DIACK Mariama **M1 SSIM**

REPUBLIQUE DU SENEGAL

UN PEUPLE-UN BUT-UNE FOI

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR, DE LA RECHERCHE ET DE L'INNOVATION

DIRECTION GENERALE DE L'ENSEIGNEMENT SUPERIEUR PRIVE

INSTITUT SUPERIEUR D'INFORMATIQUE

Nagios® GROUPE ISI Rocky Linux™

Projet de Nagios sous Rocky Linux 8

Supervision de périphériques et services sous Nagios avec notifications par Email

Présenté par :

Mlle. Mariama DIACK

Sous la direction de :

M. Massamba LO

Année académique : 2023-2024

PLAN

I. Introduction

II. Définition

III. Informations techniques

IV. Déploiement

V. Conclusion

I. Introduction

Dans un monde où les systèmes informatiques sont de plus en plus complexes et interconnectés, la surveillance de ces systèmes est devenue cruciale pour garantir leur bon fonctionnement. Nagios, un logiciel open source, joue un rôle clé dans la surveillance des infrastructures IT en offrant des capacités de surveillance à la fois simples et robustes.

II. Définition

Nagios est une plateforme de surveillance open-source conçue pour surveiller les systèmes, les réseaux, et les infrastructures IT. Créé par Ethan Galstad en 1999 sous le nom de NetSaint, il a été renommé Nagios et est devenu un outil de référence dans le domaine de la surveillance informatique. Nagios permet aux administrateurs de surveiller les équipements et services essentiels au sein d'une infrastructure, tels que les serveurs, les bases de données, les applications, et même les services cloud.

Grâce à une interface web intuitive, Nagios fournit des alertes en temps réel lorsqu'un problème survient et génère des rapports détaillés pour analyser les performances. Parmi ses principales fonctionnalités, on retrouve la surveillance des services (comme HTTP, SMTP, DNS), des ressources système (CPU, RAM, espace disque), ainsi que des performances réseau (utilisation de la bande passante, latence). De plus, Nagios offre des alertes et notifications personnalisées, ainsi que des graphiques de performance, permettant de prévenir les problèmes avant qu'ils n'affectent les utilisateurs finaux.

III. Informations techniques

1 Utilisation de DNS

Pour que Nagios puisse surveiller efficacement un réseau, la résolution de noms de domaine (DNS) est fondamentale. DNS permet de traduire les noms de domaine en adresses IP, facilitant ainsi la surveillance des machines et services à

travers un réseau. Nagios utilise le DNS pour résoudre les noms des hôtes surveillés, garantissant une surveillance fiable et précise.

2 Intégration avec apache ou Nginx

L'interface web de Nagios repose sur un serveur web pour être accessible. Généralement, les serveurs Apache ou Nginx sont utilisés à cette fin. Apache est souvent préféré pour sa robustesse et ses nombreuses fonctionnalités, tandis que Nginx est apprécié pour sa performance et son efficacité. Quel que soit le serveur choisi, il est configuré pour servir l'interface de Nagios, permettant ainsi aux administrateurs de surveiller l'infrastructure depuis n'importe quel navigateur web.

3 Fonctionnement de Nagios

Nagios est conçu pour surveiller les services critiques comme HTTP, SMTP, POP3, et bien d'autres. Il utilise des plugins pour vérifier l'état de ces services. Si un problème est détecté, Nagios envoie des alertes via différents canaux (email, SMS, etc.) pour que les administrateurs puissent intervenir rapidement. Nagios est hautement configurable et peut être étendu avec de nombreux plugins tiers pour répondre aux besoins spécifiques de chaque infrastructure.

IV. Déploiement avec Rocky Linux 8

Rocky Linux est une distribution Linux open-source, créée par Gregory Kurtzer en hommage à Rocky McGaugh, co-fondateur de CentOS. Elle a été développée comme une alternative gratuite à Red Hat Enterprise Linux (RHEL) après les changements apportés à CentOS. Rocky Linux offre une compatibilité binaire complète avec RHEL, garantissant stabilité, sécurité et fiabilité pour les environnements de production. Géré par la communauté, Rocky Linux se présente comme une solution pérenne pour les entreprises cherchant une alternative à CentOS et RHEL, sans les contraintes de licences commerciales.

