TUTOS-INFO
L'informatique par l'exemple

TUTORIEL

APACHE 2.4

Installer et configurer Apache 2

APACHE
HTTP SERVER

SOMMAIRE

1. QU'EST-CE QUE APACHE ?
2. INSTALLATION D'APACHE 2.4 SUR UNE MACHINE DEBIAN
3. CONFIGURER SSH SUR DEBIAN 11
4. CONFIGURATION DE L'ACCES HTTP AU SITE WEB
5. CONFIGURATION DE L'ACCES HTTPS AU SITE WEB
6. INSTALLER ET CONFIGURER OPEN SSL
7. REDIRECTION DES REQUETES HTTP VERS HTTPS
8. LES COMMANDES UTILES

EXPERT
DIFFICULTE

1 – QU'EST-CE QUE APACHE 2.4 ?

Apache est un logiciel de serveur web gratuit et open-source qui est utilisé par environ 45 % des sites web à travers le monde. Le nom officiel est SERVEUR APACHE HTTP et il est maintenu et développé par Apache Software Foundation.

Il permet aux propriétaires de sites web de servir du contenu sur le web. Apache est l'un des serveurs web les plus anciens et les plus fiables avec une première version sortie en 1995.

Lorsqu'un internaute souhaite visiter un site web, il saisit un nom de domaine dans la barre d'adresse de son navigateur et le serveur web fournit les fichiers demandés en agissant comme un livreur virtuel.

Bien que nous appelions Apache un serveur web, ce n'est pas un serveur physique mais plutôt un logiciel qui s'exécute sur un serveur. Son travail consiste à établir une connexion entre un serveur et les navigateurs des visiteurs du site web (Firefox, Google Chrome, Safari, etc.) tout en délivrant des fichiers entre eux (structure client-serveur). Apache est un logiciel multiplateforme, il fonctionne donc à la fois sur les serveurs Unix et Windows.

Lorsqu'un visiteur souhaite charger une page sur votre site web, par exemple, la page d'accueil ou votre « A propos de nous », son navigateur envoie une requête à votre serveur et Apache renvoie une réponse avec tous les fichiers demandés (texte, images, etc.). Le serveur et le client communiquent via le protocole http et Apache est responsable de la communication fluide et sécurisée entre les deux machines.

Apache est hautement personnalisable, car il a une structure basée sur des modules. Les modules permettent aux administrateurs de serveur d'activer ou de désactiver des fonctionnalités supplémentaires. Apache possède des modules pour la sécurité, la mise en cache, la réécriture d'URL, l'authentification par mot de passe et encore plus. Vous pouvez également configurer vos propres configurations du serveur via un fichier appelé « .htaccess », qui est un fichier de configuration Apache.

APACHE ET LA CONCURRENCE

NGINX, prononcez « Engine-X », est une application récente de serveur web, lancée en 2004. A ce jour, elle a acquis une certaine popularité auprès des propriétaires de sites web. Nginx a été créé pour résoudre le problème appelé **c10k**, ce qui signifie qu'un serveur web utilisant des fils pour gérer les demandes des utilisateurs ne peut pas gérer plus de 10 000 connexions simultanément.

1. Etant donné qu'Apache utilise la structure basée sur les fils, les propriétaires de sites web avec un trafic élevé peuvent rencontrer des problèmes de performances. Nginx est l'un des serveurs web qui traitent le problème de c10k et probablement le plus réussi.

2. Nginx possède une architecture pilotée par les événements qui ne crée pas de nouveau processus pour chaque requête. Au lieu de cela, il gère chaque demande entrante dans un seul fil. Ce processus maître gère plusieurs processus de travail qui effectuent le traitement réel des demandes. Ce modèle de Nginx répartit les requêtes des utilisateurs entre les processus de travail de manière efficace, conduisant ainsi à une meilleure évolutivité.

3. Si vous avez besoin de gérer un site web avec un trafic élevé, Nginx est un excellent choix, car il peut le faire en utilisant un minimum de ressources. Ce n'est pas une coïncidence s'il est utilisé par de nombreux sites web à forte visibilité tels que Netflix, Hulu, Pinterest et Airbnb.

[2]**TOMCAT** est un serveur web également développé par Apache Software Foundation. Son nom officiel est APACHE TOMCAT.

C'est un serveur HTTP aussi mais il alimente les applications Java au lieu des sites web statiques. Tomcat peut exécuter différentes spécifications Java telles que Java Servlet, JavaServer Pages (JSP), Java EL et WebSocket.

1. Tomcat a été créé spécifiquement pour les applications Java, alors qu'Apache est un serveur HTTP à usage général. Vous pouvez utiliser Apache avec différents langages de programmation (PHP, Python, Perl, etc.)) l'aide du module Apache approprié (mod_php, mod_python, mod_perl, etc.).

2. Bien que vous puissiez utiliser un serveur Tomcat pour servir également des pages web statiques, il est moins efficace qu'Apache. Par exemple, Tomcat pré-charge la machine virtuelle Java et les autres bibliothèques liées à Java dont vous n'auriez pas besoin sur la plupart des sites web.

3. Tomcat est également moins configurable que les autres serveurs web. Par exemple, pour mettre en marche Wordpress, le meilleur choix est un serveur HTTP à usage générale tel qu'Apache ou NGINX.

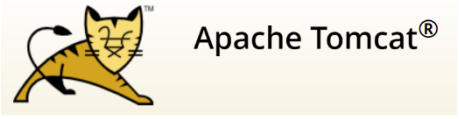
AVANTAGES ET INCONVENIENTS D'APACHE

Avantages :

1. Open-source et gratuit même pour un usage commercial.
2. Logiciel fiable et stable.
3. Mise à jour régulière, correctifs de sécurité réguliers.
4. Flexible grâce à sa structure basée sur des modules.
5. Facile à configurer, adapté aux débutants.
6. Plateforme-Cross (fonctionne sur les serveurs Unix et Windows).
7. Fonctionne avec les sites WordPress.
8. Grande communauté et support disponible en cas de problème.

Inconvénients :

1. Problèmes de performances sur les sites web avec un énorme trafic.
2. Trop d'options de configuration peuvent mener à la vulnérabilité de la sécurité.



