
define("SSL_MODE", "");
define("SSL_KEY", "");
define("SSL_CERT", "");
define("CA_CERT", "");



?>
```

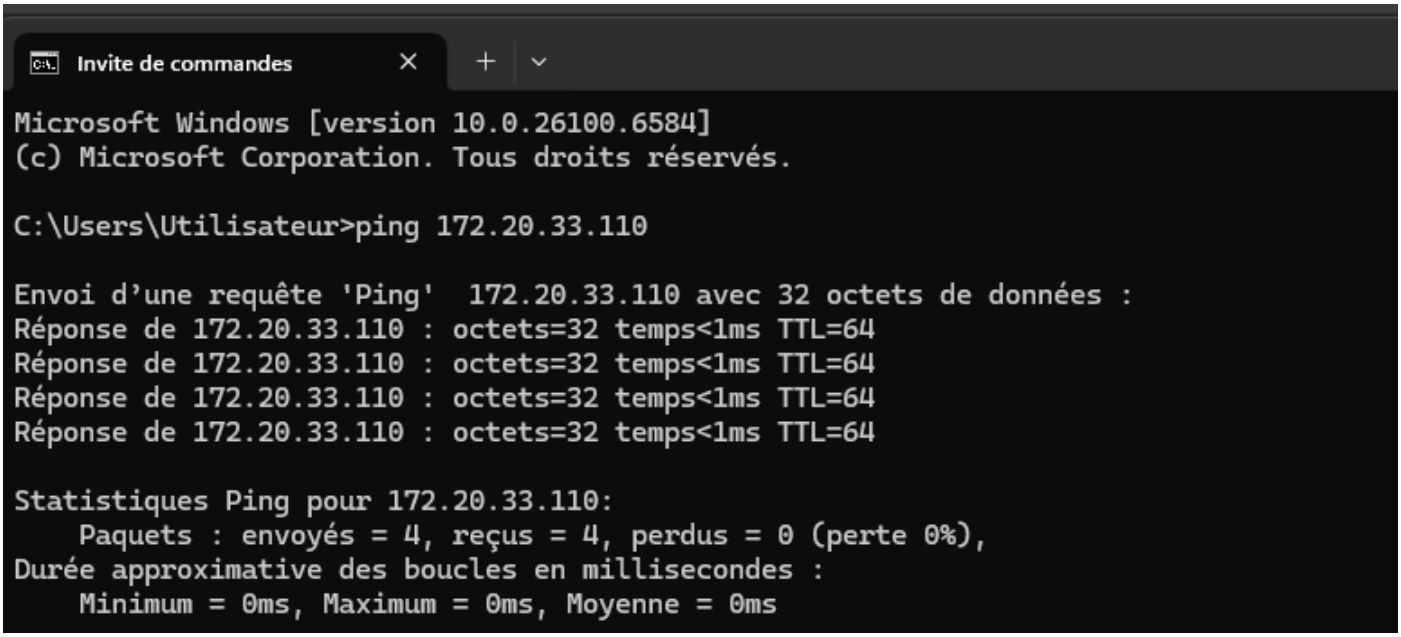
systemctl restart apache2

Ensuite nous redémarrons le service apache2 pour que les différents changements soient pris en compte.

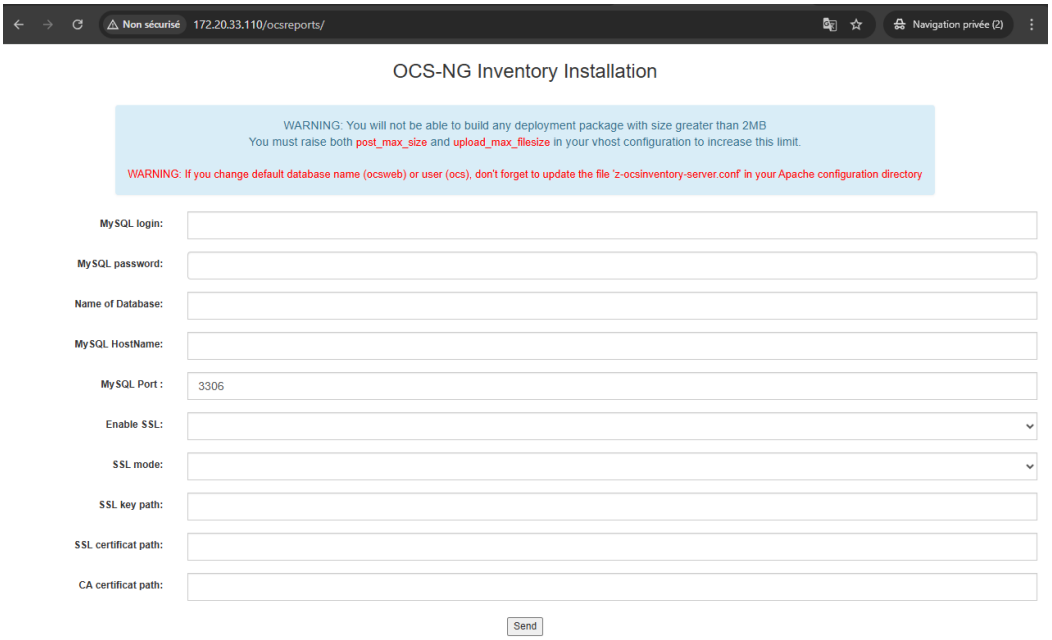
Maintenant nous allons tester la connexion entre le serveur OCS et le client Windows.

```
root@serveurocs:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b9:be:dd brd ff:ff:ff:ff:ff:ff
    altname enx080027b9bedd
    inet 172.20.33.110/16 brd 172.20.255.255 scope global enp0s3
       valid_lft forever preferred_lft forever
```

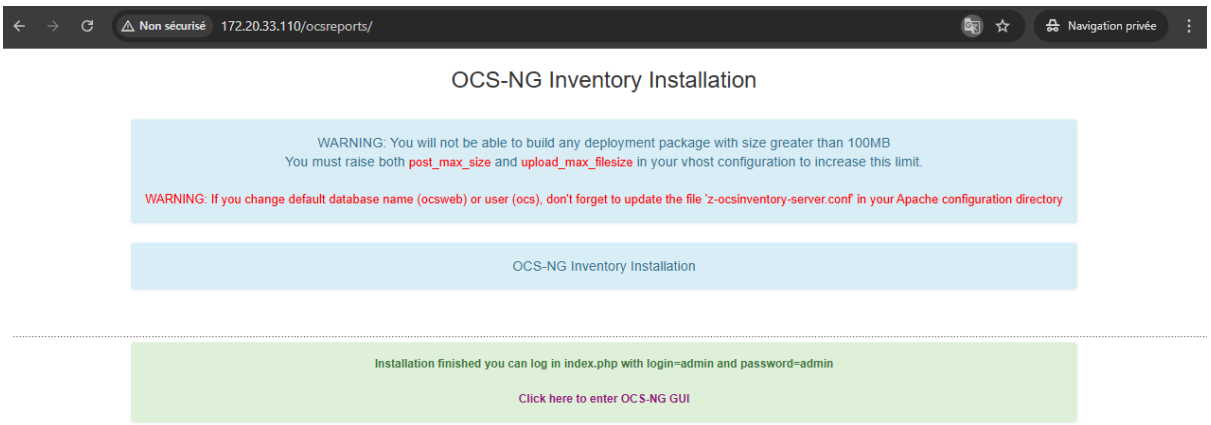
Pour ce faire nous faisons un ping du serveur OCS depuis le client Windows 11.



