

Table des matières

| | |
|---|----|
| 1. Installation de LAMP sur notre Debian | 1 |
| 2. Téléchargement de Wordpress sur notre Debian | 2 |
| 3. Création d'une base de données Wordpress | 2 |
| 4. Décompression de l'archive Wordpress à la racine de notre site | 2 |
| 5. Installation de Wordpress sur notre Debian | 3 |
| 6. Installation d'IIS sur notre Windows Serveur | 4 |
| 7. Mise en réseau privé des différentes VMs | 5 |
| 8. Mise en place de la supervision en ligne de commande | 6 |
| 9. Conclusion | 10 |

J'effectue ce TP en binôme avec Lucas LEGER. Nous ferons donc un seul compte rendu en commun

1. Installation de LAMP sur notre Debian

Pour commencer il faut installer LAMP sur notre VM Debian :

```
sudo apt-get update  
sudo apt-get upgrade  
sudo apt install ca-certificates apt-transport-https
```

Installation d'Apache2 :

```
sudo apt-get install apache2
```

Installation de MySQL :

```
sudo apt-get install default-mysql-server
```

Installation de PHP :

```
wget -q https://packages.sury.org/php/apt.gpg -O- | sudo apt-key add -  
sudo echo "deb https://packages.sury.org/php/ stretch main" | tee /etc/apt/sources.list.d/php.list
```

On update notre VM et on installe les dépendances :

```
sudo apt update  
sudo apt install php php-mysql libapache2-mod-php
```

2. Téléchargement de Wordpress sur notre Debian

Téléchargement de la dernière version de Wordpress :

```
cd /tmp  
wget https://wordpress.org/latest.zip
```

3. Création d'une base de données Wordpress

Connexion à mysql ou MariaDB :

```
mysql -u root -p
```

Création de la base données :

```
CREATE DATABASE wp202110_itconnect;
```

Listons nos base de données pour vérifier la bonne création :

```
SHOW DATABASES;
```

Création d'un utilisateur avec tous les droits sur notre base de données :

```
CREATE USER 'admin'@'localhost' IDENTIFIED BY 'admin';
```

Don des droits utilisateurs :

```
GRANT ALL PRIVILEGES ON wp202110_itconnect.* TO admin@localhost;
```

Actualisation de nos droits :

```
FLUSH PRIVILEGES;
```

On peut enfin quitter MySQL ou MariaDB :

```
exit
```

4. Décompression de l'archive Wordpress à la racine de notre site

```
sudo rm /var/www/html/index.html
```

Installation du paquet zip :

```
sudo apt-get update  
sudo apt-get install zip
```

Décompression de l'archive :

```
sudo unzip latest.zip -d /var/www/html
```

Déplacement dans le répertoire de stockage des sites :

```
cd /var/www/html  
sudo mv wordpress/* /var/www/html/  
sudo rm wordpress/ -Rf
```

Don des droits pour l'utilisateur correspondant à Apache sur tous les fichiers du wordpress :

```
sudo chown -R www-data:www-data /var/www/html/
```

5. Installation de Wordpress sur notre Debian

Il faut se rendre sur la page web de notre wordpress :

```
http://IP_Serveur/
```

Une fois sur la page web de notre serveur il faut installer wordpress de façon graphique

- 1) Choisir la langue du site et de l'interface wordpress
- 2) Configuration de notre base de données :

Nom de la base de données : dans cet exemple, ce sera "wp202110_itconnect"