2 – INSTALLATION D'APACHE 2.4 SUR UNE MACHINE DEBIAN 11

Pour ce tutoriel, nous partons d'une machine virtuelle Debian 11.3 fraîchement installée. La procédure d'installation du serveur web Apache 2.4 est assez simple et s'effectue à partir des commandes suivantes :

- Ouvrez une session en tant que « root »
- Vérifiez si de nouveaux « paquets » peuvent être mis à jour avec la commande suivante :

apt install update

```
root@debian:~# apt update
```

Si vous avez des paquets à mettre à jour, Debian vous affiche un message avec le nombre de paquets à mettre à jour :

```
10 paquets peuvent être mis à jour.
```

Dans ce cas, saisissez la commande suivante :

apt install upgrade

```
root@debian:~# apt upgrade
```

- Installez Apache 2.4 en saisissant la commande suivante :

apt install apache2

Faites « Entrée » pour lancer l'installation du paquet et patientez pendant son installation :

```
root@debian:~# apt install apache2
```

VERIFICATION DU STATUT DU SERVEUR APACHE

Une fois Apache installé, vous pouvez vérifier que le service est actif avec la commande suivante :

systemctl status apache2

Vous devez obtenir cet affichage avec le statut « active (running) » :

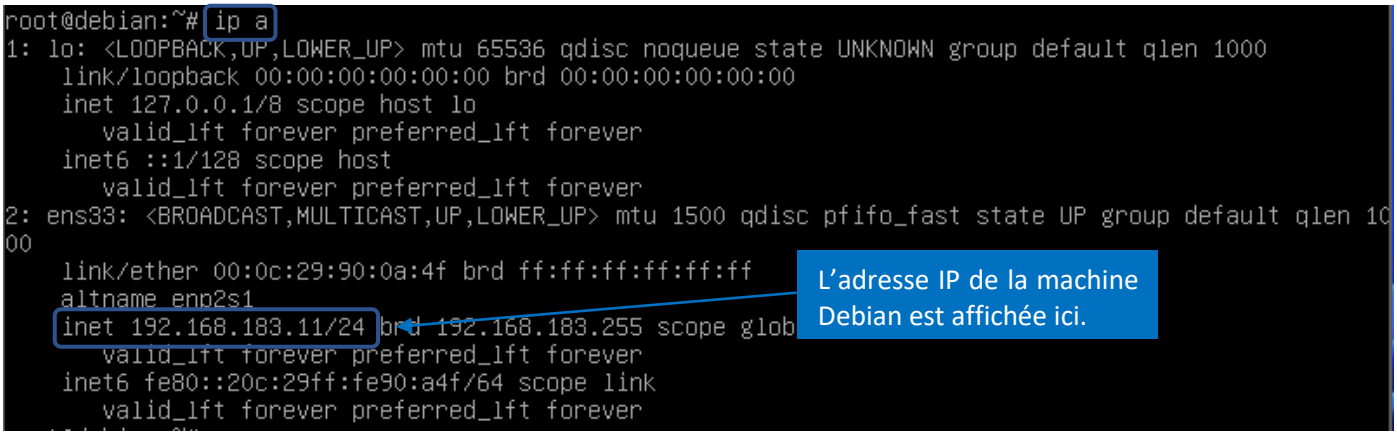
- Pressez les touches **CTRL + C** pour sortir de l'affichage du statut.

AFFICHAGE DE LA PAGE D'ACCUEIL D'APACHE

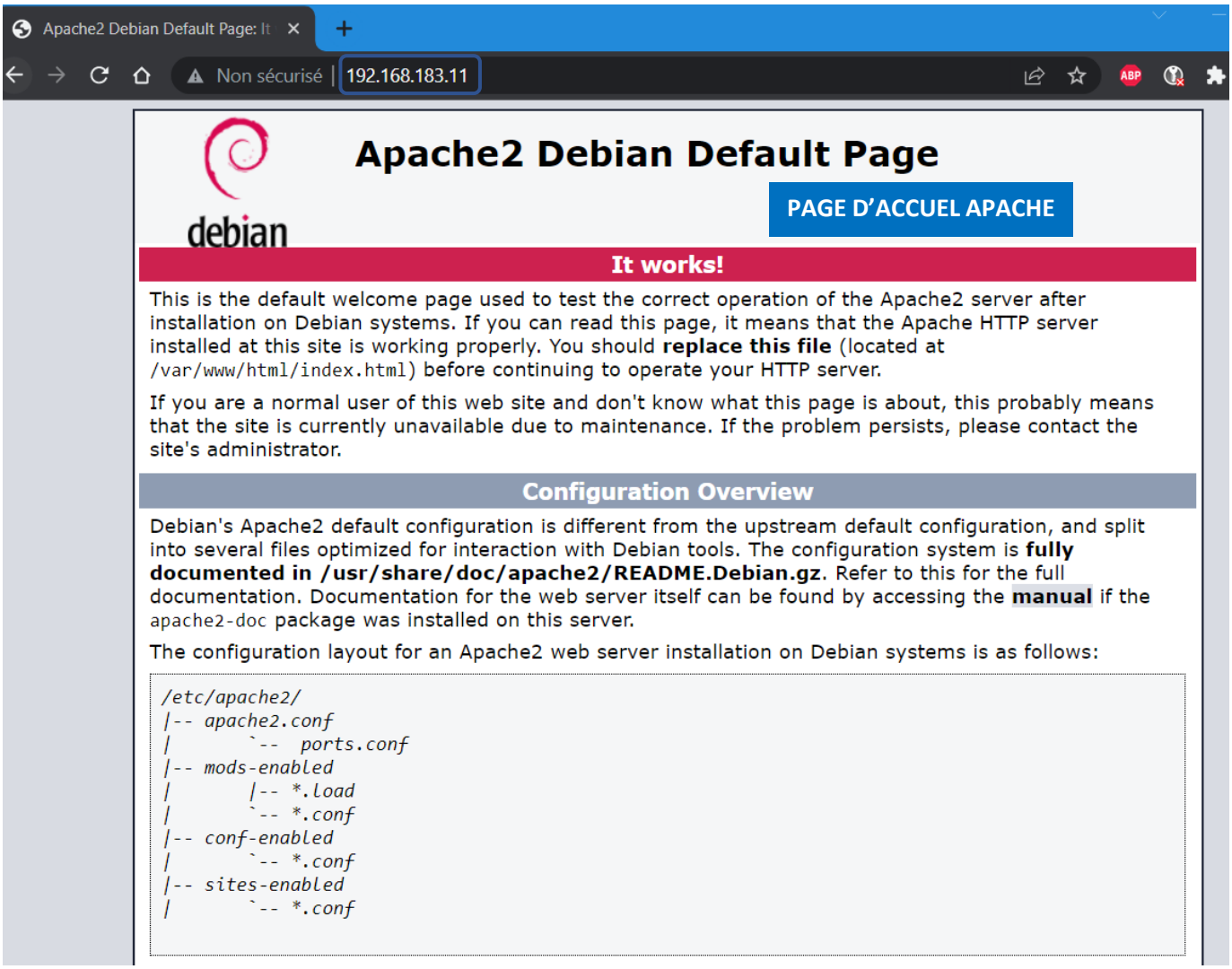
Une fois Apache installé (et que le statut est bien actif), vous pouvez afficher la page d'accueil par défaut en ouvrant un navigateur et en saisissant l'IP de votre machine Debian.