Une fois que nous avons vérifié que le serveur OCS et le client Windows communique bien entre eux nous ouvrons un navigateur sur le client Windows et nous tapons l'adresse ip du serveur OCS dans l'url **http://172.20.34.110/ocsreports/**.

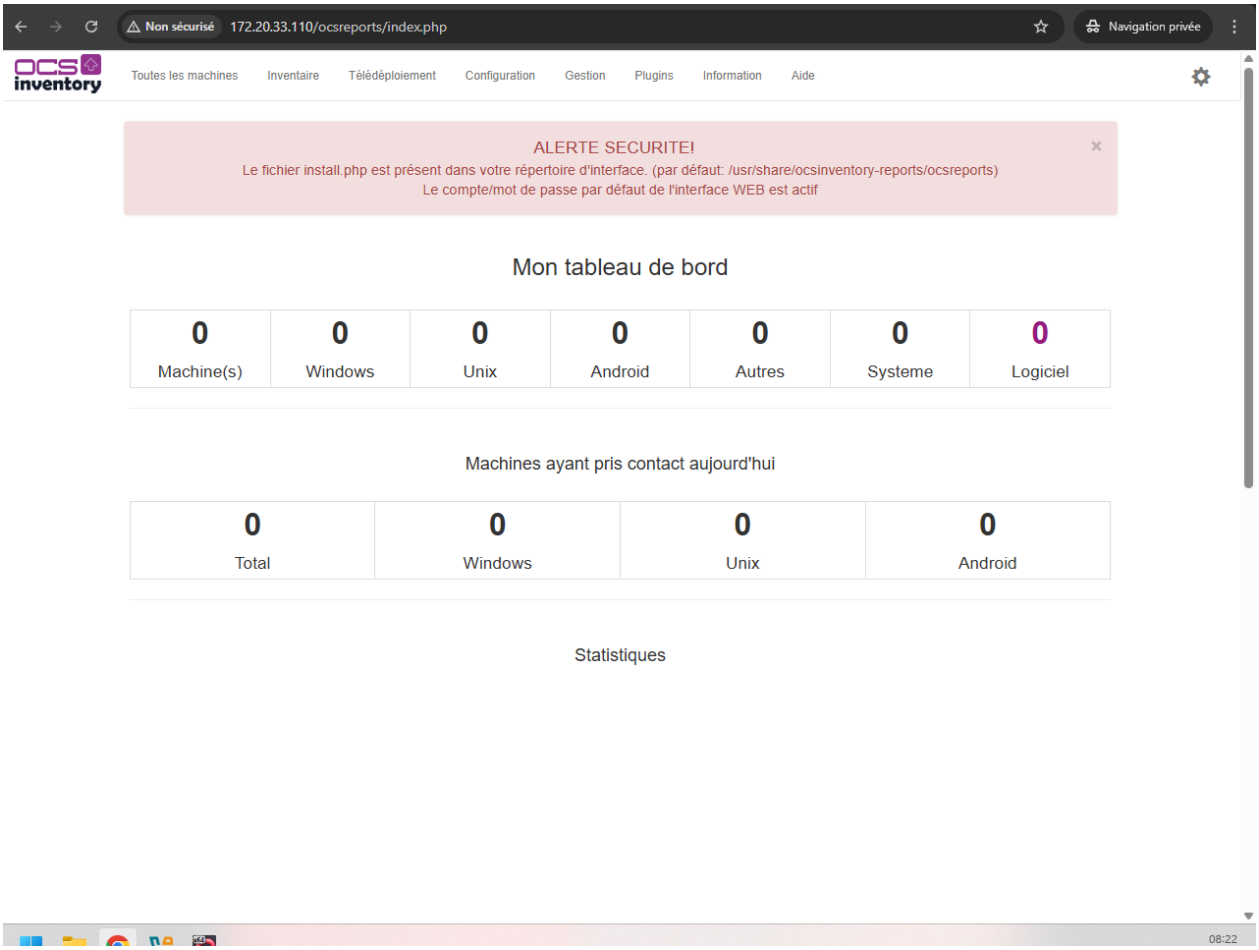


Ensuite nous renseignons nos informations de connexion qui sont les mêmes que ceux de notre base de données pour accéder au panel de OCS Inventory.



Nous voyons qu'à la première connexion il nous indique qu'il y a des failles de sécurités comme le nom de notre base de données et la taille maximale des fichiers qui sont déployable par OCS Inventory. Nous pouvons faire ces changements si nous le souhaitons en modifiant ces paramètres-là dans les fichiers de configurations de notre base de données et dans ceux de OCS Inventory.

Après le message nous affichant les failles de sécurités nous arrivons sur le panel de OCS Inventory et nous voyons que tout a fonctionné correctement.



Ensuite sur le client Windows nous installons l’agent OCS pour le lier avec notre serveur OCS.

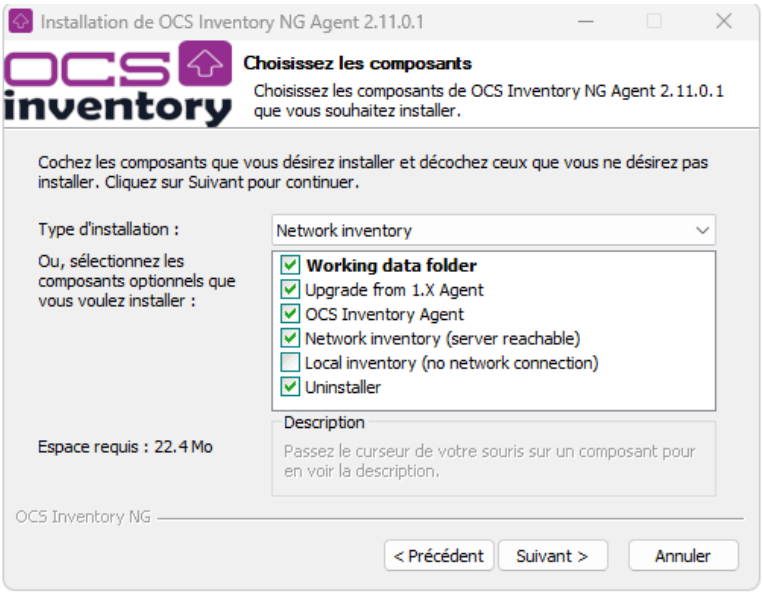
Nous allons regarder la documentation officielle d’installation de l’agent OCS Inventory.

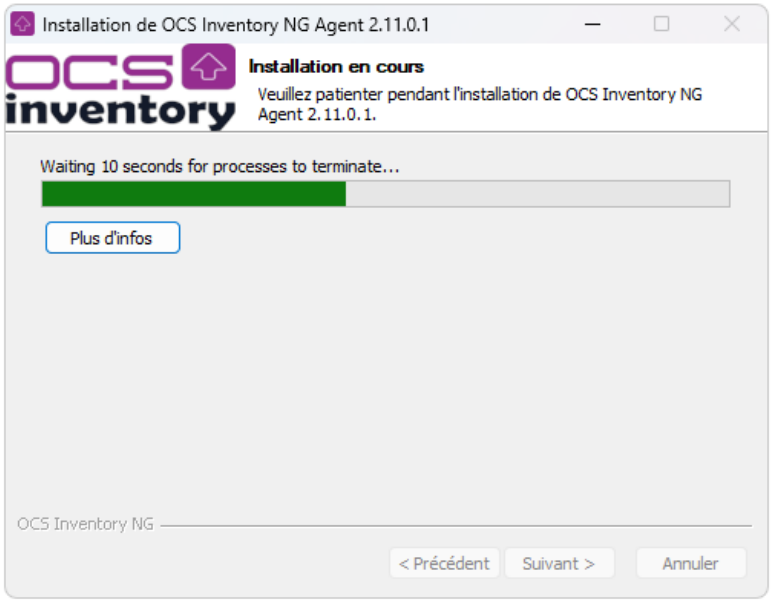
https://wiki.ocsinventory-ng.org/03.Basic-documentation/Setting-up-the-Windows-Agent-2.xon-client-computers/

Nous récupérons l'agent OCS pour le client Windows sur leur GitHub officiel.

