



DDWS



*École La Plateforme,
Établissement d'enseignement supérieur à Marseille,
Bachelor IT 1A, Programmation informatique.*



Sommaire

Sommaire	2
Job01	3
Job02	3
Job03	4
Apache HTTP Server, le leader	4
Nginx, le spécialiste des connexions simultanées	4
Apache Tomcat, origine Java	4
Job04	5
Job05	6
Job06	7
Configuration des fichier /etc/apache2/	7
Configuration des fichier /etc/bind/	7
Configuration de l'hôte	9
Job07	10
Job08	11
Job09	12
Job10	13
Pour aller plus loin...	13
Prérequis	13
Génération de la clé privée	13
Génération du fichier de demande de signature	14
Génération du certificat	14
Configuration des fichier /etc/apache2/	14
Configuration des fichier /etc/bind/	15
Configuration de l'hôte	16
Questions	17
Annexes	18
Tables des Figures	18

Job01

Installation d'une VM Debian avec interface graphique.

Job02

Installation du serveur Web Apache2, à l'aide de la commande : `apt-get -y install apache2`, afin d'installer le paquet. A la fin de l'installation, nous pouvons nous rendre sur la page de lancement d'apache en renseignant son IP dans la barre de recherche de son navigateur internet.

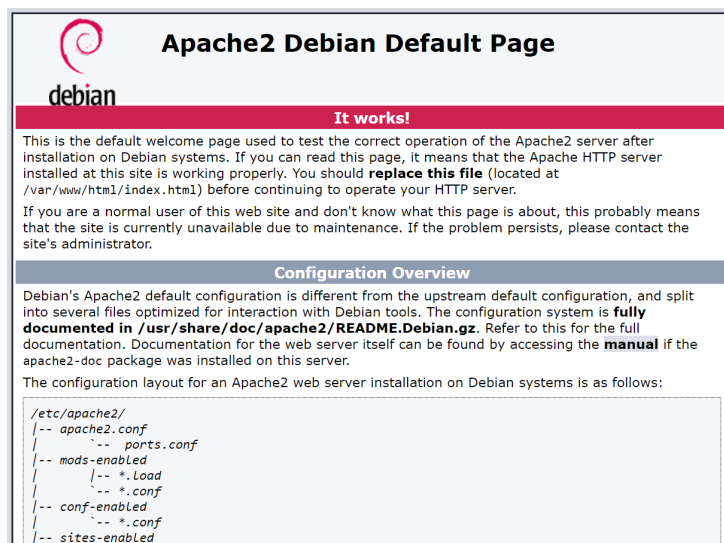


Figure 1 : page de lancement d'Apache

N.B : Pour tous les exercices précédant la mise en place d'un DHCP, je vais configurer une IP static dans le fichier `/etc/network/interfaces` voir figure 2. Afin d'actualiser ce fichier, on entre la commande suivante : `sudo /etc/init.d/networking restart`, on peut vérifier la mise en place de la configuration avec `ip a`.

```
GNU nano 5.4 /etc/network/interfaces *  
# This file describes the network interfaces available on your system  
# and how to activate them. For more information, see interfaces(5).  
  
source /etc/network/interfaces.d/*  
  
# The loopback network interface  
#auto lo  
#iface lo inet loopback  
  
auto enp2s1  
#iface enp2s1 inet dhcp  
iface enp2s1 inet static  
    address 10.0.0.1  
    netmask 255.255.255.0  
    gateway 10.0.0.254
```

Figure 2 : configuration pour une IP static

Job03

Voici quelques serveurs WEB existant, comme nous travaillons sur du linux les serveurs présentés seront open source.

Apache HTTP Server, le leader

“Le logiciel HTTP Apache Server, nommé souvent httpd ou plus simplement Apache, à plus de 20 ans d'âge. Une pérennité qui s'explique par une popularité impressionnante : 52% des sites web dans le monde sont hébergés sur un serveur qui utilise Apache. Surtout, si l'on croise souvent Apache sur des distributions Linux, le produit est disponible sur OS X et Windows.

Apache est disponible sous licence... Apache v2. Ce serveur Web utilise une architecture modulaire et évolutive, de manière à étendre les fonctionnalités en cas de besoin, comme l'équilibrage de charge par exemple. A noter que depuis la version 2.4, Apache prend en charge le protocole HTTP/2 grâce au module mod_http2. Enfin, il convient de dire que la longévité de ce logiciel lui confère une documentation importante tout comme l'intégration d'autres logiciels.”