- Faites afficher l'IP de votre machine Debian en saisissant la commande suivante :

ip a



En saisissant cette adresse dans un navigateur on obtient l'affichage de la page par défaut d'Apache :



EMPLACEMENT DE LA PAGE D'ACCUEIL D'APACHE

Les fichiers et dossiers Apache sont situés à cet emplacement :

/var/www/html

- Saisissez la commande « ***cd /var/www/html*** » pour vous déplacer dans le dossier par défaut :

```
root@debian:/# cd /var/www/html
```

- Saisissez la commande « ***ls*** » : la page par défaut d'Apache est bien située dans « html » :

```
root@debian:/var/www/html# ls
index.html
```

DOSSIERS IMPORTANTS A CONNAITRE

Apache utilise différents dossiers pour son bon fonctionnement. Ces dossiers sont situés à cet emplacement :

/etc/apache2

- Déplacez-vous dans ce dossier et faites afficher les dossiers :

cd /etc/apache1
ls

```
root@debian:/# cd /etc/apache2
root@debian:/etc/apache2# ls
apache2.conf    conf-enabled  magic           mods-enabled  sites-available
conf-available  envvars       mods-available  ports.conf    sites-enabled
```

Les principaux fichiers et dossiers à connaître :

apache2.conf	La configuration d'Apache est effectuée en plaçant des directives dans ce fichier de configuration principal
sites-available	contient les fichiers de configuration des **sites disponibles**
sites-enabled	contient des liens symboliques vers les configurations des sites disponibles (dans sites-available) et indique les sites actifs (si plusieurs sites sont présents)
ports.conf	Indique les ports d'écoute du serveur web Apache (par défaut le port « 80 »).

CONTENU DU DOSSIER « SITES-ENABLED »

Ce dossier indique les sites « actifs » sur Apache. Si on ouvre le dossier, on constate qu'un site est actif par défaut :

```
root@debian:/etc/apache2# cd sites-enabled/
root@debian:/etc/apache2/sites-enabled# ls
000-default.conf
```

Ici, on constate que le site web actif par défaut correspond au fichier « **000-default.conf** ».

CONTENU DU DOSSIER « SITES-AVAILABLE »

On retrouve le fichier de configuration du site « 000-default.conf » dans le dossier des sites disponibles :

```
root@debian:/etc/apache2# cd sites-available
root@debian:/etc/apache2/sites-available# ls
000-default.conf  default-ssl.conf
```

Le fichier « 000-default.conf » correspond à la configuration du site web par défaut d'Apache. Ce site est automatiquement activé par défaut lors de l'installation d'Apache. On constate qu'il y a également un fichier « default-ssl.conf ». Ce fichier correspond à la configuration du site web par défaut en https. Attention, Apache n'active pas le https par défaut. Seul le http est activé dans un premier temps lors de l'installation (voir dossier « sites-enabled »).

Contenu du fichier « 000-default.conf » :

```
<VirtualHost *:80>
        # The ServerName directive sets the request
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further vi
        #ServerName www.example.com

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        # For most configuration files from conf-available/, which are
        # enabled or disabled at a global level, it is possible to
        # include a line for only one particular virtual host. For example the
        # following line enables the CGI configuration for this host only
        # after it has been globally disabled with "a2disconf".
        #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
```

Le serveur web Apache écoute par défaut sur le port « 80 ».

Emplacement par défaut de la page d'accueil du site.

Emplacement des fichiers de log et d'erreur.

SUPPRESSION DE LA SIGNATURE DU SERVEUR

Il peut être intéressant, pour des raisons de sécurité, de masquer l'affichage de la version Apache utilisée. Si on saisit une adresse erronée de type http://ip_apache/test, on obtient ce message avec les indications sur la version Apache, l'IP et le port d'écoute (en bas) :

- Saisissez nano /etc/apache2/conf-available/security.conf
- Modifier la valeur de **ServerTokens** OS en **ServerTokens** Prod et **ServerSignature** On en **ServerSignature** Off

```
# ServerTokens
# This directive configures what you return as the Server HTTP response
# Header. The default is 'Full' which sends information about the OS-Type
# and compiled in modules.
# Set to one of:  Full | OS | Minimal | Minor | Major | Prod
# where Full conveys the most information, and Prod the least.
#ServerTokens Minimal
ServerTokens Prod
#ServerTokens Full

#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
# listings, mod_status and mod_info output etc., but not CGI generated
# documents or custom error documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of:  On | Off | EMail
#ServerSignature Off
ServerSignature Off
```

- Enregistrez les modifications et relancez le service Apache « ***systemctl reload apache2*** » :

```
root@debian:/etc/apache2/conf-available# systemctl reload apache2
```

- Refaites un essai : les indications ont disparu :

Commandes à connaître :

Afficher le statut du serveur web Apache	**systemctl status apache2**
Redémarrer le serveur web Apache	**systemctl restart apache2**
Stopper le serveur web Apache	**systemctl stop apache2**
Recharger le service web Apache	**systemctl reload apache2**

Pour la réalisation de ce TP, nous avons besoin de connaître le **FQDN** (*nom complet*) de notre machine Debian. Pour cela, il suffit de saisir la commande : `hostname -f`

Le nom complet de notre machine Debian est ici : `debian.tutosio`

Les fichiers de base de notre serveur web Apache se trouvent ici : `/etc/apache2`

2 dossiers sont importants : `sites-available` `sites-enabled`

Le dossier « *sites-available* » contient les **sites DISPONIBLES** sur le serveur web Apache
Le dossier « *sites-enabled* » contient les **sites ACTIFS** sur le serveur web Apache

Les fichiers de log (accès au serveur web Apache et erreurs de log) sont ici : `/var/log/apache2`

Le fichier « *access.log* » affiche les connexions au serveur web Apache et le fichier « *error.log* » affiche les erreurs de connexion au serveur web Apache `access.log` `error.log`

3 – CONFIGURATION DE L'ACCES SSH SUR LA MACHINE DEBIAN

Configuration du « hosts » sur la machine Debian et sur la machine Windows

Pour la réalisation de ce TP, nous devons configurer les fichiers « ***hosts*** » des machines Debian et Windows car nous n'avons pas de domaine hébergé chez un fournisseur (nous restons en « local »). **Pour cela nous devons modifier les fichiers « hosts »** afin de simuler un hébergement de domaine. Nous procédons ainsi :

Sur la machine Debian : `nano /etc/hosts`

☑ On ajoute la ligne correspondant à notre IP Debian et notre FQDN :

```
127.0.0.1       localhost
127.0.1.1       debian.tutosio  debian

192.168.4.108   debian.tutosio
```

Attention, vous devrez adapter à votre configuration IP et au nom FQDN de votre serveur Debian !

Sur la machine Windows :

☑ Ouvrez le Powershell en « tant qu'administrateur » et accédez au fichier « hosts » de Windows ainsi :

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. Tous droits réservés.