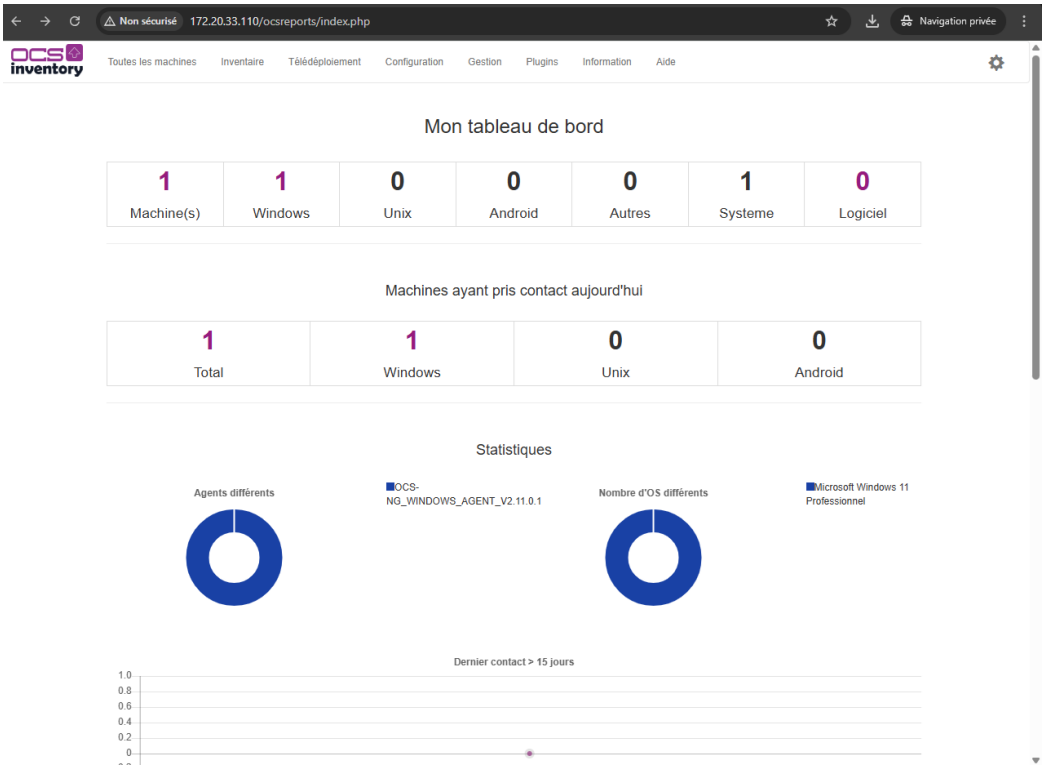
https://github.com/OCSInventory-NG/WindowsAgent/releases/download/2.11.0.1/OCS-WindowsAgent-2.11.0.1_x64.zip

Une fois le fichier télécharger nous le décompressons puis nous l'exécutons et nous cliquons sur suivant.

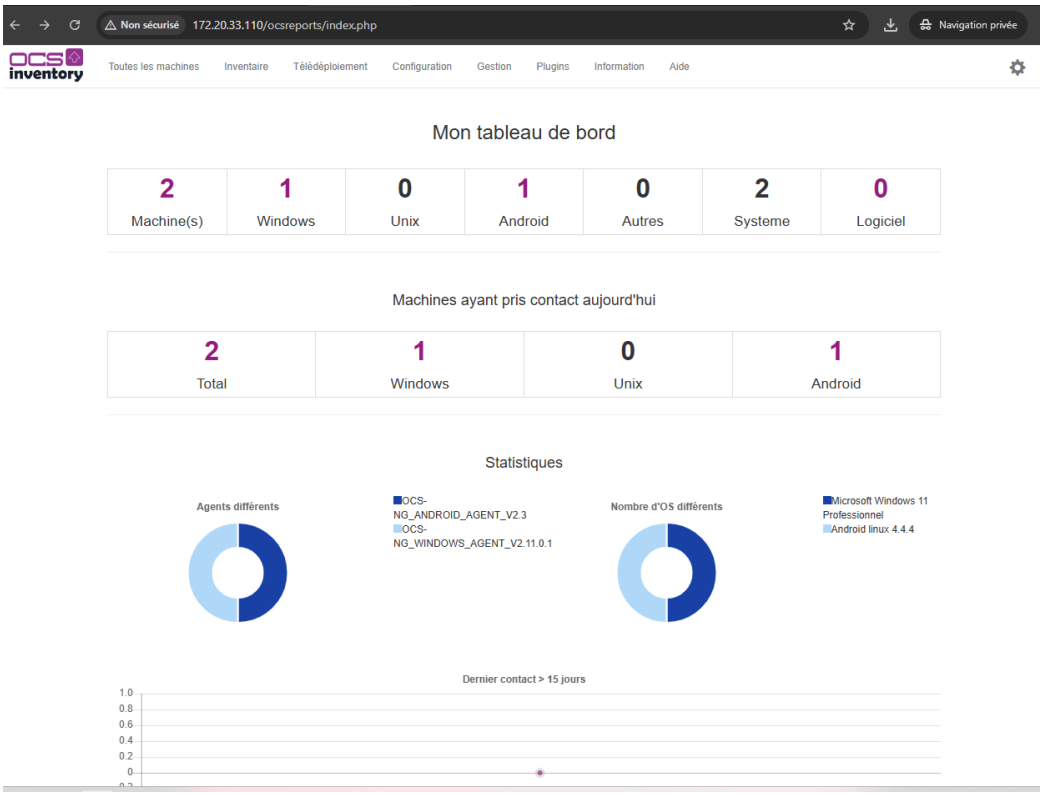




Après avoir finaliser l'installation de l'agent OCS sur le client Windows nous l'exécutons et nous retournons sur notre panel OCS. Nous voyons que l'agent a bien remonté notre client Windows ce qui signifie que cela fonctionne correctement.



Pour continuer nous ajoutons un client Android pour vérifier que tout est bien fonctionnel.



a) Pensez-vous que cette procédure soit digne d'une procédure d'entreprise ? si non, justifiez et argumentez ce qui manque, ce qui doit être ajouté/modifié…

Non, cette procédure n'est pas optimale pour une procédure d'entreprise car tout d'abord, cette procédure possède des défauts de sécurité car pas de gestion des permissions préciser, affichage du mot de passe 'ocs'

de l'utilisateur affiché en dur dans la procédure, manque de durcissement du système comme un pare feu et pas de chiffrement HTTPS !