- ## Configuration de DNS

Nous commencerons d'abord à faire un test

```
[root@10 ~]# dhclient
[root@10 ~]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=110 time=65.6 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=110 time=62.2 ms
^C
```

Puis installer les packets

```
[root@10 ~]# yum install bind bind-utils
Dernière vérification de l'expiration des métadonnées effectuée il y a 0:07:51 l
e jeu. 14 mars 2024 17:15:20 EDT.
Le paquet bind-utils-32:9.11.36-11.el8.x86_64 est déjà installé.
Dépendances résolues.
================================================================================
```

Nous passons ensuite à l'étape des configurations proprement dites

```
[root@10 ~]# ifconfig enp0s3 192.168.1.1
```

Nous allons éditer le fichier de configuration Hosts pour donner les informations comme le nom du serveur, nom de domaine et l'adresse ip

```
[root@10 ~]# vim /etc/hosts
```

```
127.0.0.1 server1   localhost localhost.localdomain localhost4 localhost4.locald
omain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.1.1 server1 server1.diack.sn server1
~
```

Nous allons éditer le fichier resolv.conf responsable de la résolution du nom de domaine en adresse ip

```
[root@10 ~]# vim /etc/resolv.conf
```

```
search diack.sn
nameserver 192.168.1.1
~
```

Nous allons éditer le fichier named.conf fixer notre @ip et @Réseau

```
[root@10 ~]# vim /etc/named.conf
```

```
10 options {
11         listen-on port 53 { 127.0.0.1; 192.168.1.1;};
12         listen-on-v6 port 53 { ::1; };
13         directory       "/var/named";
14         dump-file       "/var/named/data/cache_dump.db";
15         statistics-file "/var/named/data/named_stats.txt";
16         memstatistics-file "/var/named/data/named_mem_stats.txt";
17         secroots-file   "/var/named/data/named.secroots";
18         recursing-file  "/var/named/data/named.recursing";
19         allow-query     { localhost; 192.168.1.0/24;};
20
```

La création des zones direct et inverse

```
56 zone "diack.sn" IN {
57           type master;
58           file "direct";
59 };
60 zone "1.168.192.in.addr.arpa" IN {
61           type master;
62           file "inverse";
63 };
```

Nous allons nous déplacer au niveau du répertoire named avec la commande cd puis copier avec la commande cp le contenu du fichier exemple de named.localhost dans direct puis apporter les modification

```
[root@10 ~]# vim /etc/named.conf
[root@10 ~]# cd /var/named
[root@10 named]# ls
data  dynamic  named.ca  named.empty  named.localhost  named.loopback  slaves
[root@10 named]# cp named.localhost direct
```

Enfin éditer le fichier direct pour modifier

```
[root@10 named]# vim direct
```

```
$TTL 1D
@       IN SOA  server1.diack.sn. root.diack.sn. (
                                        0       ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum
        IN      NS      server1.diack.sn.
server1 IN      A       192.168.1.1
www     IN      CNAME   server1.diack.sn.
~
```

Copier le contenu de direct dans inverse et modifier juste la dernière ligne

```
[root@10 named]# cp direct inverse
```

```
[root@10 named]# vim inverse
```

```
$TTL 1D
@       IN SOA  server1.diack.sn. root.diack.sn. (
                                        0       ; serial
                                        1D      ; refresh
                                        1H      ; retry
                                        1W      ; expire
                                        3H )    ; minimum
        IN      NS      server1.diack.sn.
server1 IN      A       192.168.1.1
1       IN      PTR     server1.diack.sn.
```

Vérifier s'il n y a pas d'erreur au niveau des fichier named.conf et les zones

```
[root@10 named]# systemctl restart named
```

Parfois après la configuration tout marche mais lorsqu'on le service DNS n'est pas opérationnel donc il suffit tout simplement d'activer les droits concernant les fichier direct et inverse

```
[root@10 named]# chmod 640 direct
[root@10 named]# chmod 640 inverse
[root@10 named]# chown -R named:named direct
[root@10 named]# chown -R named:named inverse
```

Redémarrons le service de nouveau avec systemctl restart named et vérifions le statut de named avec la commande systemctl status named

```
[root@10 named]# systemctl status named
● named.service - Berkeley Internet Name Domain (DNS)
   Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; vendor prese
   Active: active (running) since Thu 2024-03-14 19:11:39 EDT; 2min 23s ago
 Main PID: 5013 (named)
    Tasks: 4 (limit: 11003)
   Memory: 17.0M
   CGroup: /system.slice/named.service
           └─5013 /usr/sbin/named -u named -c /etc/named.conf

mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
mars 14 19:13:56 10.0.2.15 named[5013]: network unreachable resolving 'mirrorli
```

- ## Configuration d'Apache http Server

Le prérequis ici c'est d'avoir un service DNS Opérationnel.