Nginx, le spécialiste des connexions simultanées

“Le développement de Nginx a débuté en 2002 sous les bons auspices d'Igor Sysoev, et la première version publique date de 2004. Nginx répond à la question de la prise en charge de multiples connexions simultanées sur un serveur web. Avec 30% des sites web qui tournent sur Nginx, nulle doute que cette mission est remplie.

Dans ce but, Nginx repose sur une architecture événementielle asynchrone, ce qui permet de gérer des sessions simultanées massives. A noter que les administrateurs apprécient aussi Nginx en raison de son usage modéré des ressources tout comme sa capacité à monter facilement en charge. Nginx est publié sous une licence BSD, et peut être déployé en tant que serveur Web mais aussi comme serveur proxy ou comme équilibreur de charge.”

Apache Tomcat, origine Java

“Apache Tomcat est un conteneur servlet Java open source qui fonctionne comme serveur Web en étendant les capacités d'un serveur. Ce sont les alternatives Java aux technologies telles que PHP et ASP.NET. La base du code de Tomcat a été donnée par Sun Microsystems à l'Apache Software Foundation en 1999. Un peu moins de 1% de tous les sites web utilisent Apache Tomcat.

Apache Tomcat est publié sous licence Apache v2, et est généralement utilisé pour exécuter des applications Java. Il peut cependant être étendu au rôle de serveur web normal en utilisant Coyote. Apache Tomcat est souvent classé parmi d'autres serveurs d'applications Java open source comme JBoss, Wildfly et Glassfish.”

Job04

Pour la mise en place d'un DNS sur le serveur Linux, donc en local, il faudra faire correspondre l'adresse IP de mon serveur au nom de domaine local, c'est à dire : l'adresse IP **10.0.0.1** au nom de domaine **dnsproject1.prepa.com** (nom de domaine différent de celui demandé pour ma part, car j'ai voulu différencier mes configurations **http://** et **https://**, mais j'y reviendrais plus tard).

Afin de mettre en place le DNS, il faut modifier le fichier **/etc/hosts** à partir de la commande **sudo nano /etc/hosts** et rentrer la ligne suivante dans le fichier :

- **10.0.0.1 dnsproject1.prepa.com.**
- Le choix de l'adresse IP est juste personnel.