Ensuite, elle manque de maintenabilité car il n'y a aucun script d'automatisation pour l'installation et requiert beaucoup d'interventions manuelles ce qui est fragile et non reproductible.

De plus, cela manque aussi de traçabilité car aucune trace des versions des paquets installés comme des logs par exemples.

Pour finir, cette procédure n'est pas digne d'une procédure d'entreprise car elle doit être reproductible par n'importe quel technicien sans aide extérieure, or elle inclut des décisions de choix de paquets etc... Qui ne sont pas documenté et expliqué.

b) Cette procédure respecte-t-elle les bonnes pratiques en matière de sécurité ? si non, dire pourquoi et ce qu'il convient d'améliorer…

Non cela ne respecte pas les bonnes pratiques de sécurité car :

On y trouve un mélange de Apt et cpan pour les installations de paquets

On installe une archive via un lien GitHub sans gérer sa version dans un dépôt interne

Même problème pour la dépendance des versions de Pearl etc.., car on installe la dernière version disponible à l'instant T mais par exemple une nouvelle version apparaît dans le cas de Pearl, alors le serveur ne peut plus démarrer

c) Adaptez cette procédure pour qu'elle respecte toutes les bonnes pratiques. Mettre en place l'installation de OCS Inventory Server avec sa documentation sur Debian 13.

Recommandations pour respecter les bonnes pratiques de sécurités :

Installer les modules Perl depuis les dépôts APT autant que possible car cela est plus sûr et la maintenance est plus simple.

On installe beaucoup de paquets, dont certains redondants ou obsolètes comme les –docs donc il faut installer uniquement les paquets nécessaires.

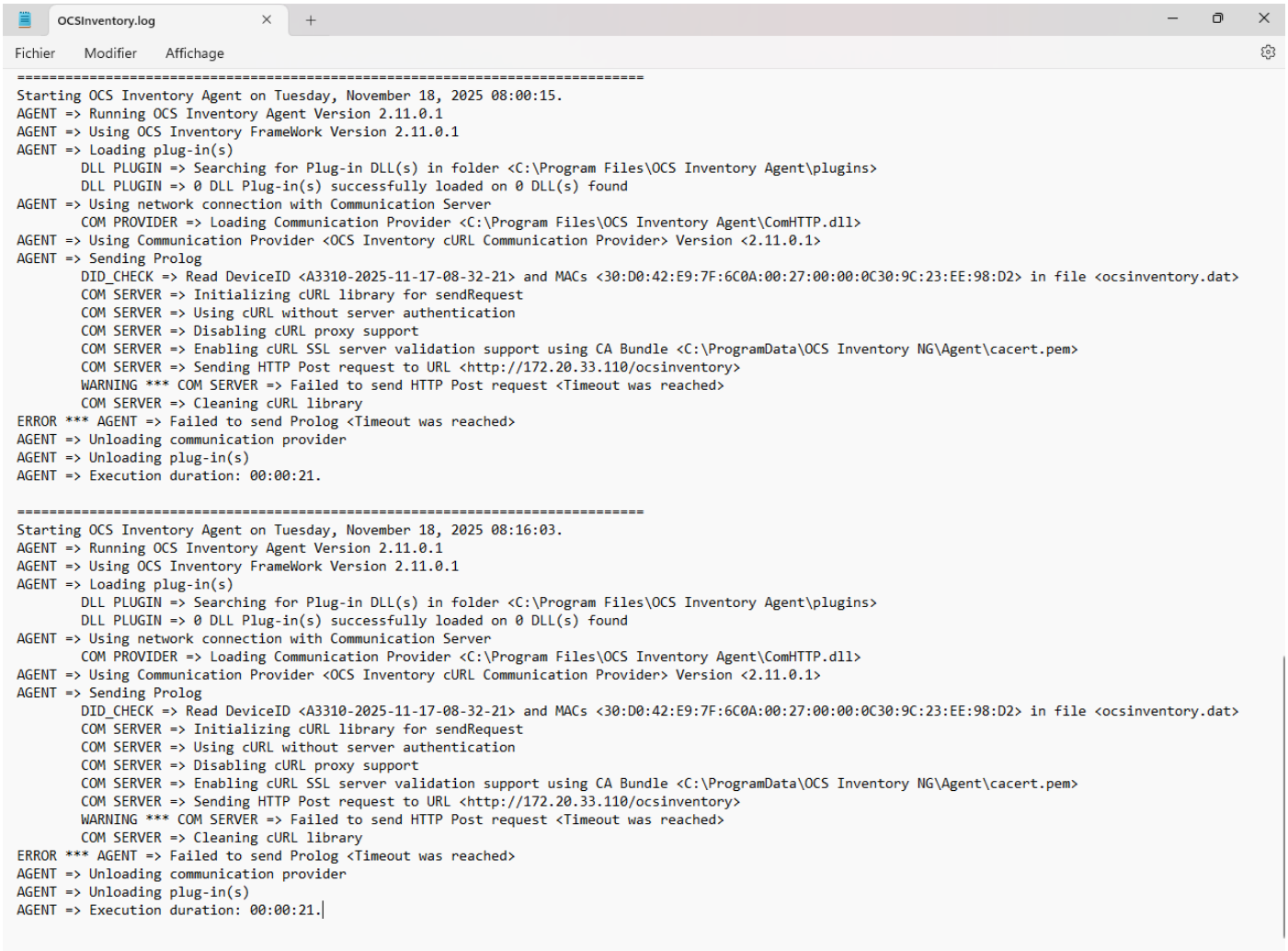
Éviter de donner la propriété à `www-data` sur des fichiers qui n'ont pas besoin d'être modifiables par Apache.

d) Donnez toutes les actions à réaliser pour corriger cette alerte sécurité.

Pour retirer cette alerte de sécurité il suffit de modifier le mot de passe de la base de données de OCS ainsi que le mot de passe de l'interface web en respectant les bonnes pratiques de mots de passe préconiser par l'ANSSI. Par ailleurs il faut supprimer le fichier install.php car un pirate pourrait réinstaller tout le OCS et ainsi avoir le contrôle total.

e) Assurez-vous que votre client Windows 10/11 contacte toujours bien le serveur. On retrouve les « logs » de OCS Agent dans : %programdata%\OCS Inventory NG\Agent\OCSInventory.log

On remarque que le client Windows 11 contact toujours bien le serveur OCS.



2) Création et utilisation d'une machine Debian 13 comme serveur OCS partie 2 :

1) Proposez et expliquez toutes les solutions possibles ; Choisissez-en une que vous mettrez en œuvre et pour laquelle vous rédigerez une procédure d'installation automatique de l'agent OCS pour un poste Windows 10 ou pour un poste Windows 11.

Version réseau :

Utilisation de PowerShell Remoting :

Prérequis :

Activer WinRM sur les postes avec la commande :

Enable-PSRemoting -Force

Autoriser l'exécution des scripts avec la commande :

Set-ExecutionPolicy RemoteSigned -Force

Droits d'administration avec la commande :

WinRM-HTTP-In

Mise en place de PowerShell Remoting :

Étape 1 :

Écrire un script PowerShell pour installer l'agent OCS Inventory NG.

Étape 2 :

Utiliser `Invoke-Command` pour exécuter le script sur les machines distantes.

Étape 3 :

Assurez-vous que PowerShell Remoting est activé et que vous avez les permissions nécessaires sur tous les postes de travail avec la commande Test-WsMan PC001.