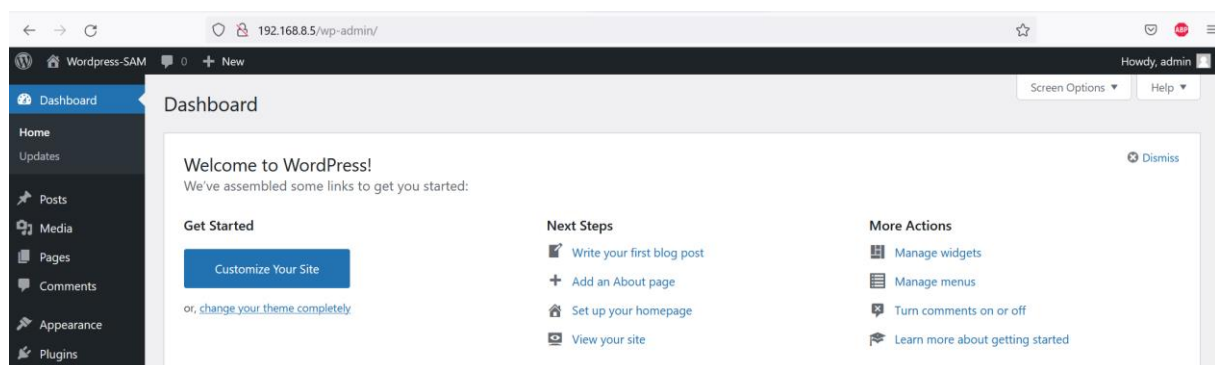
Identifiant : le nom de l'utilisateur qui a les droits sur la base de données, en l'occurrence "admin"

Mot de passe : le mot de passe de cet utilisateur (ici admin)

Adresse de la base de données : si le serveur Web et la base de données sont sur le même serveur, indiquez "localhost", sinon indiquez l'adresse IP du serveur distant

Préfixe des tables : chaque table de la base de données WordPress aura un préfixe. **Par défaut, ce préfixe est "wp"** donc par exemple