|  |  |
| --- | --- |
| Une image contenant texte, clipart  Description générée automatiquement | **Sites Web sécurisés (HTTPS)** |

## Installation

Pour installer Apache seulement, on installe simplement le paquet nginx …  
**>> sudo apt update && sudo apt upgrade -y && sudo apt install nginx**

À la suite de cette installation le serveur doit fonctionner et être accessible à l'adresse …  
**http://localhost** (à partir du même hôte)  
ou  
**http://<Adresse IP>**

La page par défaut s'afficher dans le fureteur (*browser*).   
Il s'agit du **contenu du fichier /var/www/html/index.html** qui est affiché par défaut.

## Lancement

nginx se lance automatiquement dès son installation, et se relance automatiquement à chaque démarrage.   
C'est l'idéal pour un serveur qui doit continuellement fournir du contenu en ligne, mais pour un serveur de test (on dit de développement) on peut éventuellement désirer un comportement différent.

Pour empêcher nginx de démarrer automatiquement …  
**>> sudo systemctl disable nginx.service**

Pour réactiver le démarrage automatique …  
**>> sudo systemctl enable nginx.service**

### Commande nginx

nginx propose une série de commande afin de gérer son propre démon (*daemon*).  
Voici les commandes de base …

Pour afficher la version de nginx utilisée …  
**>> sudo nginx -v  
nginx version: nginx/1.18.0**

Pour tester l'ensemble de la configuration de nginx …  
**>> sudo nginx -t  
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  
nginx: configuration file /etc/nginx/nginx.conf test is successful**

Pour le démarrage de nginx, on peut utiliser la commande systemctl comme la plupart des services.

Une fois nginx démarré, il peut être contrôlé en appelant l'exécutable nginx avec le paramètre -s …  
La syntaxe de la commande est la suivante …  
**>> sudo nginx -s <signal>**

Où le signal peut être l'un des suivants …

* **stop**  
  Arrêt rapide ;
* **quit**  
  Arrêt en douceur ;
* **reload**  
  Rechargement du fichier de configuration ;
* **reopen**  
  Réouvrir les fichiers journaux

Par exemple, pour arrêter les processus nginx en attendant que les processus de travail aient fini de traiter les demandes en cours …  
**>> sudo nginx -s quit**

## Mise en place d’un site web sécurisé

Les **fichiers qui seront livré par le serveur** se situent dans le répertoire **sous /var/www/html**.

### Création du certificat et des clés

On crée le répertoire /etc/ssl/private, répertoire qui va contenir le fichier de clé privée …  
**>> sudo mkdir --verbose /etc/nginx/certs  
mkdir: création du répertoire '/etc/nginx/certs'  
>> sudo chmod --verbose 700 /etc/nginx/certs  
Le mode de '/etc/nginx/certs' a été modifié de 0755 (rwxr-xr-x) en 0700 (rwx------)**

La prochaine est la création d’une paire de clé et d’un certificat auto-signé avec OpenSSL en une seule commande …  
**>> sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/nginx/certs/nginx.key   
 -out /etc/nginx/certs/nginx.crt**

openssl propose une série de questions afin de personnaliser le certificat.

On répond à ces questions de manière appropriée.   
La ligne la plus importante est celle qui demande le nom commun (par exemple, le FQDN du serveur ou son nom).

**Country Name (2 letter code) [AU]:CA  
State or Province Name (full name) [Some-State]:Quebec  
Locality Name (eg, city) []:Montreal  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Web-Maisonneuve  
Organizational Unit Name (eg, section) []:Formation  
Common Name (e.g. server FQDN or YOUR name) []:www.monsite.info  
Email Address []:webmaster@monsite.info**

Les deux fichiers qui ont été créés seront placés dans le répertoire /etc/ngginx/certs.

**Remarque** …  
Pour vérifier le contenu du certificat généré …  
**>> sudo** **openssl x509 -noout -text -in /etc/nginx/certs/nginx.crt**

### Secret de transmission parfait

Lors de l’utilisation d’OpenSSL, il est intéressant de créer également un groupe Diffie-Hellman (Merkel) fort.

On peut créer ce secret de transmission parfait …  
**>> sudo openssl dhparam -out /etc/nginx/certs/dhparam.pem 2048**

L’exécution de cette commande peut prendre quelques minutes, mais une fois terminé, on aura un groupe DH puissant dans /etc/nginx/certs/dhparam.pem qu’il sera possible d’utiliser dans la configuration du site Web …  
**>> sudo ls -l /etc/nginx/certs  
total 12  
-rw-r--r-- 1 root root 424 13 nov 09:18 dhparam.pem  
-rw------- 1 root root 1704 12 nov 16:06 monsite.info.key  
-rw-r--r-- 1 root root 1566 12 nov 16:08 monsite.info.pem**

### Modification du fichier index.html

Afin de créer un site personnalisé, on remplace le fichier /var/www/html/index.html.