Passons à présent à l'installation des packages

```
[root@localhost ~]# yum install httpd
```

Editons le fichier de configuration httpd.conf

```
[root@localhost ~]# vim /etc/httpd/conf/httpd.conf
```

A la ligne 89 ajouter le nom de domaine

```
89 ServerAdmin root@diack.sn
```

A la ligne 89 ajouter le nom du site

```
98 ServerName www.diack.sn:80
```

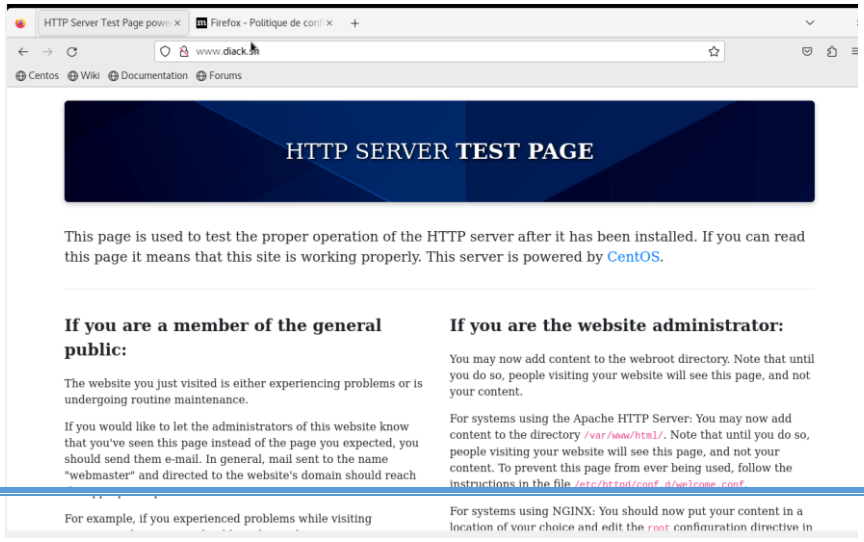
A la ligne 154 mettre ALL

```
154     AllowOverride ALL
```

Redémarrer les services

```
[root@localhost ~]# systemctl restart httpd
[root@localhost ~]# systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr
/lib/systemd/system/httpd.service.
[root@localhost ~]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor prese>
   Active: active (running) since Thu 2024-03-14 21:56:44 EDT; 13s ago
     Docs: man:httpd.service(8)
 Main PID: 5261 (httpd)
   Status: "Running, listening on: port 80"
    Tasks: 213 (limit: 11003)
   Memory: 17.6M
   CGroup: /system.slice/httpd.service
           ├─5261 /usr/sbin/httpd -DFOREGROUND
```

On teste via le navigateur



On crée notre propre page html

```
[root@localhost ~]# vim /var/www/html/index.html
```

```
<h1> Bienvenue chez Maryam Global Business </h1>
```

On redémarre le service

```
[root@localhost ~]# systemctl restart httpd
```

Nous allons tester

- ## Configuration de Nagios

Installation des dépendances requises

Avant d'installer Nagios Core, nous devons installer certaines bibliothèques requises pour que Nagios Core fonctionne correctement.

```
yum install -y httpd httpd-tools php gcc glibc glibc-common gd gd-devel make net-snmp
```

Créons maintenant un utilisateur et un groupe Nagios pour celui-ci.

```
useradd nagios
groupadd nagcmd
```

Ensuite, ajoutons les utilisateurs nagios et apache au groupe **nagcmd** .

```
usermod -G nagcmd nagios
usermod -G nagcmd apache
```

Téléchargeons Nagios Core et Nagios Plugin

Créons un répertoire pour installer Nagios en utilisant la commande suivante.

```
mkdir /root/nagios
cd /root/nagios
```

Téléchargeons Nagios et les plugins Nagios en utilisant ces deux commandes.

```
wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.6.tar.gz
wget https://nagios-plugins.org/download/nagios-plugins-2.2.1.tar.gz
```

Extrayons ensuite Nagios et ses plugins.

```
tar -xvf nagios-4.4.6.tar.gz
tar -xvf nagios-plugins-2.2.1.tar.gz
```

Configurons Nagios

Pour exécuter le fichier de configuration Nagios, changeons le répertoire actuel en nagios-4.4.6.

```
cd nagios-4.4.6/
```

Et puis exécutons le fichier de configuration Nagios.

```
./configure --with-command-group=nagcmd
```

Sortir :

```
*** Configuration summary for nagios 4.4.6 2020-04-28 ***:

 General Options:
 -------------------------
        Nagios executable:  nagios
        Nagios user/group:  nagios,nagios
       Command user/group:  nagios,nagcmd
            Event Broker:  yes
        Install ${prefix}:  /usr/local/nagios
    Install ${includedir}:  /usr/local/nagios/include/nagios
```

```
             Lock file:  /run/nagios.lock
  Check result directory:  /usr/local/nagios/var/spool/checkresults
         Init directory:  /lib/systemd/system
 Apache conf.d directory:  /etc/httpd/conf.d
          Mail program:  /bin/mail
              Host OS:  linux-gnu
        IOBroker Method:  epoll

 Web Interface Options:
 ------------------------
             HTML URL:  http://localhost/nagios/
              CGI URL:  http://localhost/nagios/cgi-bin/
 Traceroute (used by WAP):

Review the options above for accuracy.  If they look okay,
type 'make all' to compile the main progra
```