la table des utilisateurs sera nommée "wp_users". **Il faut personnaliser ce préfixe et le rendre un peu plus aléatoire pour des raisons de sécurité.** Pour ma part, je vais partir sur "web14_", mais vous pouvez prendre aussi quelque chose d'aléatoire comme "sg389_".

Une fois toute l'installation faite on devrait arriver sur une interface comme celle-ci :



6. Installation d'IIS sur notre Windows Serveur

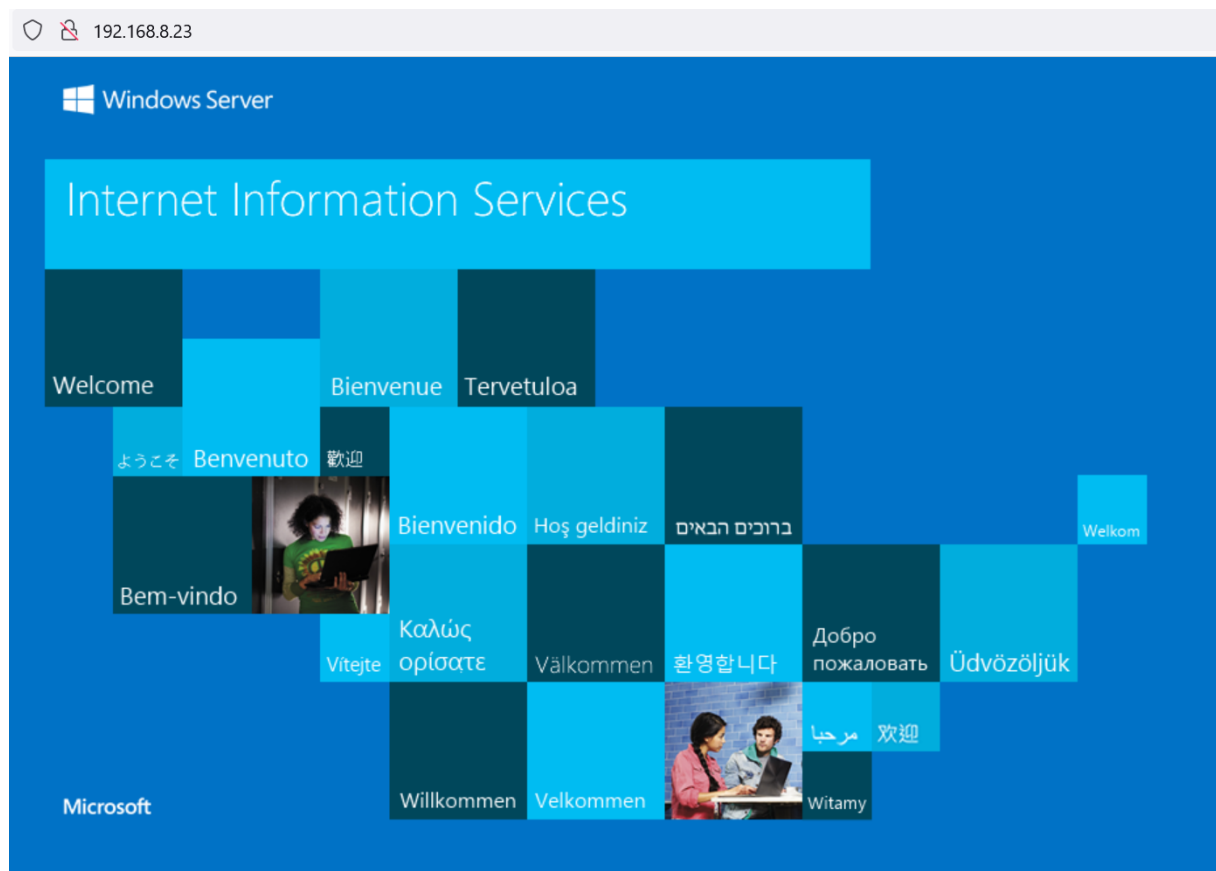
Sur notre Windows Serveur il faut aller en mode Powershell :