PS C:\Windows\system32> cd drivers/etc
PS C:\Windows\system32\drivers\etc> notepad hosts
```

☑ Dans Notepad, ajoutez, à la fin, l'IP de votre machine Debian et son FQDN (vous adapterez l'IP et le FQDN) :

```
192.168.4.108    debian.tutosio
```

☑ Enregistrez le fichier « hosts » et fermez Notepad
☑ Videz le cache DNS en saisissant la commande « ***ipconfig /flushdns*** » :

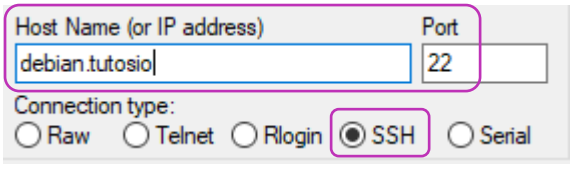
```
PS C:\Windows\system32\drivers\etc> ipconfig /flushdns

Configuration IP de Windows

Cache de résolution DNS vidé.
```

Configuration du SSH sur la machine Debian

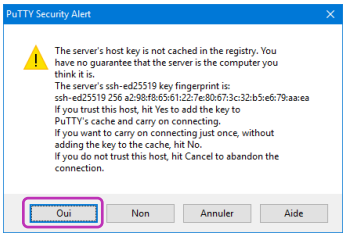
Pour mener à bien ce TP, nous avons besoin d'installer OpenSSH-server, Putty et WinSCP :

La configuration de l'accès SSH s'effectue en saisissant la commande : `apt install openssh-server`

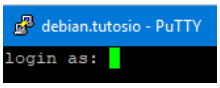
☑ Sur la machine Windows, téléchargez l'outil « Putty » ici : Download PuTTY: latest release (0.74) (greenend.org.uk)
☑ Lancez Putty en indiquant les paramètres de votre machine Debian :



Pour la 1ère connexion en SSH, cliquez sur « oui »



La fenêtre d'authentification s'affiche :



☑ Saisissez les identifiants préalablement configurés
Attention, *l'accès SSH en tant que root, sur la machine Debian, n'est pas possible par défaut* pour des raisons de sécurité. Il est possible d'activer l'accès root en SSH de la manière suivante :

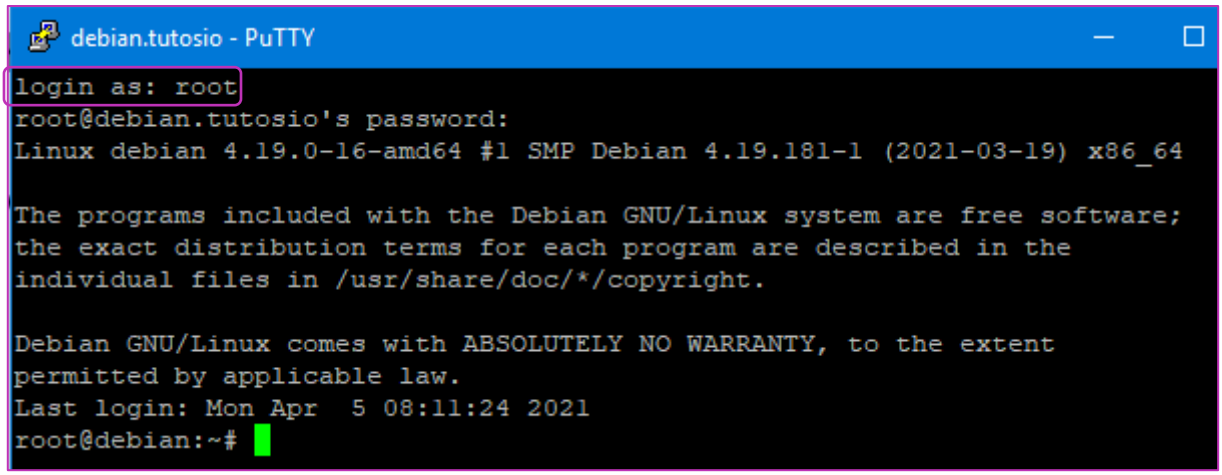
☑ Déplacez-vous dans le dossier « /etc/ssh » : `cd /etc/ssh`
☑ Ouvrez, avec nano, le fichier « sshd_config » : `nano sshd_config`

☑ Modifiez la section « # Authentification » ainsi :
☑ Sauvegardez les modifications

```
# Authentication:

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

☑ Relancez le service SSH sur votre machine Debian (« ***systemctl restart ssh*** ») et testez l'accès :

L'accès SSH en tant que root est fonctionnel. Attention, nous présentons cette méthode à des fins pédagogiques mais il est fortement déconseillé de laisser l'accès au root en mode SSH pour des raisons de sécurité !!!



4 – CONFIGURATION DE L'ACCES HTTP DU SERVEUR WEB APACHE

Lors de l'installation du serveur web Apache 2.4, ***seul l'accès via le port HTTP (80) est actif par défaut***. Pour comprendre pourquoi seul le site http est actif, nous pouvons effectuer les vérifications suivantes :

☑ Contrôle des sites actifs : `ls /etc/apache2/sites-enabled`

Ici nous constatons que le site « *000-default.conf* » est bien actif sur le serveur web Apache puisqu'il fait partie du dossier « *sites-enabled* » : `000-default.conf`

☑ Editez le fichier « *000-default.conf* » **depuis le dossier « /etc/apache2/sites-available** :

```
nano /etc/apache2/sites-available/000-default.conf
```

Pour accéder à la page web (en http), nous avions saisi précédemment l'adresse IP du serveur web Apache. Nous allons modifier le fichier « 000-default.conf » de manière à afficher la page en saisissant le nom de domaine local plutôt que l'adresse IP (dans notre cas http://debian.tutosio).

☑ Saisissez la commande suivante : `nano /etc/apache2/sites-available/000-default.conf`

Le fichier « virtualhost » du site web par défaut d'Apache s'ouvre :

☑ Ajoutez la ligne « *ServerName debian.tutosio* » (adaptez le nom de domaine en fonction du FQDN de votre machine Debian)

Le fichier « ***000-default.conf*** » doit se présenter ainsi :

```
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last r
        # However, you must set it for any further virtua
        #ServerName www.example.com

        ServerAdmin webmaster@localhost
        ServerName debian.tutosio
        DocumentRoot /var/www/html

        # Available loglevels: trace8, ..., trace1, debug
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        # For most configuration files from conf-available/, which are
        # enabled or disabled at a global level, it is possible to
        # include a line for only one particular virtual host. For example the
        # following line enables the CGI configuration for this host only
        # after it has been globally disabled with "a2disconf".
        #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
```

Ici on ajoute le nom de domaine local qui correspond à notre FQDN. Ainsi, nous pourrons accéder directement à la page par défaut d'Apache en saisissant le nom de domaine dans la barre d'adresse du navigateur.

Emplacement des fichiers « error.log » et « access.log ».