**Remarque** …  
Si ce fichier existe, on peut en créer une sauvegarde …  
**>> sudo mv /var/www/html/index.html /var/www/html/index.html.original**

On crée la page d’accueil du site …  
**>> sudo nano /var/www/html/index.html**

On y ajoute le contenu suivant …  
**<!DOCTYPE html>**

**<html>**

**<head>**

**<!-- En-tête de la page -->**

**<meta charset="utf-8" />**

**<title>En construction</title>**

**</head>**

**<body>**

**<!-- Corps de la page -->**

**Site en construction**

**</body>**

**</html>**

**Remarque** …  
Il est aussi possible de copier dans le répertoire /var/www/monsite.info un site Web existant.

## Configuration pour l’utilisation de SSL

La prochaine étape est la création d’un nouveau fichier dans le répertoire /etc/nginx/sites-available/ afin de configurer un bloc server qui répondra aux requêtes HTTPS à l'aide des fichiers de certificat préalablement cré.   
Il sera ensuite possible de configurer ce bloc de serveur par défaut afin de rediriger les requêtes HTTP vers HTTPS.

### Création du bloc server TLS/SSL

On doit informer nginx de l’existence du site Web, de l’utilisation de SSL/TLS et de la manière de le servir.  
**>> cd /etc/nginx/**

C'est dans ce répertoire que se trouvent les fichiers de configuration nginx.

Les deux répertoires qui intéressant à ce stade sont sites-available et sites-enabled …

* **sites-available**  
  contient des fichiers de configuration individuels pour tous les sites Web statiques possibles ;
* **sites-enabled**   
  contient des liens vers les fichiers de configuration que nginx lira et exécutera réellement.

La prochaine étape est de modifier le fichier de configuration existant nommé **default** ou de **créer un fichier de configuration dans le répertoire sites-available** et de **créer un lien symbolique (un pointeur) de ce fichier vers le répertoire sites-enabled** afin d’indiquer à nginx que ce site est activé.

**server {**

**listen 80 default\_server;**

**listen [::]:80 default\_server;  
 server\_name \_;  
 return 301 https://$host$request\_uri;**

**}**

**server {  
 listen 443 http2 ssl;  
 listen [::]:443 http2 ssl;  
 root /var/www/html;  
 # Add index.php to the list if you are using PHP  
 index index.html index.htm index.nginx-debian.html;  
 server\_name \_;**

**ssl\_certificate /etc/nginx/certs/nginx.crt;  
 ssl\_certificate\_key /etc/nginx/certs/nginx.key;  
 ssl\_dhparam /etc/nginx/certs/dhparam.pem;**

**location / {  
 # First attempt to serve request as file, then  
 # as directory, then fall back to displaying a 404.  
 try\_files $uri $uri/ =404;  
 }  
}**

### Création d’une redirection de HTTP vers HTTPS (optionel)

Afin de s’assurer que nginx réponde en tout temps sur le port TCP 443 (même si la requête est adressée au port TCP 80), on ajoute au premier bloc server la ligne suivante …  
**return 301 https://$host$request\_uri/;**

Cette directive configure le bloc server HTTP sur le port 80 (par défaut) afin qu’il redirige les demandes entrantes vers le bloc de serveur HTTPS.

### Publication du site Web

Maintenant que le fichier de configuration du site est créé, on doit créer un lien symbolique vers le   
répertoire sites-enabled pour indiquer à nginx de l'activer.

La syntaxe est la suivante …  
**>> ln -s <Fichier source> <fichier destination>**

Dans cet exemple, la syntaxe ressemble à …  
**>> sudo ln -s /etc/nginx/sites-available/default /etc/nginx/sites-enabled/default**

### Activation des modifications

Après avoir créé ou modifier un ou plusieurs fichiers de configuration, on doit redémarrer le service nginx afin qu’il tienne compte des changements.

Toutefois, au préalable, il est possible de vérifier que les modifications apportées ne comportent pas des erreurs de syntaxe …  
**>> sudo nano nginx -t  
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  
nginx: configuration file /etc/nginx/nginx.conf test is successful**

Et on redémarre le service nginx …  
**>> sudo systemctl restart nginx.service**

Le processus Nginx est redémarré, mettant en œuvre les paramètres SSL nouvellement configurés.

**Remarque** …  
On peut également utiliser la commande suivante …  
**>> sudo nginx -s restart**

## Chiffrement fort

Afin de s’assurer que nginx n’utilise que des algorithmes identifiés comme forts, il est possible de le prpéciser dans le fichier de configuration /etc/nginx/sites-available/default.