Powershell

Une fois dans le mode Powershell il suffit de passer la commande suivante pour installer IIS :

`Install-WindowsFeature -name Web-Server -IncludeManagementTools`

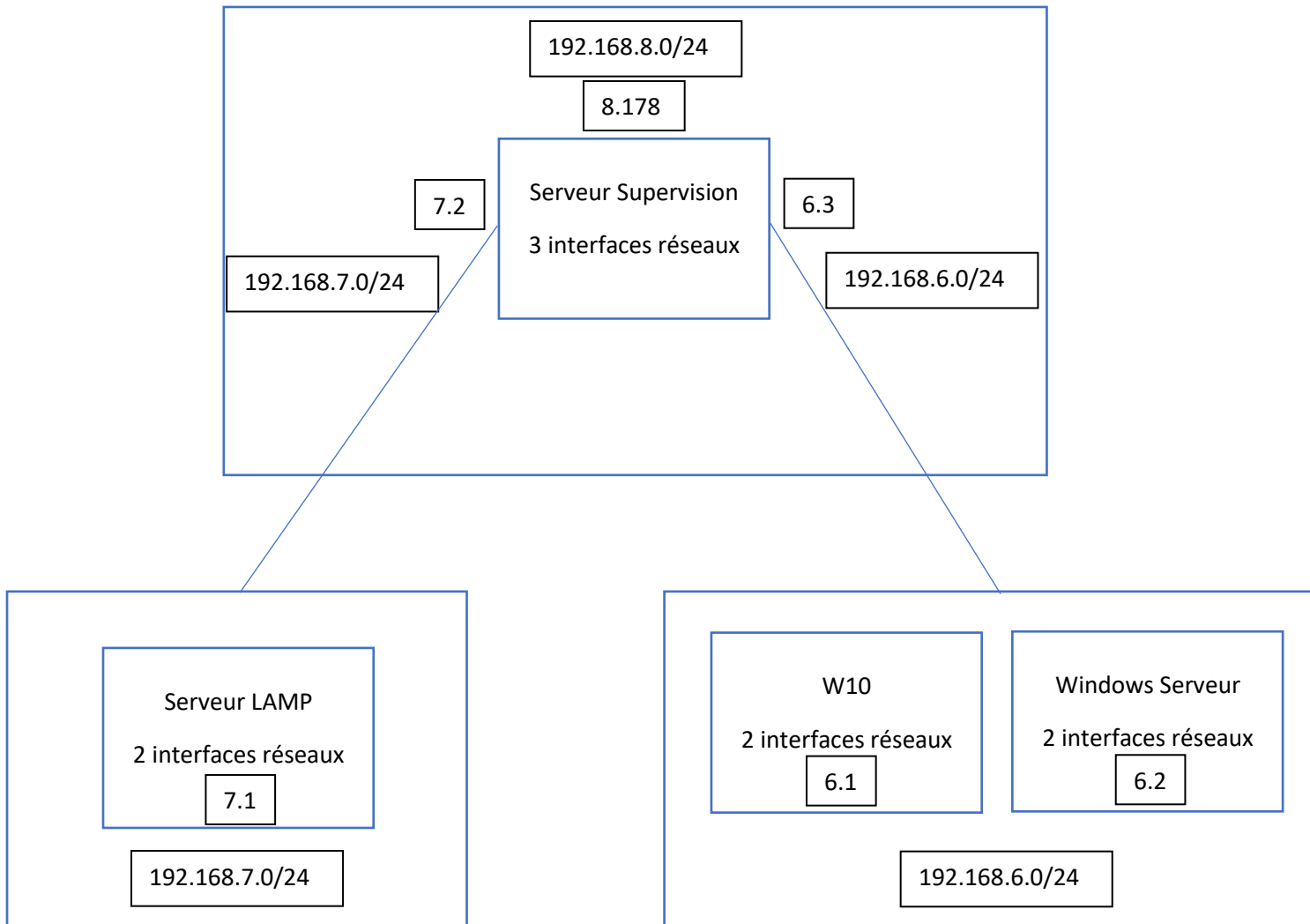
Pour vérifier que IIS est bien installé on peut aller sur notre navigateur Web et entrer l'IP de notre serveur :