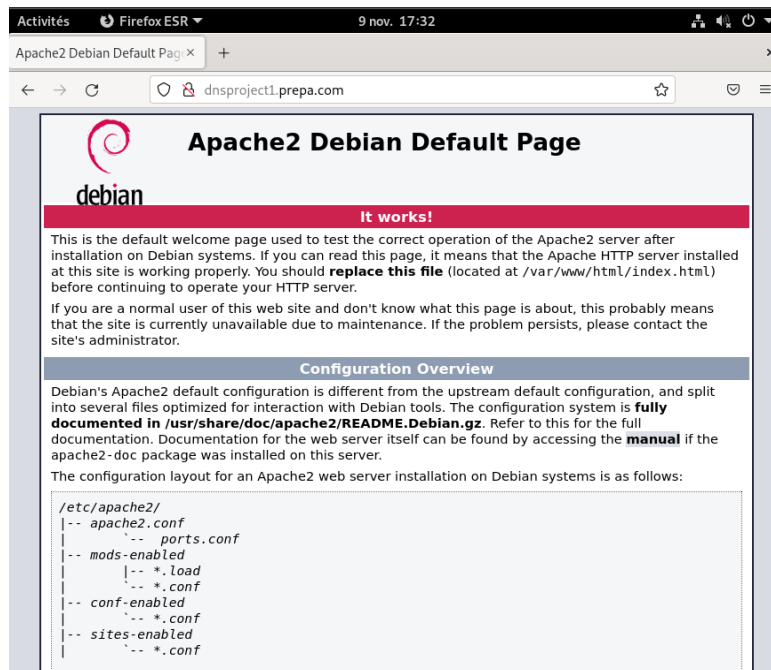


Figure 3 : page de lancement d'Apache avec un DNS

Il est également possible de ping le serveur via le nom de domaine donné lors de l'étape précédente. Pour la réalisation du ping, on utilise la commande : **ping dnsproject1.prepa.com -c 4** (**-c 4** afin que seulement quatre paquets soient transmis lors du ping). On remarque bien que le nom de domaine : **dnsproject1.prepa.com**, correspond à l'adresse IP **10.0.0.1**.

```
yoann@debian:~$ ping dnsproject1.prepa.com -c 4
PING dnsproject1.prepa.com (10.0.0.1) 56(84) bytes of data.
64 bytes from www.dnsproject1.prepa.com (10.0.0.1): icmp_seq=1 ttl=64 time=0.030 ms
64 bytes from www.dnsproject1.prepa.com (10.0.0.1): icmp_seq=2 ttl=64 time=0.053 ms
64 bytes from www.dnsproject1.prepa.com (10.0.0.1): icmp_seq=3 ttl=64 time=0.054 ms
64 bytes from www.dnsproject1.prepa.com (10.0.0.1): icmp_seq=4 ttl=64 time=0.054 ms

--- dnsproject1.prepa.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3041ms
rtt min/avg/max/mdev = 0.030/0.047/0.054/0.010 ms
```

Figure 4 : ping du DNS de la page d'apache2

Job05

Le nom de domaine est l'appellation qui identifie un site internet et constitue le moyen de localisation et d'accès aux pages de ce site internet. Les règles de réservation d'un nom de domaine varient selon la nature du site :

Domaines géographiques à vocation nationale, selon la localisation géographique de l'entreprise :

- .fr (France)
- .de (Allemagne)
- .it (Italie)
- .eu (Union européenne)

Domaines génériques, à vocation internationale :

- .com (pour les activités commerciales)
- .net (pour les entreprises)
- .org (pour les associations ou organisations non gouvernementales, etc.)

Le nom de domaine est attribué à celui qui en demande la réservation en premier. C'est donc la règle du premier arrivé, premier servi qui prévaut. Pour réserver un nom de domaine, il faut s'adresser à l'organisme gestionnaire qui en a la charge.

La réservation du nom de domaine n'est pas une protection au niveau de la propriété intellectuelle. En effet, le nom de domaine ne correspond pas à une marque, qui est un titre de propriété intellectuelle protégé après son dépôt.

Pour protéger son nom de domaine des cybersquatteurs ou des concurrents, il est recommandé d'enregistrer également le nom de domaine sous forme de marque en complément de la réservation du nom de domaine.

Il est possible, avant d'effectuer une réservation de nom de domaine et un dépôt de marque, d'en vérifier la disponibilité, pour éviter les conflits entre noms de domaine, marques ou dénominations sociales.

Job06

Pour la réalisation de cet exercice, il va falloir connecter mon hôte au nom de domaine local de mon serveur, afin que la page de lancement d'Apache soit accessible via ce même nom de domaine. Pour cela, on installe en plus du paquet **apache2**, le paquet **bind9** à l'aide de la commande : `apt-get -y install bind9`, (**bind9** nous permet de communiquer avec les serveurs de noms secondaires).

Configuration des fichier `/etc/apache2/`

Le fichier nous intéressant est `/etc/apache2/sites-available/000-default.conf`, donc on réalise une copie de ce fichier avec un nom adapté, pour cela on entre la commande suivante :

```
sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/dnsproject1.prepa.com.conf.
```

Puis on modifie le fichier `dnsproject1.prepa.com.conf`, à l'aide de la commande `sudo nano /etc/apache2/sites-available/dnsproject1.prepa.com.conf`. Afin d'obtenir la configuration de la **figure 5**. Une fois les modifications apportées au fichier `^O^X`, puis on entre la commande suivante : `sudo a2ensite dnsproject1.prepa.com`, permettant d'activer un site **apache2**, ainsi que la commande `sudo systemctl restart apache2`.

```
/etc/apache2/sites-available/dnsproject1.prepa.com.conf *
```

```
<VirtualHost *:80>
```

```
    ServerName dnsproject1.prepa.com
```

```
    ServerAlias www.dnsproject1.prepa.com
```

```
    DocumentRoot /var/www/html
```

```
    ErrorLog ${APACHE_LOG_DIR}/error.log
```

```
    CustomLog ${APACHE_LOG_DIR}/access.log combined
```

```
</VirtualHost>
```

```
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Figure 5 : fichier de configuration d'apache2 du dns1

Configuration des fichier `/etc/bind/`

Le fichier nous intéressant est `/etc/bind/db.empty`, donc on réalise une copie de ce fichier avec un nom adapté, pour cela on entre la commande suivante :

```
sudo cp /etc/bind/db.empty /etc/bind/dnsproject1.prepa.com.
```

Puis on modifie le fichier `dnsproject1.prepa.com`, à l'aide de la commande `sudo nano /etc/bind/dnsproject1.prepa.com`. Afin d'obtenir la configuration de la **figure 6**. Une fois les modifications apportées au fichier `^O^X`, puis on entre la commande suivante : `sudo systemctl restart bind9`.