☑ Enregistrez le fichier avec la modification
☑ Relancez le service Apache `systemctl restart apache2`
☑ Ouvrez un onglet dans le navigateur et testez l'accès en saisissant « **http://debian.tutosio** » :

5 – CONFIGURATION DE L'ACCES HTTPS DU SERVEUR WEB APACHE

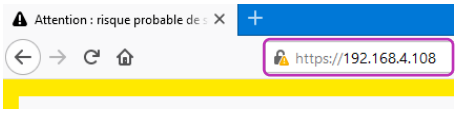
Apache propose un accès HTTPS. Cependant, *cet accès n'est pas activé par défaut*. Le site web HTTPS par défaut se trouve dans « /etc/apache2/sites-available », sous le nom « **default-ssl.conf** » :

```
root@debian:~# ls /etc/apache2/sites-available
000-default.conf  default-ssl.conf
```

Nous pouvons activer le site web HTTPS de la manière suivante :

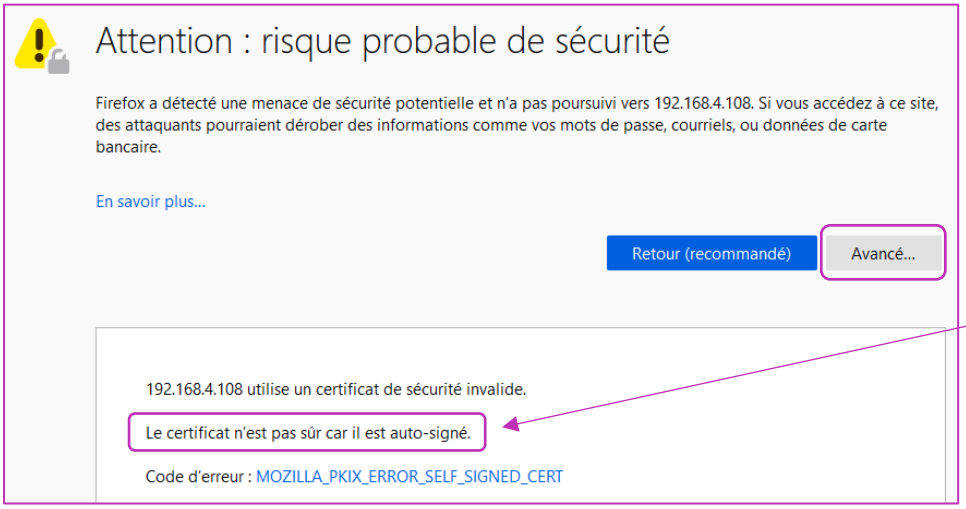
☑ Activez le site « *default-ssl.conf* » en saisissant a commande : `a2ensite default-ssl.conf`
☑ Relancez Apache pour activer le site HTTPS par défaut : `systemctl reload apache2`
☑ Activez le mode SSL sur le serveur web Apache : `a2enmod ssl`
☑ Redémarrez Apache pour valider l'activation du mode SSL : `systemctl restart apache2`

☑ Testez l'accès à votre serveur web en mode HTTPS en saisissant « ***https://IP de votre machine Debian*** » :



L'accès HTTPS est fonctionnel mais le navigateur affiche une alerte liée au certificat auto-signé et l'absence d'Autorité de Certification.

L'accès HTTPS est fonctionnel mais le navigateur affiche une alerte liée au certificat auto-signé et l'absence d'Autorité de Certification est indiquée par le navigateur puisque nous sommes en présence d'un *certificat auto-signé* (bouton « Avancé… » :



Par défaut, Apache a généré un *certificat auto-signé valable 10 ans* mais ce certificat n'a pas été validé par une Autorité de Certification reconnue par le navigateur.

Analyse et configuration du fichier « *default-ssl.conf* » (*attention, ici, il est conseillé de copier le fichier d'origine et de travailler sur une copie pour préserver l'original*) :

```
nano /etc/apache2/sites-available/default-ssl.conf
```

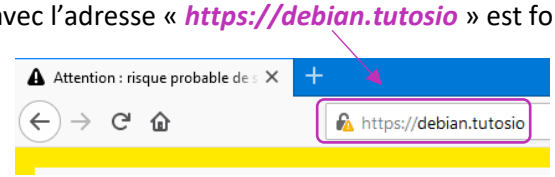
```
<IfModule mod_ssl.c>
        <VirtualHost _default_:443>
                ServerAdmin webmaster@localhost
                ServerName debian.tutosio
                DocumentRoot /var/www/html

                # Available loglevels: trace8, ..., trace1, d
                # error, crit, alert, emerg.
                # It is also possible to configure the loglev
                # modules, e.g.
                #LogLevel info ssl:warn

                ErrorLog ${APACHE_LOG_DIR}/error.log
                CustomLog ${APACHE_LOG_DIR}/access.log combined

                # For most configuration files from conf-available/
                # enabled or disabled at a global level, it is possi
                # include a line for only one particular virtual hos
                # following line enables the CGI configuration for this host only
                # after it has been globally disabled with "a2disconf".
                #Include conf-available/serve-cgi-bin.conf

                #   SSL Engine Switch:
                #   Enable/Disable SSL for this virtual host.
                SSLEngine on

                #   A self-signed (snakeoil) certificate can be create
                #   the ssl-cert package. See
                #   /usr/share/doc/apache2/README.Debian.gz for more i
                #   If both key and certificate are stored in the same file, only the
                #   SSLCertificateFile directive is needed.
                SSLCertificateFile      /etc/ssl/certs/ssl-cert-snakeoil.pem
                SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```

Ici on ajoute le nom de domaine local qui correspond à notre FQDN. Ainsi, nous pourrons accéder directement à la page HTTPS par défaut d'Apache en saisissant le nom de domaine dans la barre d'adresse du navigateur.

Emplacement des fichiers « error.log » et « access.log ».

Emplacement des fichiers de certificat auto-signé du serveur Apache.

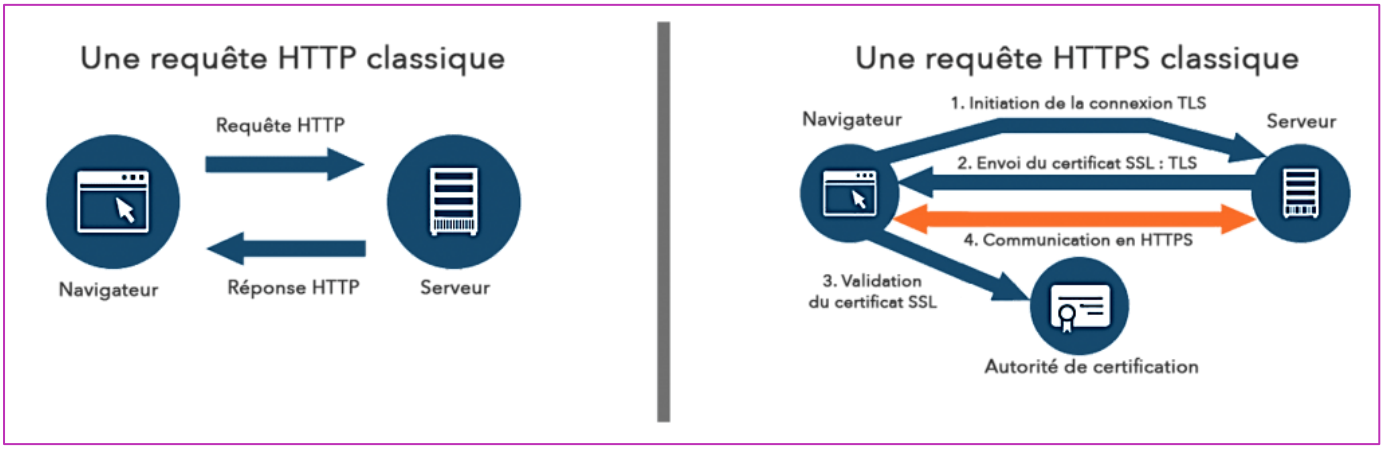
☑ Ajoutez la ligne « *ServerName debian.tutosio* » et relancez Apache : « *systemctl restart apache2* »
☑ L'accès HTTPS avec l'adresse « ***https://debian.tutosio*** » est fonctionnel (avec l'alerte de sécurité) :



6 – CONFIGURATION OPENSSL

Dans cette partie, nous allons installer le module OPENSSL sur la machine Debian afin de créer une Autorité de Certification qui signera les certificats émis.

Principe :



☑ Installez OpenSSL sur votre machine Debian en saisissant : `apt install openssl`

1ère partie : CREATION DE L'AUTORITE DE CERTIFICATION (CA)

Ici, nous n'aurons pas recours à une autorité reconnue mais à nous-mêmes (nous serons autorité de certification). *Nous allons créer les certificats SSL dans un dossier spécifique pour plus de clarté*. Cette phase comporte 2 parties :

- ✓ la création de la clé privée de l'Autorité de Certification (fichier *.key*)
- ✓ la création du certificat auto-signé de l'Autorité de Certification (fichier *.crt*)

CREATION DE LA CLE PRIVEE DE L'AUTORITE DE CERTIFICATION

☑ Créez un dossier « ssl » dans « /etc/apache2 » : `cd /etc/apache2` puis `mkdir ssl`
☑ Placez-vous dans le dossier ssl créé précédemment de manière à générer les certificats dedans
☑ Créez la clé privée de votre Autorité de Certification : `openssl genrsa 4096 > ca.key`

Ici, la clé est générée selon un algorithme de chiffrement asymétrique de type RSA 4096 bits et portera le nom de « ca.key ». Nous générons une clé privée sans « pass phrase » pour simplifier mais il faudrait, dans la pratique, protéger cette clé par un mot de passe fort (minimum 12 caractères selon l'ANSSI). Pour ajouter une « pass phrase », il faut ajouter -des3 ou -aes256 par exemple après 4096.

CREATION DU CERTIFICAT AUTO-SIGNE DE L'AUTORITE DE CERTIFICATION

☑ Saisissez la commande suivante et complétez les options :