Après avoir configuré, compilons et installons les bibliothèques requises à l'aide de la commande **make**

```
make all
make install
```

Sortir :

```
*** Compile finished ***

If the main program and CGIs compiled without any errors, you
can continue with testing or installing Nagios as follows (type
'make' without any arguments for a list of all possible options):
  make test
     - This runs the test suite
```

```
make install
   - This installs the main program, CGIs, and HTML files

make install-init
   - This installs the init script in /lib/systemd/system

make install-daemoninit
   - This will initialize the init script
     in /lib/systemd/system

make install-groups-users
   - This adds the users and groups if they do not exist

make install-commandmode

  - This installs and configures permissions on the
     directory for holding the external command file

make install-config
   - This installs *SAMPLE* config files in /usr/local/nagios/etc
     You'll have to modify these sample files before you can
     use Nagios.  Read the HTML documentation for more info
     on doing this.  Pay particular attention to the docs on
     object configuration files, as they determine what/how
     things get monitored!

make install-webconf
   - This installs the Apache config file for the Nagios
     web interface
```

```
make install-exfoliation
   - This installs the Exfoliation theme for the Nagios
     web interface

make install-classicui
   - This installs the classic theme for the Nagios
     web interface


.
.
.
```

Dans la sortie ci-dessus, nous pouvons voir qu'il répertorie certaines bibliothèques à installer. Nous pouvons les installer à l'aide de la commande **make install**

par exemple,

```
make install-init
make install-commandmode
make install-config
```

Installons et configurons l'interface Web pour Nagios

Exécutons la commande suivante pour installer l'interface Web pour Nagios.

```
make install-webconf
```

Définissons un mot de passe pour l'interface Web.

```
htpasswd -s -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

Une fois le mot de passe configuré, redémarrons le service Apache.

```
systemctl start httpd.service
```

Ouvrons maintenant le port 80 dans le pare-feu

```
firewall-cmd --zone=public --add-port=80/tcp --permanent
```

```
firewall-cmd --reload
```

Compilons et installons le plugin Nagios libreires

Pour installer le plugin Nagios, changeons le répertoire actuel vers le répertoire du plugin Nagios.

```
cd /root/nagios/nagios-plugins-2.2.1/
```

Et exécutons le fichier de configuration du plugin Nagios

```
./configure --with-nagios-user=nagios --with-nagios-group=nagios
```

Ensuite, compilons et installons les bibliothèques requises

```
make all
make install
```

Commande pour vérifier les fichiers de configuration Nagios

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

Sortir :

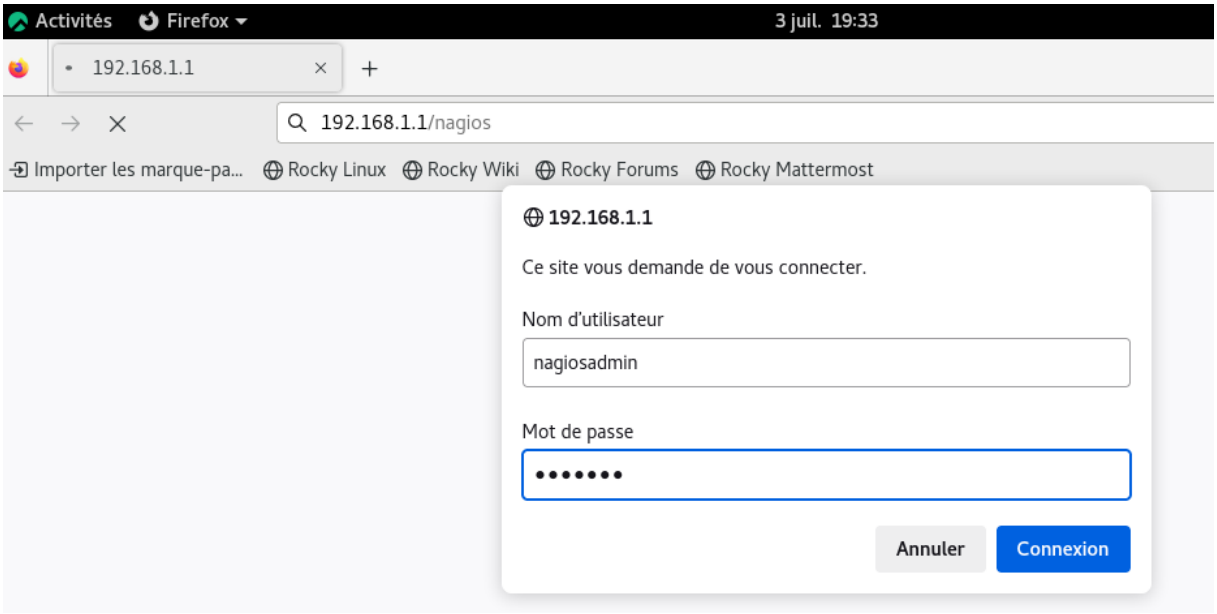
```
Nagios Core 4.4.6
Copyright (c) 2009-present Nagios Core Development Team and Community
Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2020-04-28
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
```