```
GNU nano 5.4 /etc/bind/dnsproject1.prepa.com *
; BIND reverse data file for empty rfc1918 zone
;
; DO NOT EDIT THIS FILE - it is used for multiple zones.
; Instead, copy it, edit named.conf, and use that copy.
;
$TTL      86400
@         IN      SOA      dnsproject1.prepa.com. root.dnsproject1.prepa.com. (
                                100          ; Serial
                                604800       ; Refresh
                                86400        ; Retry
                                2419200     ; Expire
                                86400 )     ; Negative Cache TTL
;
@         IN      NS       ns1.dnsproject1.prepa.com.
@         IN      A        10.0.0.1
ns1       IN      A        10.0.0.1
www       IN      A        10.0.0.1
```

Figure 6 : fichier de configuration de bind du dns1

Un DNS est constitué de plusieurs enregistrements, les Ressources Records (RR), définissant les diverses informations relatives au domaine. Le premier enregistrement est consacré à la résolution de noms, dans notre cas, il s'agit du fichier **db.empty**, que l'on a copié en **dnsproject1.prepa.com**.

Quelques explications :

\$TTL : (Time To Live) exprime la durée (en secondes) de validité, par défaut, des informations que contiennent les RR. Une fois ce délai expiré, il est nécessaire de vérifier à nouveau les données.

Les différents types :

- **SOA**: permet de définir les informations relatives à la zone. En l'occurrence le nom du serveur DNS primaire "**dnsproject1.prepa.com.**" et l'adresse mail du contact technique (**root.dnsproject1.prepa.com.** ; le **@** est remplacé par un point). Il est composé de plusieurs champs :
 - **Serial** : est un entier non signé 32 bits. C'est le numéro de série à incrémenter à chaque modification du fichier. Il permet au serveur secondaire de recharger les informations qu'ils ont. L'usage général vient à le formater de cette manière YYYYMMDDXX, soit pour la première modification du 09/11/2022 -> 2007040101, pour la seconde 2007040102.
 - **Refresh** : définit la période de rafraîchissement des données.
 - **Retry** : si une erreur survient au cours du dernier rafraîchissement, celle-ci sera répétée au bout du délai Retry.
 - **Expire** : le serveur sera considéré comme non disponible au bout du délai Expire.
 - **Negative cache TTL** : définit la durée de vie d'une réponse NXDOMAIN de notre part.
- **NS** : renseigne le nom des serveurs de noms pour le domaine.
- **A** : associé une nom d'hôte à une adresse ipv4 (32 bits).

Les classes : **IN** détermine l'association a la classe Internet.

Il existe aussi d'autre type comme :

- **MX** : renseigne sur le serveur de messagerie. Plusieurs peuvent être définis. Ainsi, il est possible de leur donner une priorité en leur affectant un numéro. Plus bas est le numéro, plus haute est la priorité.
- **AAAA** : associe une nom d'hôte à une adresse ipv6 (128 bits)
- **CNAME** : identifie le nom canonique d'un alias (un nom pointant sur un autre nom)
- **PTR** : c'est simplement la résolution inverse (le contraire du type A).

Ensuite, il faut éditer le fichier `/etc/bind/named.conf.local`, ce fichier contient la configuration locale du serveur DNS, on y déclare les zones associées au domaine. Dans notre cas, il faut déclarer une zone `dnsproject1.prepa.com`, voir **figure 7**. Une fois toutes les configurations faites, on doit recharger chaque fichier donc on entre les commandes suivantes :

- `sudo systemctl restart apache2`, redémarrage d'**apache2**.
- `sudo systemctl restart bind9`, redémarrage de **bind9**.