Pour que le serveur Nginx n'accepte que les protocoles TLS 1.2 et TLS 1.3, protocoles identifiés comme forts et qu’il n’accepte que des algorithmes de chiffrement forts …  
**>> sudo nano /etc/nginx/sites-available/default**

On ajoute les lignes suivantes …  
**server {  
 listen 443 http2 ssl;  
 listen [::]:443 http2 ssl;  
 root /var/www/html;  
…  
 ssl\_certificate /etc/nginx/certs/nginx.crt;  
 ssl\_certificate\_key /etc/nginx/certs/nginx.key;  
 ssl\_dhparam /etc/nginx/certs/dhparam.pem;  
  
 ssl\_protocols TLSv1.2 TLSv1.3;  
 ssl\_prefer\_server\_ciphers off;  
 ssl\_ciphers "ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM- SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20- POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES128-SHA";  
  
 location / {  
…  
 }  
}**

On doit redémarrer le service nginx …  
**>> sudo systemctl restart nginx.service**

### Verification du bon fonctionnement

On peut vérifier si le site répond positivement ou négativement aux requêtes utilisant protocoles et algorithmes forts.

Depuis un hôte distant …  
**>> openssl s\_client -tls1\_1 -connect <Adresse ou nom DNS du serveur>:443 2> /dev/null | grep -i -E "cipher|protocol"  
>> openssl s\_client -tls1\_1 -connect 192.168.1.100:443 2> /dev/null | grep -i -E "cipher|protocol"  
New, (NONE), Cipher is (NONE)  
 Protocol : TLSv1.1  
 Cipher : 0000**La sortie de la commande indique que le serveur n’accepte pas le protocole TLSv1.1.

Depuis un hôte distant …  
**>> openssl s\_client -tls1\_2 -connect <Adresse ou nom DNS du serveur>:443 2> /dev/null | grep -i -E "cipher|protocol"  
>> openssl s\_client -tls1\_2 -connect 192.168.1.100:443 2> /dev/null | grep -i -E "cipher|protocol"  
New, TLSv1.2, Cipher is ECDHE-RSA-AES256-GCM-SHA384  
 Protocol : TLSv1.2  
 Cipher : ECDHE-RSA-AES256-GCM-SHA384**   
La sortie de la commande indique que le serveur n’accepte pas le protocole TLSv1.1.

Depuis un hôte distant …  
**>> openssl s\_client -tls1\_3 -connect <Adresse ou nom DNS du serveur>:443 2> /dev/null | grep -i -E "cipher|protocol"  
>> openssl s\_client -tls1\_3 -connect 192.168.1.100:443 2> /dev/null | grep -i -E "cipher|protocol"  
New, TLSv1.3, Cipher is TLS\_AES\_256\_GCM\_SHA384**

On peut constater l’activation de l'utilisation de TLS 1.2 et TLS 1.3 sur le serveur Nginx.

## Autres fonctionnalités

### HTTP/2

Pour utiliser le protocole http/2 sur un serveur nginx, dans la zone HTTPS, on ajoute HTTP2 à l'élément de configuration LISTEN …  
**>> sudo nano /etc/nginx/sites-available/default**

On modifie les lignes suivantes …  
**…  
listen 443 ssl http2;  
listen [::]:443 ssl http2;**

On doit redémarrer le service nginx …  
**>> sudo systemctl restart nginx.service**

**Remarque** …  
Il est important de s’assurer que l’on dispose d’une site HTTPS configuré sur le serveur nginx.   
HTTPS est une exigence pour HTTP2.

### Vérification de l’utilisation de HTTP/2

Il est possible de vérifier si le serveur peut travailler avec le protocole HTTP/2.

Depuis un hôte distant …  
**>> curl -I https://<Adresse IP ou Nom DNS>  
>> curl -I https://192.168.1.100  
HTTP/2 200  
server: nginx/1.18.0 (Debian)  
…**

La première ligne affichée confirme l'utilisation du protocole HTTP2.

### HTTP Strict Transport Security (HSTS)

Une politique HSTS est publiée en envoyant l'en-tête de réponse HTTP suivant à partir de sites Web sécurisés (HTTPS) …  
**Strict-Transport-Security: max-age=31536000**

Pour activer HSTS sur un serveur ngonx…  
**>> sudo nano /etc/nginx/sites-available/default**

On ajoute les lignes suivantes …

**server {  
 listen 443 ssl default\_server;  
 listen [::]:443 ssl default\_server;  
...  
 add\_header Strict-Transport-Security "max-age=31536000; ncludeSubDomains";  
...**

On doit redémarrer le service nginx …  
**>> sudo systemctl restart nginx.service**

**Remarque** …  
Il est important de s’assurer que l’on dispose d’une site HTTPS configuré sur le serveur nginx.   
HTTPS est une exigence pour HTTP2.

### Vérification de l’utilisation de HSTS

Il est possible de vérifier si le serveur peut travailler avec le protocole HSTS.

Depuis un hôte distant …  
**>> curl -I https://<Adresse IP ou Nom DNS>  
>> curl -I https://192.168.1.100  
HTTP/2 200  
server: nginx/1.18.0 (Debian)  
…  
Strict-Transport-Security: max-age=31557600  
…**