7. Mise en réseau privé des différentes VMs

Maintenant que nous avons installé tous les outils nécessaires sur nos VMs il faut que nous les passions dans des réseaux différents. D'un côté nous aurons un réseau pour les machines Windows et de l'autre côté un réseau pour notre machine Linux. Le serveur se trouve aussi dans un réseau séparé mais est relié aux 2 réseaux différents afin de mettre en place la supervision

Voici le schéma de nos réseaux :



8. Mise en place de la supervision en ligne de commande

1^{ère} supervision : check NRPE sur notre Windows 10

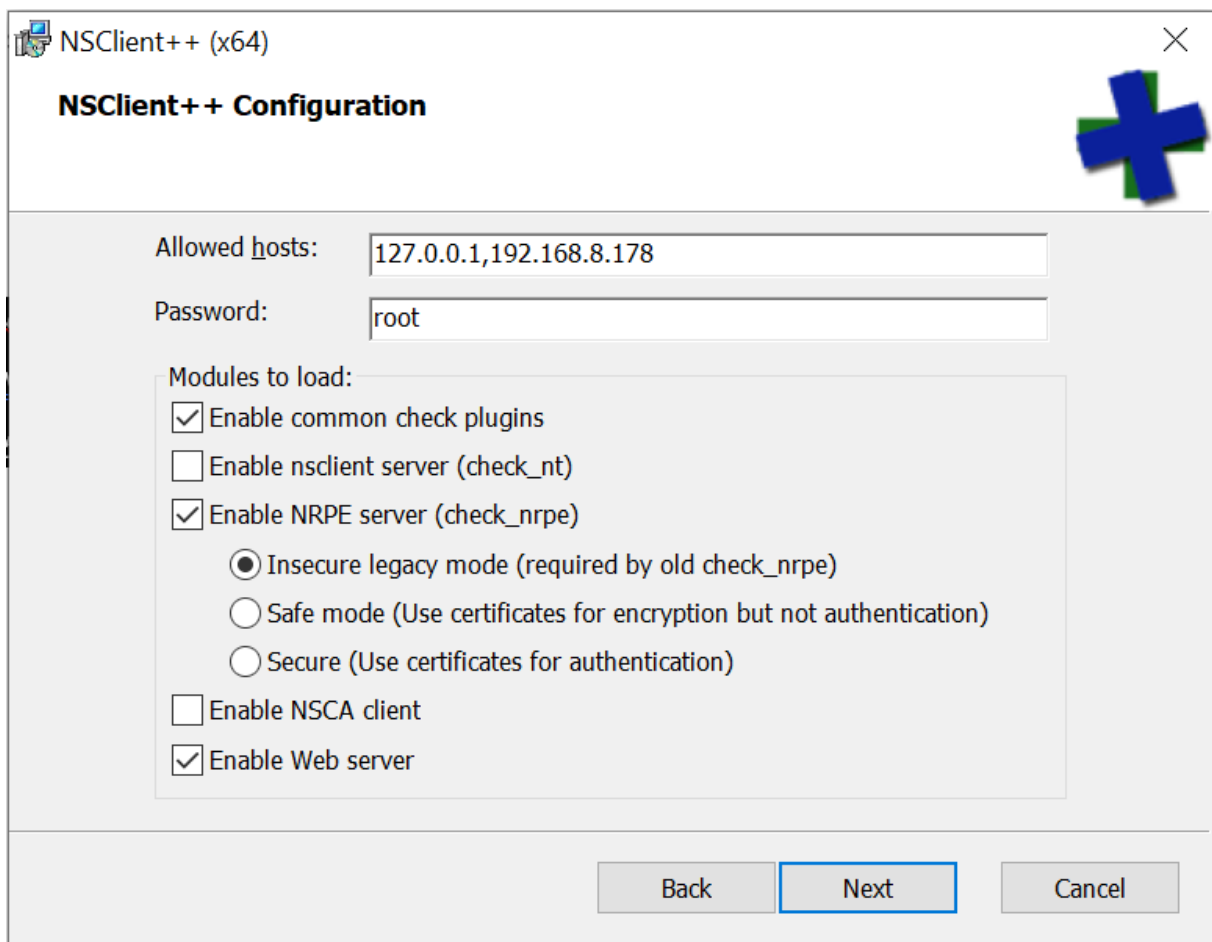
Pour effectuer des checks NRPE il faut commencer par installer le démon NRPE sur notre serveur :

```
yum install centreon-nrpe3-plugin
```

Il faut ensuite installer NSClient++ sur notre Windows (on installe la version generic puis typical) :

Quand on arrive sur cet onglet il faut mettre de cette façon-là :

Il faut mettre l'IP de notre serveur de supervision dans les hôtes autorisés



Il faut aussi activer les différents modules dans le fichier nsclient.ini pour pouvoir passer des commandes en CLI :

En ligne de commande (administrateur) Windows se déplacer dans le dossier suivant :

```
Cd "%Program Files%\NSClient++"
```

Par exemple le module de check de base :

```
nscp settings --path /modules --key "CheckSystem" --set "enabled"
```

Il faut ensuite redémarrer le service en mode administrateur :

```
Net stop nscp && net start nscp
```