```
GNU nano 5.4 /etc/bind/named.conf.local *
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

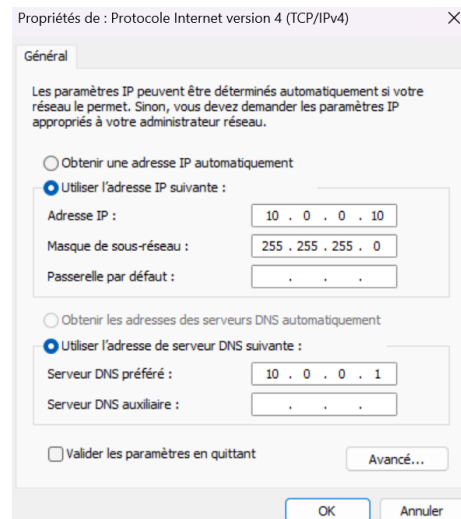
zone "dnsproject1.prepa.com"{
    type master;
    file "/etc/bind/dnsproject1.prepa.com";
};
```

Figure 7 : fichier de configuration de bind zone du dns1

Configuration de l'hôte

Il faudra se rendre sur son OS windows et taper dans le menu démarrer : *panneau de configuration - réseau et internet - centre réseau et partage - modifier les paramètre de la carte* (comme je suis sur vmware), prendre *VMware Network Adapter VMnet8 - propriété - protocole internet version 4* et mettre les adresse adapté à votre configuration, voir **figure 8**, **N.B** : cette configuration est adapté à mes paramètres.

Figure 8 : configuration réseau windows



Job07

Le protocole DHCP (Dynamic Host Configuration Protocol) est un service réseau permettant d'attribuer automatiquement des adresses IP aux clients sur un réseau. Il suit une architecture serveur-client où le client demande à un serveur DHCP d'obtenir une adresse IP. La plupart des routeurs ont un serveur DHCP intégré, mais nous pouvons également utiliser notre propre serveur DHCP.

Installation d'un serveur DHCP, à l'aide de la commande : `sudo apt-get -y install isc-dhcp-server`. Une fois le paquet installé, il faut configurer son serveur DHCP ISC, à partir des fichiers de configuration se trouvant dans `/etc/dhcp/dhcpd.conf`. Il est recommandé de faire une sauvegarde ou une backup des fichiers de configuration d'origine, par exemple, `sudo cp /etc/dhcp/dhcpd.conf /etc/dhcp/backup_dhcpd.conf`.

Avant de configurer le fichier `/etc/dhcp/dhcpd.conf`, on va éditer le fichier `/etc/resolv.conf` voir **figure 9**, afin d'appliquer les modifications on entre la commande `ifconfig enp2s1 10.0.0.1`, permettant de renseigner l'IP voulue pour une carte réseau donnée. La commande `nslookup www`, nous permet de vérifier si les modifications ont bien été apportées, voir **figure 10**.

```
GNU nano 5.4 /etc/resolv.conf *
# Generated by NetworkManager
search dnsproject1.prepa.com
nameserver 10.0.0.1
```

Figure 9 : fichier resolv.conf

```
root@debian:~# nslookup www
Server:      10.0.0.1
Address:     10.0.0.1#53

Name:   www.dnsproject1.prepa.com
Address: 10.0.0.1
```

Figure 10 : vérification nslookup www

Maintenant, il faut modifier le fichier `/etc/dhcp/dhcpd.conf`, à l'aide d'un éditeur comme nano, voir **figure 11** :

```
GNU nano 5.4 /etc/dhcp/dhcpd.conf

# A slightly different configuration for an internal subnet.
subnet 10.0.0.0 netmask 255.255.255.0 {
    range 10.0.0.20 10.0.0.40;
    option domain-name-servers 10.0.0.1;
    option domain-name "dnsproject1.prepa.com";
    option routers 10.0.0.1;
    option broadcast-address 10.0.0.255;
    default-lease-time 600;
    max-lease-time 7200;
}
```

Figure 11 : fichier dhcpd.conf

Selon la configuration actuelle, on a :

- La durée de bail par défaut pour un client est de 10 minutes (600 secondes) et la durée de bail maximale est de 2 heures (7200 secondes).
- Ce serveur DHCP est le serveur officiel du réseau local (indiqué par authoritative, ligne a décommenter plus haut dans le fichier de configuration).
- Le serveur transmettra l'adresse IP de la plage 10.0.0.20 à 10.0.0.40.
- Le serveur "conseillera" également au client d'utiliser 10.0.0.1 comme passerelle par défaut et comme serveurs DNS.

N.B : Penser à éditer le fichier `/etc/default/isc-dhcp-server`, afin de lier l'interface au serveur dans mon cas : `INTERFACESv4="enp2s1"`.

Une fois le serveur DHCP ISC configurer, il ne manque plus qu'à le lancer avec la commande `sudo systemctl restart isc-dhcp-server && sudo systemctl enable isc-dhcp-server`, on peut également vérifier son fonctionnement avec la commande `sudo systemctl status isc-dhcp-server`, voir **figure 12**.