Version dev :

- Etape 1 : Récupération du code source avec par exemple :
  git clone https://github.com/OCSInventory-NG/WindowsAgent.git
  cd WindowsAgent

- Etape 2: Compilation du code source avec CMake et MSBuild :
  mkdir build
  cd build
  cmake ..
  Msbuild OCSInventory.sln /p:Configuration=Release

- Etape 3 : Structure finale du client OCS :

  1. Diriger les fichiers générer précédemment dans **C:\OCS-Agent-Dev**
  2. Créer un fichier **ocsinventory.ini** où l'on va mettre le script suivant :

  [OCS Inventory Agent]
  server=http://MON_SERVEUR_ocs/ocsinventory
  logfile=C:\OCS-Agent-Dev\agent.log
  tag=DEVCLIENT
  debug=1
  ssl=0
  now=1

3. Créer un fichier **install-service.ps1** où l'on va mettre le script suivant :

```powershell
$serviceName = "OCSInventory"

# Arrêter et supprimer si déjà présent
if (Get-Service $serviceName -ErrorAction SilentlyContinue) {
    Stop-Service $serviceName -Force
    sc.exe delete $serviceName
}

# Installer le service
sc.exe create $serviceName binPath= "C:\OCS-Agent-Dev\OCSInventoryService.exe" start= auto

# Démarrer le service
Start-Service $serviceName

Write-Output "Service OCS version DEV installé et démarré."
```

4. Créer un fichier **run-ocs-inventory.ps1** où l'on va mettre le script suivant :

Start-Process "C:\OCS-Agent-Dev\OCSInventory.exe"
-ArgumentList "/ini=C:\OCS-Agent-Dev\ocsinventory.ini"
-Wait

Write-Output "Inventaire envoyé."

- Etape 4 : Déploiement automatique :

  Il suffit de copier ce dossier sur les différents postes, puis d'éxecuter :

```powershell
Copy-Item "\\SERVEUR\DEV\OCS-Agent-Dev" "C:\OCS-Agent-Dev" -Recurse -Force
powershell.exe -ExecutionPolicy Bypass -File C:\OCS-Agent-Dev\install-service.ps1
powershell.exe -ExecutionPolicy Bypass -File C:\OCS-Agent-Dev\run-ocs-inventory.ps1
```

Nous allons utiliser la méthode dev car elle est plus simple à mettre en place vue qu'elle ne nécessite pas de GPO donc pas d'AD contrairement à la méthode réseau.

2) Vous mettrez en œuvre trois de ces plugins au choix et réaliserez un mode opératoire. Précisez les tests fonctionnels nécessaires.

https://github.com/pluginsOCSInventory-NG

https://plugins.ocsinventory-ng.org/

Nous allons choisir ces trois plugins là :

- Récupérer les services Windows et leur état
- Récupérer les processus en cours d'exécution
- Remonte les imprimantes connectées par le réseau

- **Récupérer les services Windows et leur état** : Il permet de voir les services essentiels pour la sécurité et le bon fonctionnement du poste de travail. Il permet également de prévenir des pannes critiques.

- **Récupérer les processus en cours d'exécution** : Il permet de savoir ce qui tourne sur chaque machine, il garde une trace des processus actifs, il peut prévenir les infections par des malwares ou détecter des applications non autorisées.
- **Remonte les imprimantes connectées par le réseau** : Il permet de centraliser les informations et de suivre l'état de l'infrastructure d'impression sur les imprimantes.

Nous nous mettons dans le répertoire **/tmp** et nous allons cloner les dépôts git des plugins sur notre serveurs OCS pour les ajouter dessus.

cd /tmp

git clone https://github.com/pluginsOCSInventory-NG/services.git

git clone https://github.com/pluginsOCSInventory-NG/runningProcess.git

git clone https://github.com/pluginsOCSInventory-NG/listprinters.git

```
root@serveurocs:/tmp# git clone https://github.com/pluginsOCSInventory-NG/services.git
Clonage dans 'services'...
remote: Enumerating objects: 55, done.
remote: Total 55 (delta 0), reused 0 (delta 0), pack-reused 55 (from 1)
Réception d'objets: 100% (55/55), 96.74 Kio | 2.36 Mio/s, fait.
Résolution des deltas: 100% (12/12), fait.
root@serveurocs:/tmp# git clone https://github.com/pluginsOCSInventory-NG/runningProcess.git
Clonage dans 'runningProcess'...
remote: Enumerating objects: 98, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 98 (delta 1), reused 1 (delta 0), pack-reused 92 (from 1)
Réception d'objets: 100% (98/98), 55.28 Kio | 1.91 Mio/s, fait.
Résolution des deltas: 100% (29/29), fait.
root@serveurocs:/tmp# git clone https://github.com/pluginsOCSInventory-NG/listprinters.git
Clonage dans 'listprinters'...
remote: Enumerating objects: 60, done.
remote: Total 60 (delta 0), reused 0 (delta 0), pack-reused 60 (from 1)
Réception d'objets: 100% (60/60), 24.64 Kio | 1.45 Mio/s, fait.
Résolution des deltas: 100% (17/17), fait.
```

Ensuite nous copions les plugins côté serveur.

cp -r /tmp/services /usr/share/ocsinventory-reports/ocsreports/extensions/

cp -r /tmp/runningProcess /usr/share/ocsinventory-reports/ocsreports/extensions/

cp -r /tmp/listprinters /usr/share/ocsinventory-reports/ocsreports/extensions/

Nous vérifions qu'ils apparaissent bien.

ls /usr/share/ocsinventory-reports/ocsreports/extensions/

```
root@serveurocs:/tmp# ls /usr/share/ocsinventory-reports/ocsreports/extensions/
listprinters  Readme.md  runningProcess  services
```

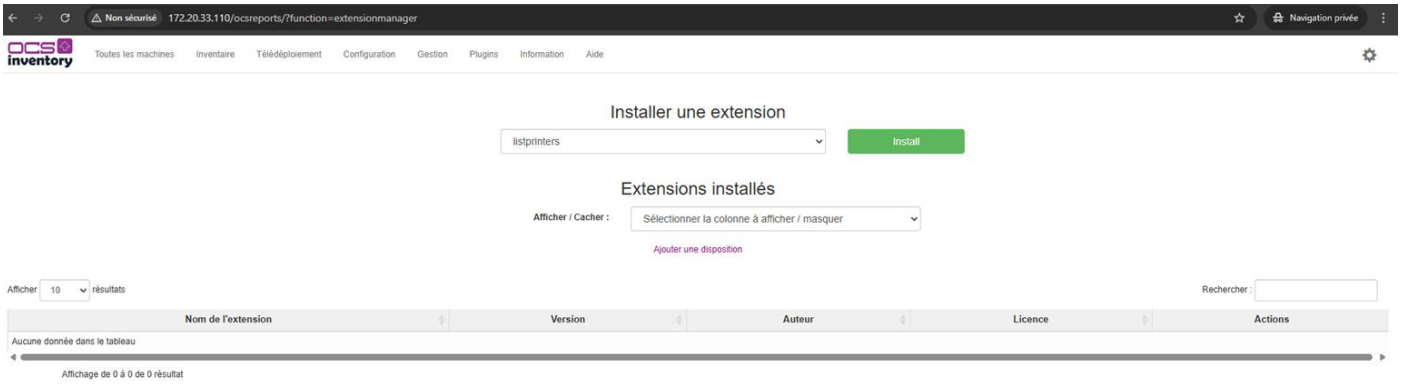
On met les permissions nécessaires pour les plugins.