Une fois NSclient++ installé sur notre Windows on peut venir passer des commandes de supervision sur notre serveur :

```
[root@localhost libexec]# /usr/lib64/nagios/plugins/check_centreon_nrpe3 -H 192.168.6.1 -c check_version  
  
CHECK_NRPE: Invalid packet version received from server.  
0.5.2.35 2018-01-28  
  
[root@localhost libexec]# /usr/lib64/nagios/plugins/check_centreon_nrpe3 -H 192.168.6.1 -c check_os_version  
  
CHECK_NRPE: Invalid packet version received from server.  
OK: Windows 10 (10.0.19044) |'version'=100;50;50  
  
[root@localhost usr]# /usr/lib64/nagios/plugins/check_centreon_nrpe3 -H 192.168.6.1 -c check_process  
  
CHECK_NRPE: Invalid packet version received from server.  
  
OK: all processes are ok. |'winlogon.exe state'=1;0;0 'lsass.exe state'=1;0;0 'svchost.exe state'=1;0;0  
'fontdrvhost.exe state'=1;0;0 'fontdrvhost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'dwm.exe state'=1;0;0  
'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0  
'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0  
'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'spoolsv.exe state'=1;0;0  
'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'svchost.exe state'=1;0;0  
'sihost.exe state'=1;0;0 'svchost.exe state'=1;0;0 'taskhostw.exe state'=1;0;0 'svchost.exe state'=1;0;0  
'Explorer.EXE state'=1;0;0 'SearchIndexer.exe state'=1;0;0 'ctfmon.exe state'=1;0;0 'svchost.exe  
state'=1;0;0 'StartMenuExperienceHost.exe state'=1;0;0 'RuntimeBroker.exe state'=1;0;0 'SearchApp.exe  
state'=1;0;0 'RuntimeBroker.exe state'=1;0;0 'SkypeApp.exe state'=1;0;0 'SkypeBackgroundHost
```

Pour pouvoir passer plus de commandes comme le check_cpu il faut autoriser les arguments dans le fichier nsclient.ini :

```
nsclient++ settings --path /settings/NRPE/server --key "allow arguments" --set true  
nsclient++ settings --path /settings/NRPE/server --key "allow nasty characters" --set true
```

Maintenant nous pouvons voir la consommation CPU :

```
[root@localhost nagios-plugins-2.2.1]# /usr/lib64/nagios/plugins/check_centreon_nrpe3 -H 192.168.6.1 -c check_cpu -a "warn=load > 60" "crit=load > 70"  
  
CHECK_NRPE: Invalid packet version received from server.  
  
OK: CPU load is ok. |'total 5m'=0%;60;70 'total 1m'=0%;60;70 'total 5s'=3%;60;70
```

Si pendant les commandes on est bloqué sur le port 5666 il faut utiliser la commande suivante :

```
iptables -I INPUT -p tcp -m tcp --dport 5666 -j ACCEPT
```

Nous pouvons aussi avoir des erreurs avec le certificat SSL donc nous pouvons suivre le lien suivant pour résoudre le problème :

<https://hodza.net/2019/09/21/failed-to-establish-secure-connection-sslv3-alert-handshake-failure-1040/>

Voici le lien vers le site de nsclient qui regroupe les différentes commandes que nous pouvons passer dans le terminal à condition que les modules soient activés dans le nsclient.ini :

<https://docs.nsclient.org/reference/>

2^{ème} supervision : SNMP sur Debian

Tout d'abord on installe SNMP sur notre serveur web Debian et sur notre serveur de supervision grâce à la commande suivante :

```
apt install snmpd snmp libsnmp-dev
```

On va ensuite renseigner notre communauté SNMP dans le fichier /etc/snmp/snmpd.conf :

```
rocommunity public 192.168.7.2
```

Où 192.168.7.2 est l'adresse IP de notre serveur de supervision et où « public » est notre communauté.

On peut désormais redémarrer le service grâce à la commande suivante :

```
systemctl restart snmpd
```

Il ne nous reste plus qu'à installer nos plugins centreon sur notre serveur de supervision :

```
yum install centreon-plugin-OperatingSystems-Linux-Snmp
```

On peut maintenant faire nos tests.