```
root@debian:~# systemctl restart isc-dhcp-server
root@debian:~# systemctl status isc-dhcp-server
● isc-dhcp-server.service - LSB: DHCP server
   Loaded: loaded (/etc/init.d/isc-dhcp-server; generated)
   Active: active (running) since Thu 2022-11-10 10:17:05 CET; 1min 18s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 2216 ExecStart=/etc/init.d/isc-dhcp-server start (code=exited, sta
    Tasks: 4 (limit: 9225)
   Memory: 4.7M
      CPU: 73ms
   CGroup: /system.slice/isc-dhcp-server.service
           └─2232 /usr/sbin/dhcpd -4 -q -cf /etc/dhcp/dhcpd.conf enp2s1

nov. 10 10:17:03 debian systemd[1]: Starting LSB: DHCP server...
nov. 10 10:17:03 debian isc-dhcp-server[2216]: Launching IPv4 server only.
nov. 10 10:17:03 debian dhcpd[2232]: Wrote 0 leases to leases file.
nov. 10 10:17:03 debian dhcpd[2232]: Server starting service.
nov. 10 10:17:05 debian isc-dhcp-server[2216]: Starting ISC DHCPv4 server: dhcp
nov. 10 10:17:05 debian systemd[1]: Started LSB: DHCP server.
lines 1-17/17 (END)
```

Figure 12 : status du serveur DHCP ISC

Job08

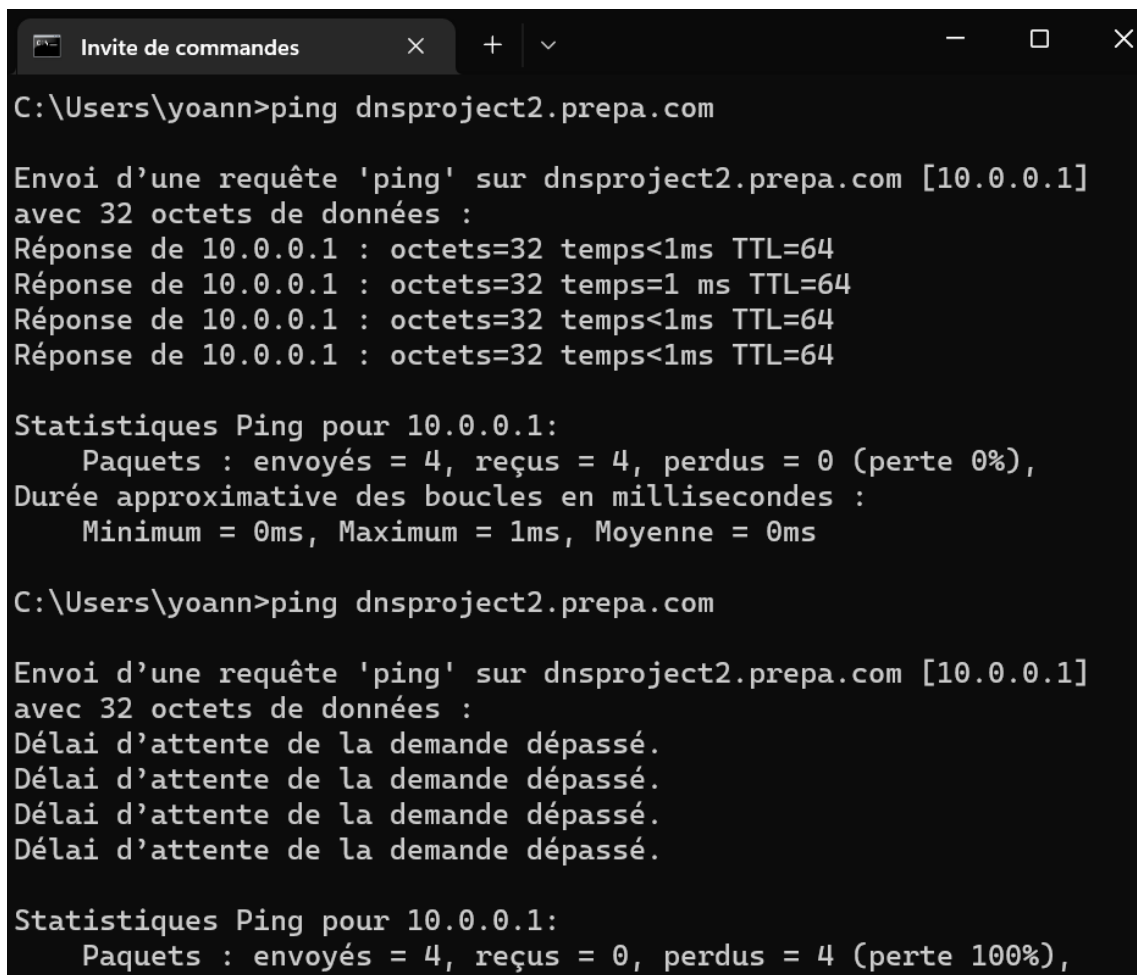
Faites en sorte que votre serveur principal serve de Gateway à vos autres machines virtuelles. Les autres machines ne pourront avoir internet qu'en passant par votre serveur principal.

N.B : déjà paramétré.

Job09

UFW, ou Uncomplicated Firewall, est une interface de gestion de pare-feu simplifiée qui masque la complexité des technologies de filtrage de paquets de niveau inférieur telles que **iptables** et **nftables**. Pour l'installation du paquet on entre la commande **sudo apt-get -y install ufw**, par défaut l'outil UFW n'est pas activé donc on entre la commande **sudo ufw enable**.

Afin de bloquer les requêtes de ping vers le serveur, il faut commenter la ligne **-A ufw-before-input -p icmp -icmp-type echo-request -j ACCEPT**, dans le fichier **/etc/ufw/before.rules**. De plus, notre hôte ne peut plus accéder à la page d'apache2 par défaut depuis l'activation du pare-feu, car il faut ouvrir le port de celui-ci, avec la commande **sudo ufw allow 80**. Pour valider toutes les modifications on entre **sudo ufw reload**, permettant de recharger UFW (la **figure 13** contient les requêtes de ping avant et après la modification du fichier **/etc/ufw/before.rules**).