chown -R www-data:www-data /usr/share/ocsinventory-reports/ocsreports/extensions/

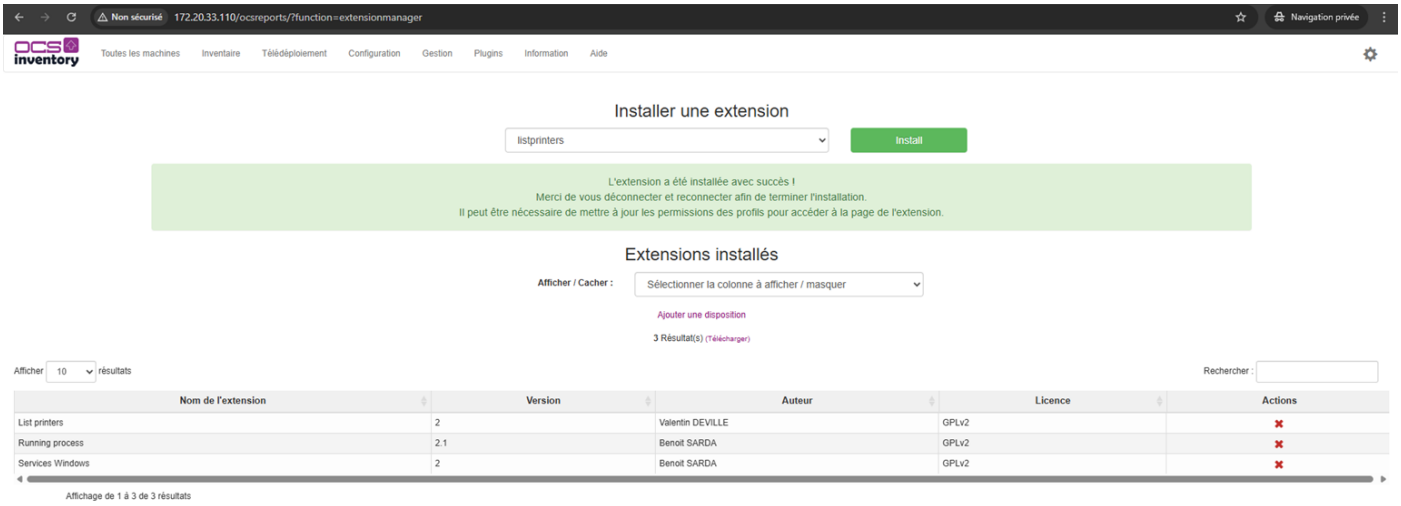
Pour continuer nous redémarrons apache pour appliquer les différents changements.

systemctl restart apache2

Nous pouvons les installer en cliquant sur le bouton vert install et nous faisons la même manipulation pour les trois plugins.



Une fois les trois plugins installés nous voyons qu'ils apparaissent correctement en bas

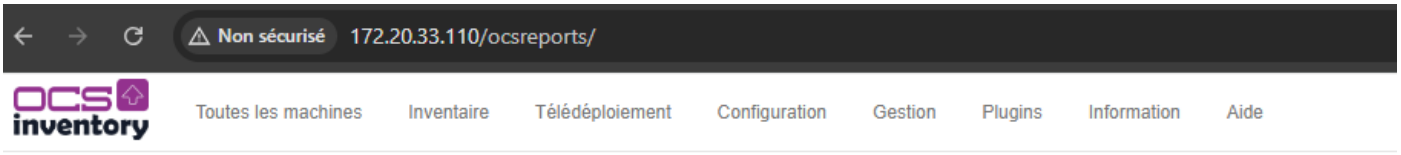


3) Rédiger une procédure explicative pour la mise en œuvre de la découverte réseau ainsi que pour la fonctionnalité de déploiement (télédiffusion). Pour cette dernière fonctionnalité, vous choisirez le déploiement d'un logiciel sous Windows indispensable pour une entreprise (contexte à préciser).

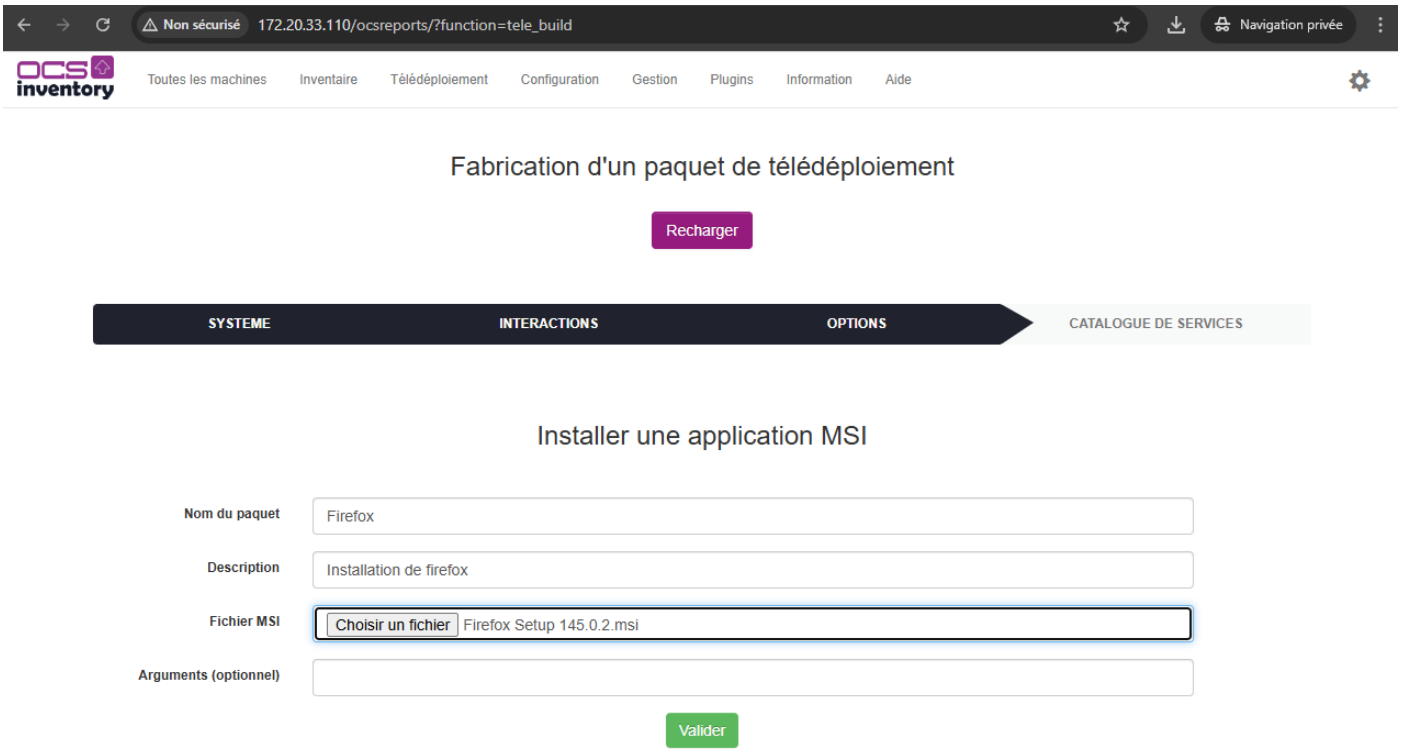
Pour utiliser le télédéploiement nous allons trouver une application à déployer par exemple Firefox mais nous choisissons bien l'extension msi pour que cela fonctionne. Donc nous allons nous rendre sur le site suivant et trouver le fichier de téléchargement.