```
        Checked 8 services.
        Checked 1 hosts.
        Checked 1 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 1 hosts
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight chec
k
```

Activons les services Nagios

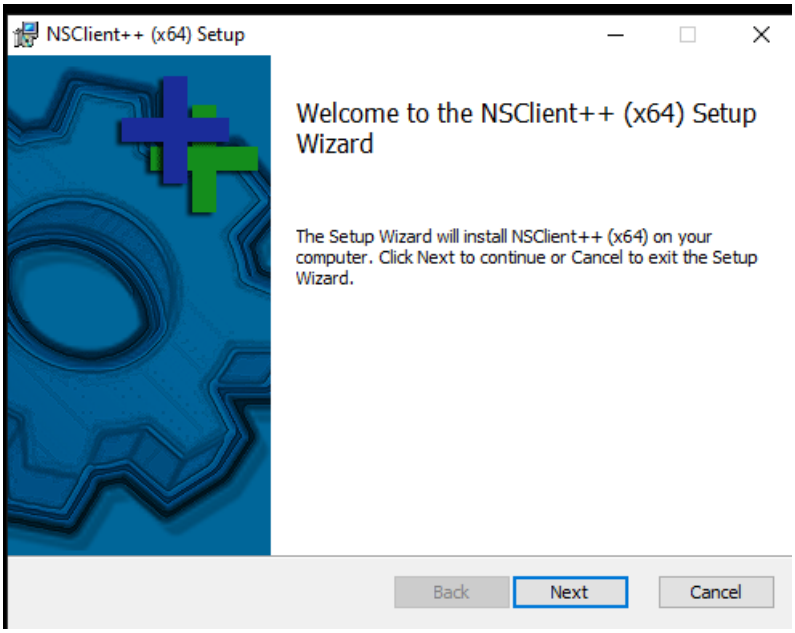
```
systemctl enable nagios
systemctl enable httpd
```

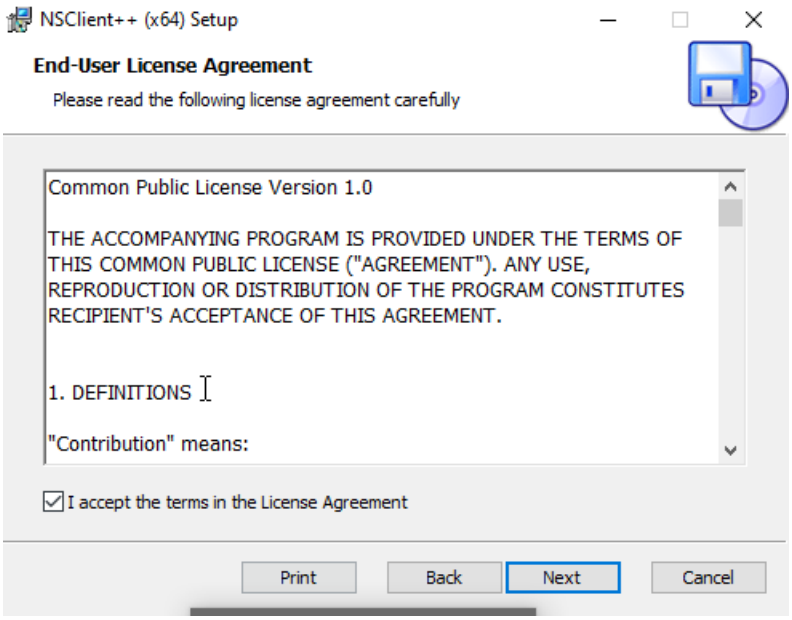
Connectons-nous à l'interface Web de Nagios