```
Invite de commandes
C:\Users\yoann>ping dnsproject2.prepa.com

Envoi d'une requête 'ping' sur dnsproject2.prepa.com [10.0.0.1]
avec 32 octets de données :
Réponse de 10.0.0.1 : octets=32 temps<1ms TTL=64
Réponse de 10.0.0.1 : octets=32 temps=1 ms TTL=64
Réponse de 10.0.0.1 : octets=32 temps<1ms TTL=64
Réponse de 10.0.0.1 : octets=32 temps<1ms TTL=64

Statistiques Ping pour 10.0.0.1:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 0ms, Maximum = 1ms, Moyenne = 0ms

C:\Users\yoann>ping dnsproject2.prepa.com

Envoi d'une requête 'ping' sur dnsproject2.prepa.com [10.0.0.1]
avec 32 octets de données :
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.

Statistiques Ping pour 10.0.0.1:
    Paquets : envoyés = 4, reçus = 0, perdus = 4 (perte 100%),
```

Figure 13 : requêtes de ping avant et après la modification du fichier

Job10

Pour la réalisation de cet exercice, il faudra installer Samba. Pour cela on entre la ligne de commande suivante : `sudo apt-get -y install samba`. Après l'avoir installé, activez le en entrant `sudo systemctl -now smbd`.

Maintenant, nous allons permettre le partage, on entre la commande `sudo nano /etc/samba.smbd.conf` et on y ajoute les lignes suivantes :