Dans un premier temps, depuis notre serveur de supervision, nous allons exécuter la commande suivante :

```
snmpwalk -v 2c -c public 192.168.7.1 system
```


Nous allons ainsi voir la MIB de notre serveur web apparaitre.

```
[root@localhost ~]# snmpwalk -v 2c -c public 192.168.7.1 system
SNMPv2-MIB::sysDescr.0 = STRING: Linux debian-sup 5.10.0-9-amd64 #1 SMP Debian 5.10.70-1 (2021-09-30) x86_64
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (358576) 0:59:45.76
SNMPv2-MIB::sysContact.0 = STRING: Me <me@example.org>
SNMPv2-MIB::sysName.0 = STRING: debian-sup
SNMPv2-MIB::sysLocation.0 = STRING: Sitting on the Dock of the Bay
SNMPv2-MIB::sysServices.0 = INTEGER: 72
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORID.1 = OID: SNMP-FRAMEWORK-MIB::snmpFrameworkMIBCompliance
SNMPv2-MIB::sysORID.2 = OID: SNMP-MPD-MIB::snmpMPDCompliance
SNMPv2-MIB::sysORID.3 = OID: SNMP-USER-BASED-SM-MIB::usmMIBCompliance
SNMPv2-MIB::sysORID.4 = OID: SNMPv2-MIB::snmpMIB
SNMPv2-MIB::sysORID.5 = OID: SNMP-VIEW-BASED-ACM-MIB::vacmBasicGroup
SNMPv2-MIB::sysORID.6 = OID: TCP-MIB::tcpMIB
SNMPv2-MIB::sysORID.7 = OID: UDP-MIB::udpMIB
SNMPv2-MIB::sysORID.8 = OID: IP-MIB::ip
SNMPv2-MIB::sysORID.9 = OID: SNMP-NOTIFICATION-MIB::snmpNotifyFullCompliance
SNMPv2-MIB::sysORID.10 = OID: NOTIFICATION-LOG-MIB::notificationLogMIB
SNMPv2-MIB::sysORDescr.1 = STRING: The SNMP Management Architecture MIB.
SNMPv2-MIB::sysORDescr.2 = STRING: The MIB for Message Processing and Dispatching.
SNMPv2-MIB::sysORDescr.3 = STRING: The management information definitions for the SNMP User-based Security Model.
SNMPv2-MIB::sysORDescr.4 = STRING: The MIB module for SNMPv2 entities
SNMPv2-MIB::sysORDescr.5 = STRING: View-based Access Control Model for SNMP.
SNMPv2-MIB::sysORDescr.6 = STRING: The MIB module for managing TCP implementations
SNMPv2-MIB::sysORDescr.7 = STRING: The MIB module for managing UDP implementations
SNMPv2-MIB::sysORDescr.8 = STRING: The MIB module for managing IP and ICMP implementations
SNMPv2-MIB::sysORDescr.9 = STRING: The MIB modules for managing SNMP Notification, plus filtering.
SNMPv2-MIB::sysORDescr.10 = STRING: The MIB module for logging SNMP Notifications.
SNMPv2-MIB::sysORUpTime.1 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.2 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.3 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.4 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.5 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.6 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.7 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.8 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.9 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.10 = Timeticks: (0) 0:00:00.00
```

Ensuite, nous pouvons essayer la commande suivante :

```
/usr/lib/centreon/plugins//centreon_linux_snmp.pl --plugin=os::linux::snmp::plugin --mode=cpu --hostname=192.168.7.1 --snmp-version='2c' --snmp-community='public' -verbose
```

Celle-ci va nous permettre de superviser notre CPU sur notre serveur web.

```
[root@localhost ~]# /usr/lib/centreon/plugins//centreon_linux_snmp.pl --plugin=os::linux::snmp::plugin --mode=cpu --hostname=192.168.7.1 --snmp-version='2c' --snmp-community='public' -verbose
OK: 1 CPU(s) average usage is 1.00 % - CPU '0' usage : 1.00 % | 'total_cpu_avg'=1.00%;;0;100 'cpu'=1.00%;;0;100
CPU '0' usage : 1.00 %
```

Finalement, nous allons pouvoir faire notre troisième supervision : un test du CPU de notre serveur web via un check SSH.