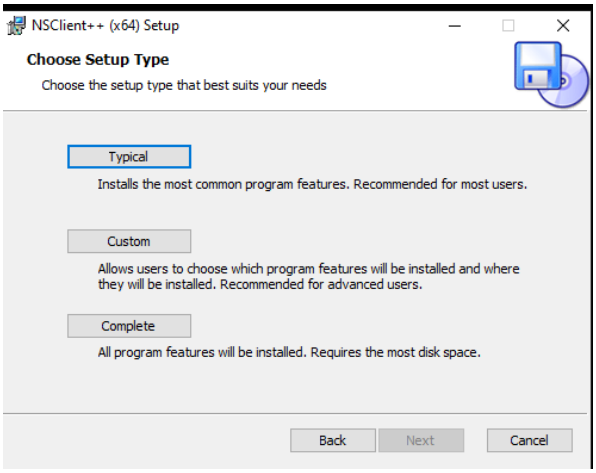


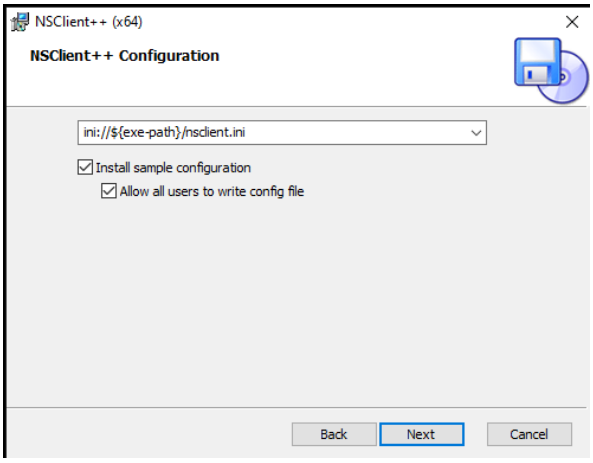
Ajout d'une machine cliente windows 10 grace à NSClient++

- **Installation de NSClient++ sous windows 10**

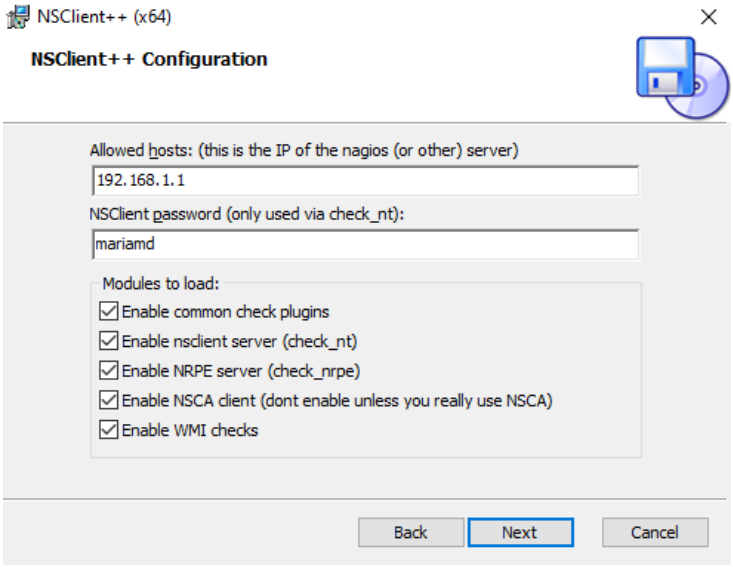








- **ajoutons l'adresse ip de la machine de Nagios et créons un mot de passe**



Renommons le fichier **windows.cfg** en **mariamd.cfg**

NB : mariamd c'est le nom de la machine cliente