```
openssl req -new -x509 -days 365 -nodes -key ca.key > ca.crt
```

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:FRANCE
Locality Name (eg, city) []:AVRANCHES
Organization Name (eg, company) [Internet Widgits Pty Ltd]:NDLP
Organizational Unit Name (eg, section) []:SIO
Common Name (e.g. server FQDN or YOUR name) []:debian.tutosio
Email Address []:
```

Attention, renseignez correctement le « common name » qui correspond à votre FQDN !

A ce stade, le dossier « /etc/apache2/ssl » comporte 2 fichiers : `ca.crt  ca.key`

- le fichier « ca.key » correspond à la clé privée de notre Autorité de Certification
- le fichier « ca.crt » correspond au certificat auto-signé de notre Autorité de Certification

2ème partie : CREATION DU CERTIFICAT AUTO-SIGNE DU SERVEUR WEB APACHE

Dans cette partie, nous aurons 3 étapes : nous allons créer une clé privée (*.key*) pour notre serveur web Apache, un fichier de demande de signature (*.csr*) et un certificat auto-signé (*.crt*) pour notre serveur web Apache.

- ✓ la création de la clé privée du serveur web Apache
- ✓ la création du certificat auto-signé du serveur web Apache

CREATION DE LA CLE PRIVEE DU SERVEUR WEB APACHE

☑ Saisissez la commande : `openssl genrsa 4096 > cleprivapache.key`

☑ Protégez la clé privée en changeant les droits : `chmod 400 cleprivapache.key`

CREATION DU FICHIER DE DEMANDE DE SIGNATURE A PARTIR DE LA CLE PRIVEE

☑ Saisissez la commande : `openssl req -new -key cleprivapache.key > demandesignature.csr`

☑ Répondez aux questions pour générer le fichier « .csr » :

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:FRANCE
Locality Name (eg, city) []:AVRANCHES
Organization Name (eg, company) [Internet Widgits Pty Ltd]:NDLP
Organizational Unit Name (eg, section) []:SIO
Common Name (e.g. server FQDN or YOUR name) []:debian.tutosio
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:labosio
An optional company name []:
```

Ici vous devez saisir un mot de passe pour générer la demande (ne le perdez pas !).

A ce stade, votre dossier /etc/apache2/ssl contient 4 fichiers :

```
ca.crt  ca.key  cleprivapache.key  demandesignature.csr
```

- le fichier « ***ca.crt*** » correspond au certificat de l'Autorité de Certification
- le fichier « ***ca.key*** » correspond à la clé privée de l'Autorité de Certification
- le fichier « ***cleprivapache.key*** » correspond à la clé privée de votre serveur Apache
- le fichier « ***demandesignature.csr*** » correspond à la demande à faire signer par l'Autorité de Certification

SIGNATURE DU FICHIER DE DEMANDE PAR L'AUTORITE DE CERTIFICATION

☑ Saisissez la commande suivante (attention, vérifiez bien la syntaxe) :
openssl x509 -req -in demandesignature.csr -out certifapache.crt -CA ca.crt -CAkey ca.key -CAcreateserial -days 365

```
root@debian:/etc/apache2/ssl# openssl x509 -req -in demandesignature.csr -out certifapache.crt -CA ca.crt -CAkey ca.key -CAcreat
eserial -days 365
Signature ok
subject=C = FR, ST = FRANCE, L = AVRANCHES, O = NDLP, OU = SIO, CN = debian.tutosio
Getting CA Private Key
```

L'option -CAcreateserial est à utiliser seulement la 1ère fois.
Le dossier « /etc/apache2/ssl » comporte maintenant 6 fichiers :