https://www.firefox.com/fr/browsers/enterprise/#download

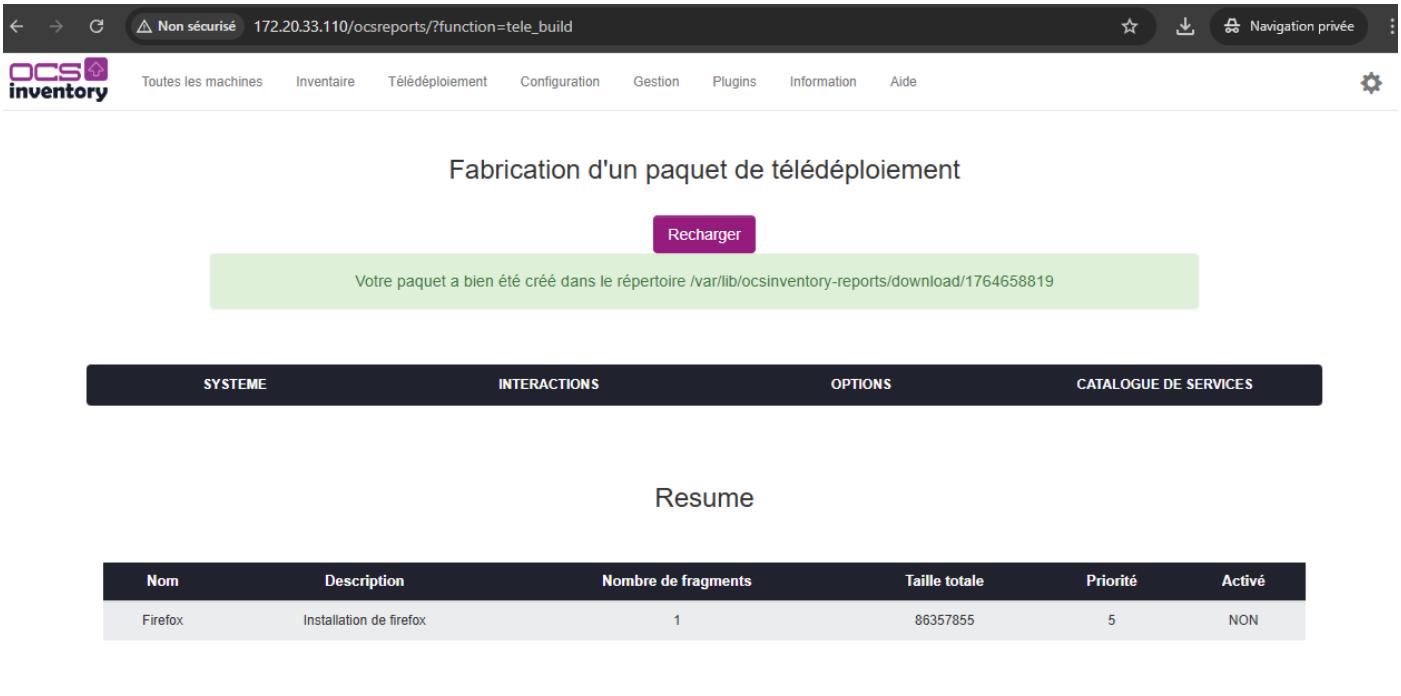
Une fois le fichier télécharger nous allons sur la version web de ocs puis nous allons dans l'onglet télédéploiement.



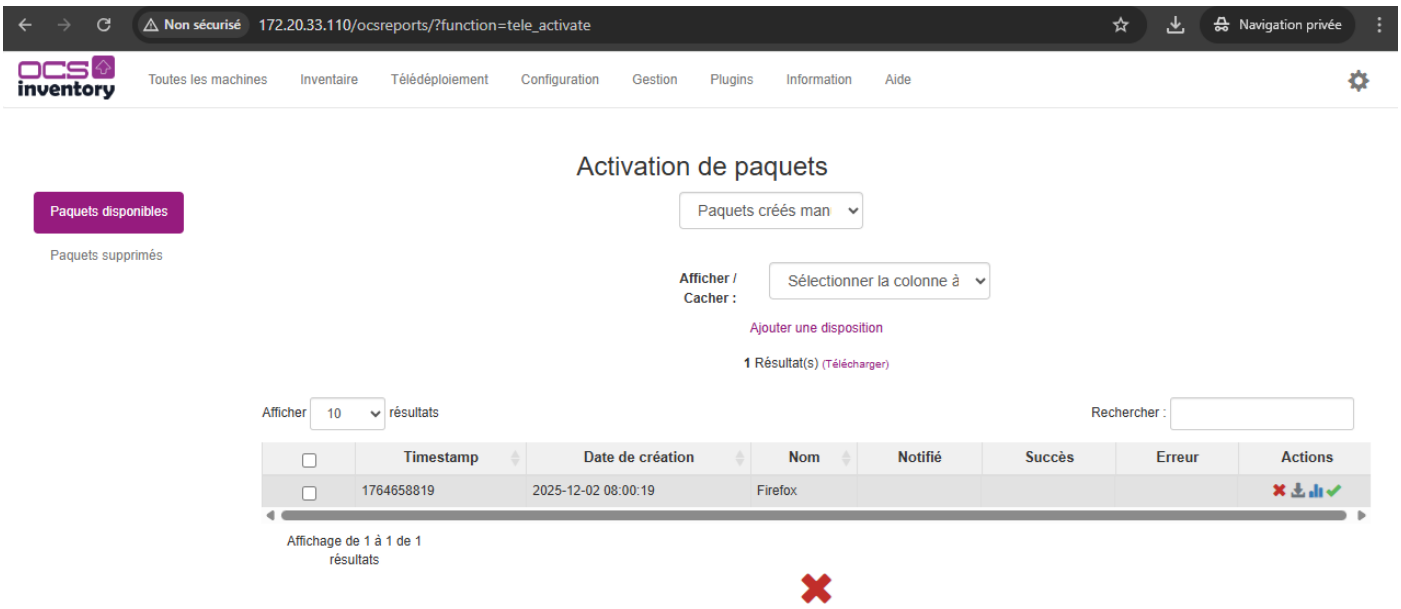
On choisit l'installation pour windows, puis le type de fichier donc msi, puis après on nomme notre installation et on sélectionne notre fichier d'installation puis nous pouvons valider.