```
[root@10 objects]# cp windows.cfg mariamd.cfg
[root@10 objects]# vim mariamd.cfg
```

Mettons le nom de la **machine cliente à la ligne 24** et **son adresse à la ligne 26**

```
23      use                     windows-server
    from a template
24      host_name               mariamd
    this host
25      alias                   My Windows Server
   ed with the host
26      address                 192.168.1.2
27  }
```

A la ligne **255** on va mettre **le mot de passe (mariamd)** qu'on a créé durant l'installation du NSClent++ dans le fichier **commands.cfg**

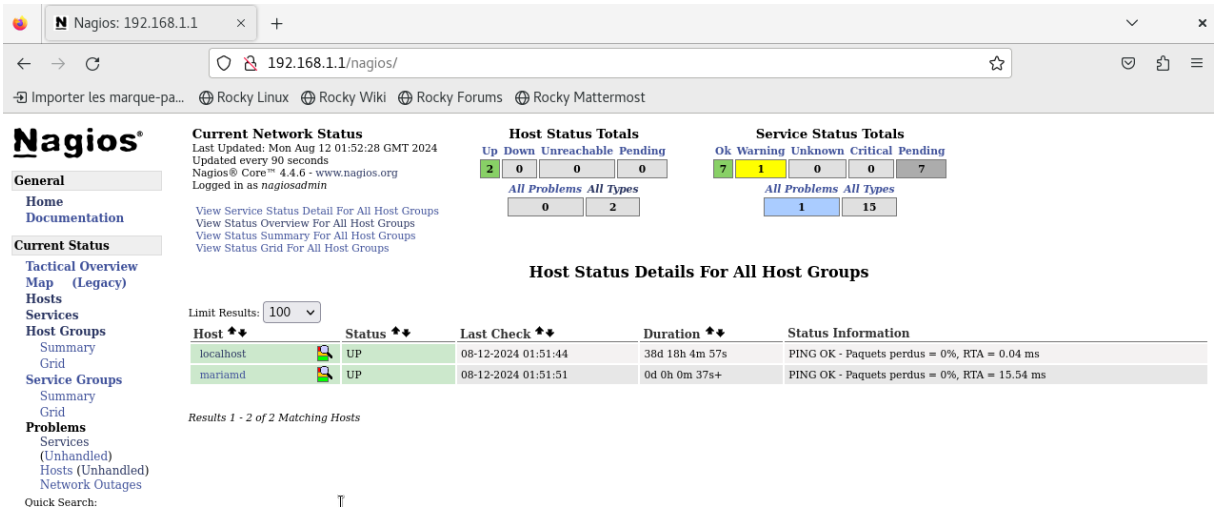
```
[root@10 objects]# vim commands.cfg
```

```
225     command_line    $USER1$/check_nt -H $HOSTADDRESS$ -p 12489 -v $ARG1$ $AR
    G2$ -s "mariamd"
```

Redémarrons le service nagios

```
[root@10 objects]# systemctl restart nagios
```

Et notre machine cliente windows 10 est ajoutée dans Nagios



❖ Notification par email

Installation des packages nécessaires

```
[root@server1 ~]# yum install postfix cyrus-sasl-plain mailx -y
```

Configuration de postfix pour Gmail, activer STARTTLS Encryption

```
sed -i 's/smtp_tls_security_level = may/smtp_tls_security_level = encrypt/'
/etc/postfix/main.cf
```

```
echo "smtp_tls_security_level = encrypt" >> /etc/postfix/main.cf
```


```
echo "smtp_tls_security_level = encrypt" >> /etc/postfix/main.cf
```


```
echo "smtp_tls_CAfile = /etc/pki/tls/certs/ca-bundle.crt" >> /etc/postfix/main.cf
```


```
echo "smtp_tls_CAfile = /etc/pki/tls/certs/ca-bundle.crt" >> /etc/postfix/main.cf
```

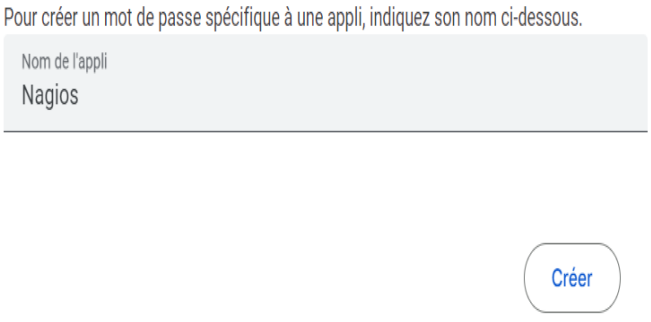

Définissons l'agent relai Gmail et SAS

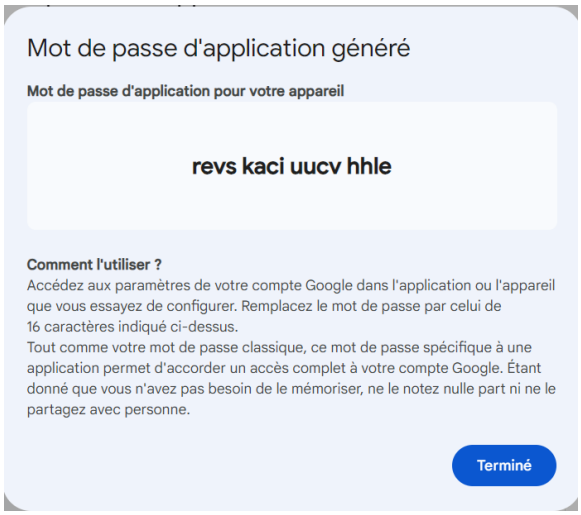
```
cat >> /etc/postfix/main.cf << EOF
relayhost = [smtp.gmail.com]:587
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_sasl_security_options = noanonymous
EOF
```

```
vim /usr/local/nagios/etc/objects/contacts.cfg
```

```
    email                   mdiack700@gmail.com ; <<*
ALL ADDRESS ******
```

Configurons SASL pour le Compte gmail avec mot de passe Application





```
vim /etc/postfix/sasl_passwd
```

```
[smtp.gmail.com]:587 mdiack700@gmail.com:nalhgmzissagnmna
```

Générons un postfix lookup