```
ca.crt  ca.key  ca.srl  certifapache.crt  cleprivapache.key  demandesignature.csr
```

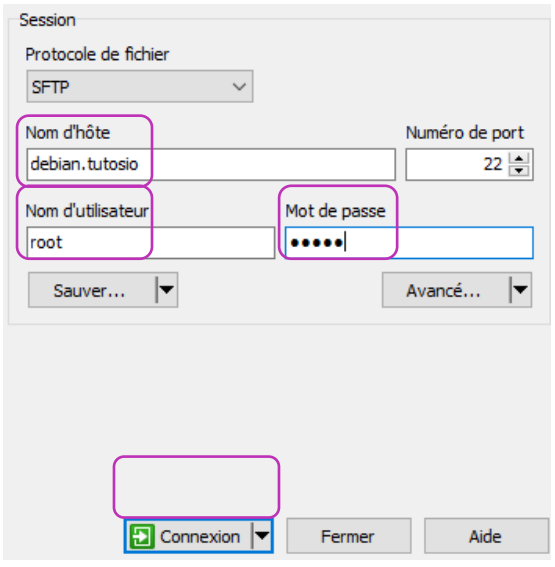
- le fichier « ***ca.crt*** » correspond au certificat de l'Autorité de Certification
- le fichier « ***ca.key*** » correspond à la clé privée de l'Autorité de Certification
- le fichier « ***ca.srl*** » contient un identifiant qui sera incrémenté pour une nouvelle demande de signature
- le fichier « ***certifapache.crt*** » correspond au certificat auto-signé de votre serveur Apache
- le fichier « ***cleprivapache.key*** » correspond à la clé privée de votre serveur Apache
- le fichier « ***demandesignature.csr*** » correspond à la demande à faire signer par l'Autorité de Certification

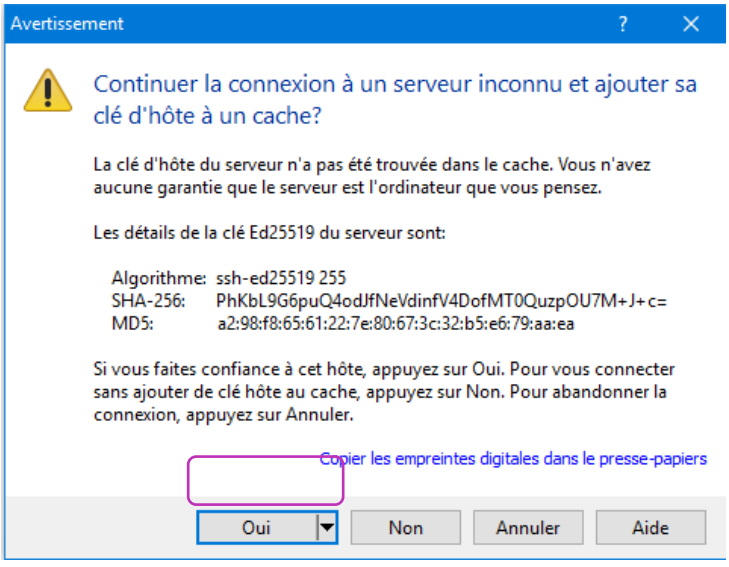
3ème partie : IMPORTATION DU CERTIFICAT DE L'AUTORITE DANS LE NAVIGATEUR WEB HOTE

Dans cette partie, nous allons ajouter notre Autorité de Certification dans le « magasin » de certificats de notre navigateur web (ici, nous utiliserons Firefox).

Pour réaliser ce travail, nous allons installer et utiliser WinSCP pour transférer le certificat vers notre machine Windows (il existe d'autres méthodes mais, pour des raisons pédagogiques, nous en profitons pour présenter WinSCP).

☑ Téléchargez et installez WinSCP depuis le lien : WinSCP Official Site Download
☑ Lancez WinSCP et connectez-vous à votre machine Debian :

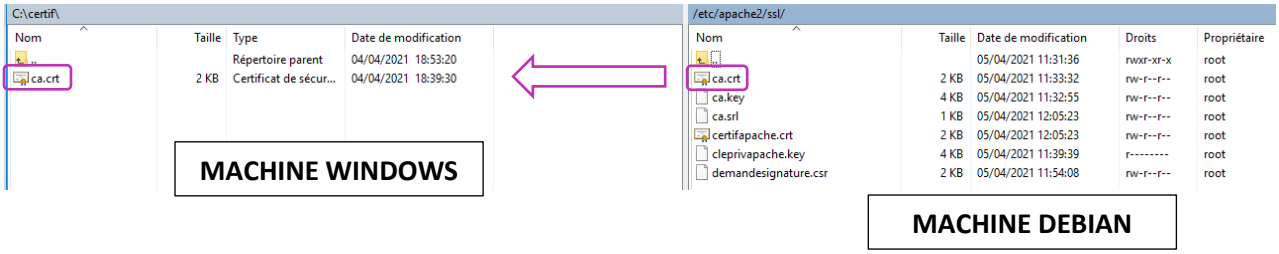




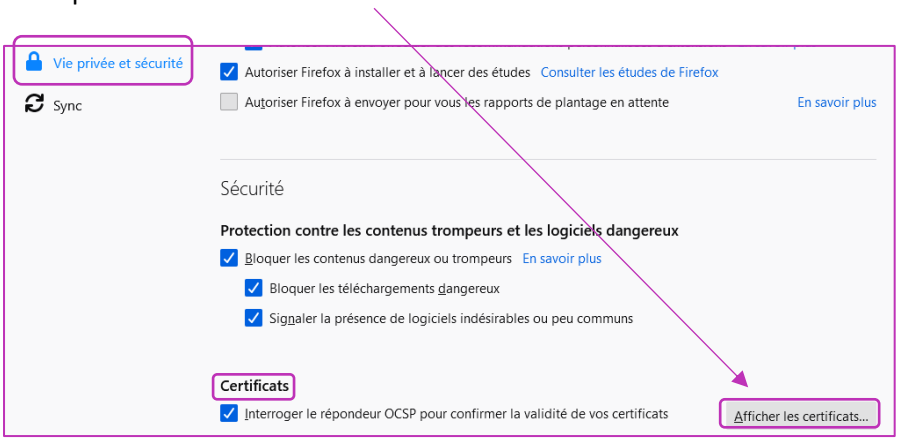
☑

WinSCP affiche une double fenêtre correspondant aux machines Windows et Debian :

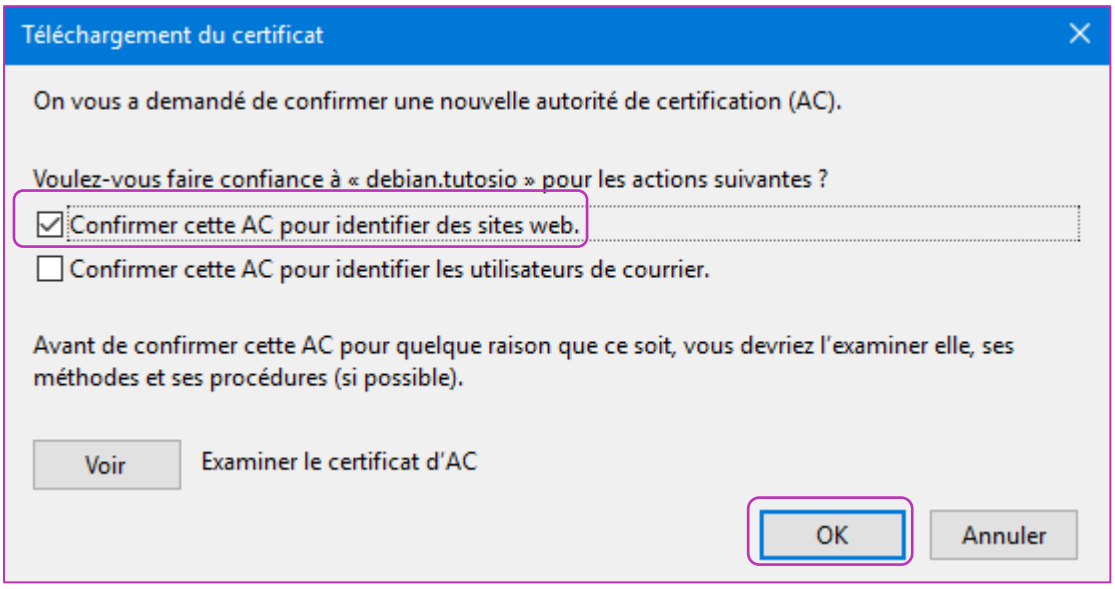
☑ Allez dans le dossier /etc/apache2/ssl et faites glisser le fichier certificat de votre Autorité de Certificaton « ca.crt » dans un dossier de votre machine Windows (afin de le récupérer) :



☑ Lancez votre navigateur (ici Firefox) et ouvrez les options
☑ Dans « Vie privée et sécurité », descendez jusqu'à la rubrique « Certificats »
☑ Cliquez le bouton « Afficher les certificats :



☑ Cliquez le bouton « Importer » puis recherchez votre certificat « ca.crt » importé précédemment
☑ Cliquez l'option « Confirmer cette AC pour identifier les sites web »
☑ Cliquez le bouton « Ok » pour confirmer l'importation



En cliquant à nouveau le bouton « Afficher les certificats » on constate que le certificat de notre Autorité est bien présent dans le « magasin des autorités racines de confiance » :

Maintenant que le certificat de l'Autorité est importé dans le navigateur, il faut modifier la configuration du fichier « /etc/apache2/sites-available/default-ssl.conf » pour stipuler l'emplacement des fichiers certificats du serveur Apache (pour rappel, ces fichiers ont été générés précédemment dans /etc/apache2/ssl).

☑ Ouvrez le fichier « default-ssl.conf » (situé dans /etc/apache2/sites-available) :

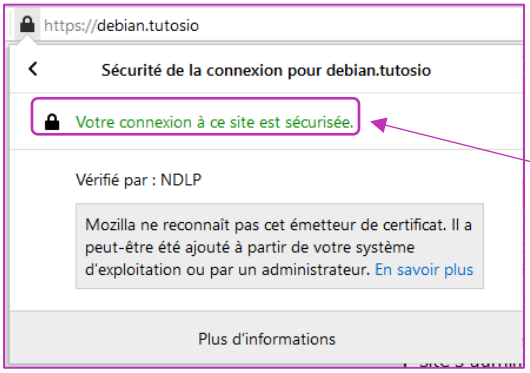
```
<IfModule mod_ssl.c>
        <VirtualHost _default_:443>
                ServerAdmin webmaster@localhost
                ServerName debian.tutosio
                DocumentRoot /var/www/html

                # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
                # error, crit, alert, emerg.
                # It is also possible to configure the loglevel for particular
                # modules, e.g.
                #LogLevel info ssl:warn

                ErrorLog ${APACHE_LOG_DIR}/error.log
                CustomLog ${APACHE_LOG_DIR}/access.log combined

                # For most configuration files from conf-available/, which are
                # enabled or disabled at a global lev
                # include a line for only one particu
                # following line enables the CGI conf
                # after it has been globally disabled
                #Include conf-available/serve-cgi-bin.conf

                #   SSL Engine Switch:
                #   Enable/Disable SSL for this virtual host.
                SSLEngine on

                #   A self-signed (snakeoil) certificate can be created by installing
                #   the ssl-cert package. See
                #   /usr/share/doc/apache2/README.Debian.gz for more info.
                #   If both key and certificate are stored in the same file, only the
                #   SSLCertificateFile directive is needed.
                SSLCertificateFile      /etc/apache2/ssl/certifapache.crt
                SSLCertificateKeyFile /etc/apache2/ssl/cleprivapache.key_
```

Indiquez, ici, l'emplacement des fichiers « .crt » et « .key » du serveur Apache.

☑ Relancez Apache en saisissant : « systemctl restart apache2 »
☑ Connectez-vous à https://debian.tutosio : il n'y a plus d'alerte de sécurité !

Votre site web HTTPS est pleinement reconnu par votre Autorité :

En cliquant le cadenas, on constate que la connexion est devenue pleinement sécurisée :



Notre Autorité de Certification (NDLP du nom de l'organisation donné lors de la création du certificat de l'Autorité) a validé le certificat signé émis par Apache : la connexion est sécurisée (mode HTTPS activé).

7 – REDIRECTION AUTOMATIQUE DES REQUETES HTTP VERS HTTPS

Dans ce guide, nous avons activé le site https par défaut d'Apache mais nous avons laissé le site HTTP actif également. Il n'est pas utile de désactiver le site « 000-default.com ».

Nous allons plutôt configurer une redirection permanente des requêtes http qui arrivent sur le port 80 vers le site sécurité https (port 443), en modifiant simplement le ficher « 000-default.conf » :

☑ Editez le fichier « 000-default.conf » et ajoutez la ligne « Redirect permanent / https://debian.tutosio) :

```
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

        ServerAdmin webmaster@localhost
        ServerName debian.tutosio
        Redirect permanent / https://debian.tutosio
        DocumentRoot /var/www/html
```

☑ Relancez Apache en saisissant la commande : « systemctl restart apache2 »
☑ Saisissez, dans votre navigateur : debian.tutosio : vous êtes redirigé(e) vers le site HTTPS automatiquement !

8 – LES COMMANDES UTILES A CONNAITRE

ANALYSE DES LOGS SUR SERVEUR APACHE - QUELQUES COMMANDES UTILES

apache2ctl configtest = Vérifier si la syntaxe des fichiers de conf est correcte = exploitable par apache2 pour démarrer. (tous les fichiers .conf : mods-available, conf-available, sites-available

apache2ctl graceful = redémarrer les process (worker) sans casser les connexions existantes.

tail -F /var/log/apache2/access.log = voir en temps réel le fichier de log

tail -n 200 /var/log/apache2/access.log = voir les 200 dernières lignes du fichier

**tail -F /var/log/apache2/access.log /var/log/apache2/access2.log /var/log/apache2/accessN.log ** = voir en temps réel les fichiers de log

less /var/log/apache2/access.log = afficher en lecture seule le fichier concerné (possibilité de naviguer dedans)

QUELQUES COMMANDES UTILES

Activer le mode SSL	**a2enmod ssl**
Activer un site Apache	**a2ensite xxx.conf (le fichier doit se trouver dans /etc/apache2/sites-available)**
Désactiver un site Apache	**a2dissite xxx.conf (le fichier doit se trouver dans /etc/apache2/sites-available)**
Vérifier les sites Apache actifs	Se rendre dans /etc/apache2/sites-enabled