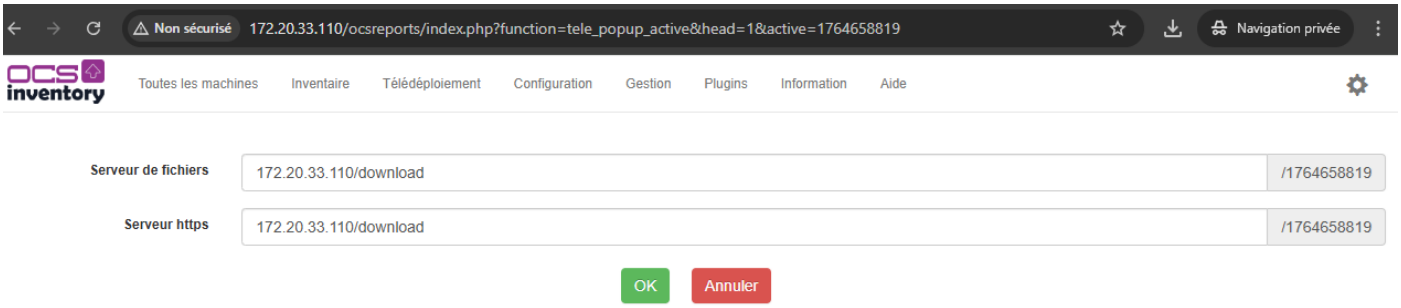
Ensuite nous arrivons sur cette page-là qui nous indique que le fichier de Firefox est chargé et qu'il est prêt à être télédéployer. Or pour le moment il n'est pas activé donc nous allons le faire.



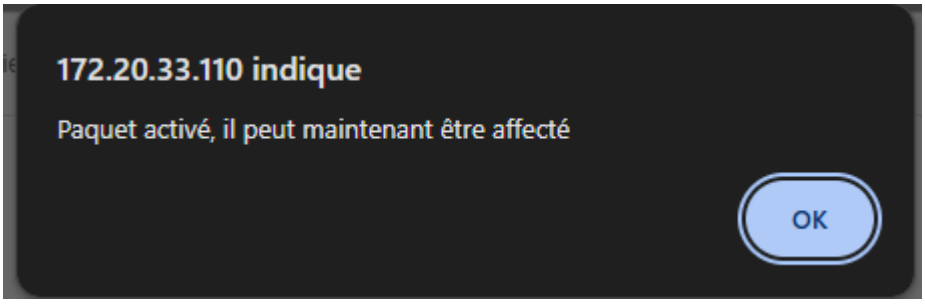
Après nous allons dans l'onglet activation dans télédéploiement puis nous voyons notre fichier Firefox qui apparaît. Pour l'activer il nous suffit de cliquer sur le check vert à droite.



Ensuite nous arrivons sur cette page-là puis pour actionner l'activation nous cliquons sur le bouton vert valider.



Pour finir après avoir cliqué sur le bouton un pop-up va apparaître nous indiquant que le fichier d'installation est actif et qu'il peut être télédéployer maintenant.



3) Mise en Place de CVE Search :

Nous allons installer les dépôts git sur le serveur OCS pour pouvoir ensuite récupérer les dépôts git de CVE Search.

apt install git –y

Une fois git installer nous clonons les dépôts git de CVE Search sur le serveur OCS.

git clone https://github.com/cve-search/cve-search.git

Nous allons configurer l'environnement nécessaire pour utiliser CVE Search et MongoDB, nous allons installer les dépendances puis créer un environnement virtuel isolé pour le projet et ainsi de préparer l'installation de MongoDB.

xargs apt install -y < cve-search/requirements.system

apt install python3-venv –y

apt install python3-virtualenv –y

python3 -m venv /pythonENV/

/pythonENV/bin/pip3 install -r cve-search/requirements.txt

apt install gnupg curl –y

curl -fsSL https://www.mongodb.org/static/pgp/server-8.0.asc | gpg -o /usr/share/keyrings/mongodb-server-8.0.gpg --dearmor

Ensuite nous ajoutons le dépôt de MongoDB sur le serveur OCS.

echo "deb [signed-by=/usr/share/keyrings/mongodb-server-8.0.gpg] https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 main" | tee /etc/apt/sources.list.d/mongodb-org-8.0.list

Après nous mettons à jour la liste des paquets et nous installons les paquets MongoDB.

apt update

apt install -y mongodb-org

Une fois tous les paquets installés nous pouvons activer et démarrer tous les services nécessaires.

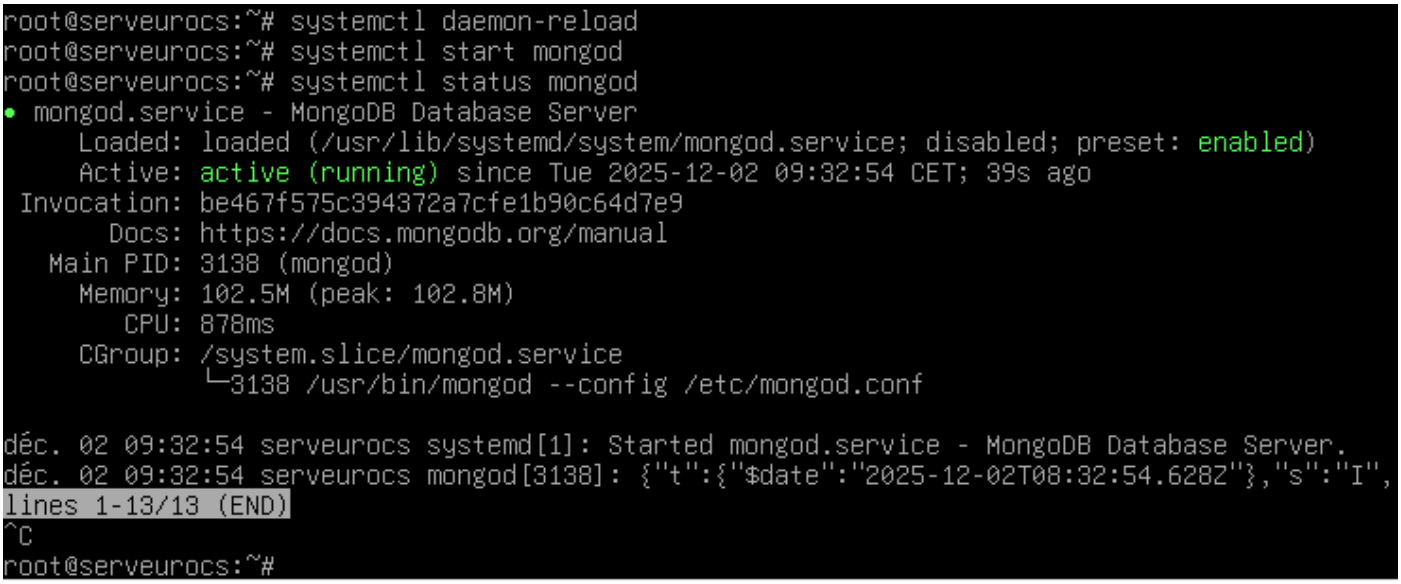
systemctl daemon-reload

systemctl start mongod

systemctl enable mongod

Nous vérifions que le service fonctionne correctement.

systemctl status mongod



Pour continuer nous ajoutons sur notre serveur OCS un utilisateur cve puis nous nous connectons avec sur le serveur OCS.

adduser cve --home /opt/cve

su – cve

Après nous allons configurer un environnement de développement isolé pour le projet CVE Search, garantissant que toutes les dépendances sont en place et sans conflits avec d'autres projets.

virtualenv cve-env

source ./cve-env/bin/activate

git clone https://github.com/cve-search/cve-search.git

cd cve-search

pip3 install -r requirements.txt

exit

Ensuite nous préparons le fichier de configuration de CVE Search et nous mettons à jour la base de données avec les dernières données sur les vulnérabilités.

cp ./cve-search/etc/configuration.ini.sample ./cve-search/etc/configuration.ini

source /pythonENV/bin/activate

./cve-search/sbin/db_updater.py -f –c