```
postmap /etc/postfix/sasl_passwd
```

Changeons les permissions

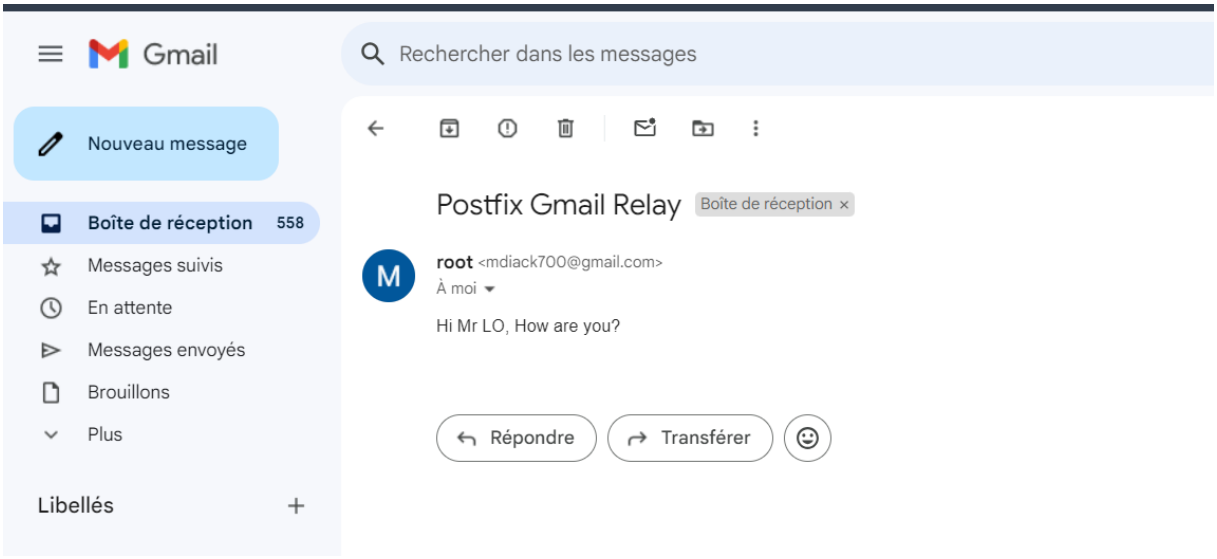
```
chown root:root /etc/postfix/sasl_passwd*

chmod 600 /etc/postfix/sasl_passwd*
```

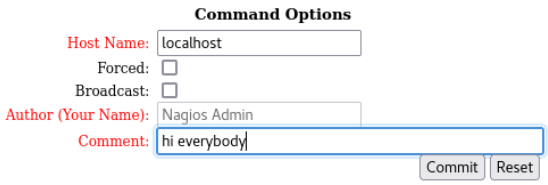
Activons postfix au démarrage

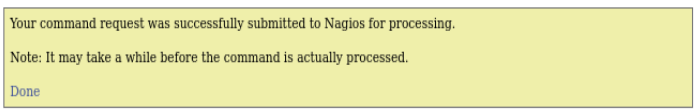
```
systemctl enable postfix --now
```

Testons l'envoi de mail sur gmail

```
echo "Hi Mr LO, How are you?" | mail -s "Postfix Gmail Relay" mdiack700@gmail.com
```







** CUSTOM Service Alert: localhost/HTTP is WARNING ** Boîte de réception ×

mdiack700@gmail.com
À moi

***** Nagios *****

Notification Type: CUSTOM

Service: HTTP
Host: localhost
Address: 127.0.0.1
State: WARNING

Date/Time: Sun Aug 11 03:02:14 GMT 2024

Additional Info:

HTTP WARNING: HTTP/1.1 403 Forbidden - 7897 octets en 0,001 secondes de temps de réponse
…

** CUSTOM Service Alert: localhost/SSH is OK ** Boîte de réception ×

mdiack700@gmail.com
À moi

***** Nagios *****

Notification Type: CUSTOM

Service: SSH
Host: localhost
Address: 127.0.0.1
State: OK

Date/Time: Sun Aug 11 03:04:16 GMT 2024

Additional Info:

SSH OK - OpenSSH_8.0 (protocol 2.0)

V. Conclusion

En conclusion, Nagios est un outil essentiel pour la surveillance des infrastructures informatiques, offrant des fonctionnalités flexibles et extensibles qui permettent de maintenir la disponibilité des services critiques. Son adoption permet aux entreprises de détecter et de prévenir les pannes, renforçant ainsi la fiabilité et la performance des systèmes. C'est un atout majeur pour toute équipe informatique cherchant à garantir la continuité des services et la satisfaction des utilisateurs.