Tout d'abord nous allons générer une clé SSH.

```
ssh-keygen -t ed25519 -a 100
```

On la copie ensuite sur notre serveur web, dont l'adresse IP est 192.168.7.1 en se connectant en SSH avec l'un de nos utilisateurs (ici l'utilisateur lucas).

```
ssh-copy-id -i .ssh/id_ed25519.pub lucas@192.168.7.1
```

Ensuite, nous allons installer les plugins centreon sur notre serveur de supervision grâce à cette commande :

```
yum install centreon-plugin-OperatingSystems-Linux-Ssh.noarch
```

Nous pouvons désormais tester en faisant la supervision du CPU du serveur web comme mentionné plus haut grâce à cette commande :

```
/usr/lib/centreon/plugins//centreon_linux_ssh.pl | --plugin=os::linux::local::plugin | --mode=cpu | --hostname='192.168.7.1' | --ssh-backend='libssh' | --ssh-username='lucas' | --ssh-password='azerty-85' | --ssh-port='22' | --warning-core='60' | --critical-core='70' | --warning-average='60' | --critical-average='75' | --verbose | --use-new-perfdata
```

Cela nous retourne donc le résultat :

```
[root@localhost ~]# /usr/lib/centreon/plugins//centreon_linux_snmp.pl --plugin=os::linux::snmp::plugin --mode=cpu --hostname=192.168.7.1 --snmp-version='2c' --snmp-community='public' --verbose
OK: 1 CPU(s) average usage is 1.00 % - CPU '0' usage : 1.00 % | 'total_cpu_avg'=1.00%;;0;100 'cpu'=1.00%;;0;100
CPU '0' usage : 1.00 %
[root@localhost ~]# /usr/lib/centreon/plugins//centreon_linux_ssh.pl --plugin=os::linux::local::plugin --mode=cpu --hostname='192.168.7.1' --ssh-backend='libssh' --ssh-username='lucas' --ssh-password='azerty-85' --ssh-port='22' --warning-core='60' --critical-core='70' --warning-average='60' --critical-average='75' --verbose --use-new-perfdata
OK: CPU(s) average usage is 1.58 % - CPU '0' usage : 1.58 % | 'cpu.utilization.percentage'=1.58%;0;60;0;75;0;100 '0#core.cpu.utilization.percentage'=1.58%;0;60;0;75;0;100
CPU '0' usage : 1.58 %
```

9. Conclusion

Nous avons donc mis en place trois types de check. Ces différents procédés sont applicables sur plusieurs systèmes d'exploitation.

Dans la partie précédente, nous avons effectué un check NRPE sur un client Windows 10. Cependant, il est tout à fait possible de l'exécuter sur un Windows Server ou sur une Debian par exemple.

De même pour le check SNMP. Nous l'avons effectué sur notre serveur web Debian mais il est aussi possible de le configurer pour Windows et Windows Server.

Concernant le check SSH, nous avons mis cela en place sur notre serveur web Debian. Nous n'avons pas trouvé de sources selon lesquelles il est possible de le mettre en place sur un système d'exploitation Windows. Cependant, théoriquement, cela serait plus compliqué du fait que c'est assez difficile d'activer le service SSH sur Windows et Windows Server.

Globalement, le check SSH est plus simple à mettre en place par rapport aux deux autres du fait qu'il demande moins de temps et de commande. Cependant, il peut y avoir des problèmes avec l'activation du service SSH ou alors avec le transfert de la clé d'une machine à l'autre.

Pour ce qui est du check SNMP, il n'oppose aucune difficulté et est réellement simple à mettre en place. C'est d'ailleurs le protocole le plus utilisé pour la supervision des systèmes. Comme nous avons pu le voir dans la partie dédiée, il suffit de configurer la communauté dans le fichier de configuration de SNMP sur notre serveur WEB et d'installer les plugins sur notre serveur de supervision pour que cela fonctionne.

Pour ce qui est du check NRPE, il n'oppose pas non plus de difficultés à conditions de bien installer le plugin centreon sur le serveur de supervision et de bien configurer le nsclient++ (nsclient.ini) pour pouvoir passer différentes commandes