```
[Public]  
path = /home/yoann/Public  
browsable = yes  
writable = yes  
read only = no  
force create mode = 0666  
force directory mode = 0777
```

Ensuite, pour enregistrer les modifications, il faudra redémarrer Samba, à l'aide de la commande `sudo systemctl restart smbd`. Il faudra ensuite créer un utilisateur nommé **guestshare**, nous devons activer ce compte pour Samba, à l'aide des commandes :

- `sudo smbpasswd -a guestshare`
- `sudo smbpasswd -e guestshare`

La première commande ajoute le compte, et la deuxième l'active.

Pour aller plus loin...

Le HTTPS, Hyper Text Transfer Protocol Secure, est un protocole qui permet d'encrypter les échanges entre le serveur et le navigateur web rendant donc impossible de récupérer les identifiants saisis. En HTTP, les identifiants passent en clair et sont donc plus faciles à intercepter. Cependant, il est nécessaire de créer un certificat SSL qui permettra d'initialiser les échanges cryptés.

Prérequis

Pour l'installation du paquet OpenSSL, on entre la commande `sudo apt-get -y install openssl`, permettant ainsi de générer les certificats.

Génération de la clé privée

On va créer une clé privée de 4096 bit encrypté avec l'algorithme de cryptage AES 256 bit, à l'aide de la commande : `sudo openssl genrsa -aes256 -out /etc/ssl/private/certificat.key 4096` (Il faut entrer un mot de passe pour la clé), on a donc généré une clé privée protégée par mot de passe.

Maintenant, il faut générer cette même clé mais déverrouillée. Dans un premier temps, on va renommer notre clé, à l'aide de la commande : `sudo mv /etc/ssl/private/certificat.key /etc/ssl/private/certificat.key.lock`.

Une fois la clé renommée, on génère notre certificat déverrouillé : `sudo openssl rsa -in /etc/ssl/private/certificat.key.lock -out /etc/ssl/private/certificat.key`.

- Clé privée déverrouillée : `certificat.key`.
- Clé privée verrouillée : `certificat.key.lock`.

```
yoann@debian:~$ sudo ls -l /etc/ssl/private/
total 12
-rw----- 1 root root 3243 9 nov. 09:20 certificat.key
-rw----- 1 root root 3326 9 nov. 09:19 certificat.key.lock
-rw-r----- 1 root ssl-cert 1708 8 nov. 18:13 ssl-cert-snakeoil.key
```

Figure 14 : création des `certificat.key`

Génération du fichier de demande de signature

Ce fichier va être utile pour obtenir notre certification de l'organisme ou pour auto-signer notre certificat. On entre donc la commande suivante : `sudo openssl -req -key /etc/ssl/private/certificat.key.lock -out /etc/ssl/certs/certificat.csr`.

- Fichier de demande de signature : `certificat.csr`.

Génération du certificat

Pour auto-signer notre certificat, on entre la commande suivante : `sudo openssl -x509 -req -days 365 -in /etc/ssl/certs/certificat.csr -signkey /etc/ssl/private/certificat.key.lock -out /etc/ssl/certs/certificat.crt`

```
yoann@debian:~$ sudo ls -l /etc/ssl/certs/
total 12
lrwxrwxrwx 1 root root 21 8 nov. 18:13 bdf9d908.0 -> ssl-cert-snakeoil.pem
-rw-r--r-- 1 root root 1818 9 nov. 09:24 certificat.crt
-rw-r--r-- 1 root root 1651 9 nov. 09:23 certificat.csr
-rw-r--r-- 1 root root 1038 8 nov. 18:13 ssl-cert-snakeoil.pem
```

Figure 15 : création des `certificat.crt` & `csr`

Configuration des fichier `/etc/apache2/`

Le fichier nous intéresse est `/etc/apache2/sites-available/default-ssl.conf`, donc on réalise une copie de ce fichier avec un nom adapté, pour cela on entre la commande suivante :

```
sudo cp /etc/apache2/sites-available/default-ssl.conf
/etc/apache2/sites-available/ssldnsproject2.prepa.com.conf.
```

Puis on modifie le fichier `ssldnsproject2.prepa.com.conf`, à l'aide de la commande `sudo nano /etc/apache2/sites-available/ssldnsproject2.prepa.com.conf`. Afin d'obtenir la configuration de la **figure 16** (faire attention à mettre les bon chemins pour les certificats SSL). Une fois les modifications apportées au fichier `^O^X`, puis on entre les commandes suivantes : `sudo a2ensite ssldnsproject2.prepa.com`, permettant d'activer un site **apache2**, `sudo a2enmod ssl` pour activer le mode SSL, ainsi que la commande `sudo systemctl restart apache2`.

```
/etc/apache2/sites-available/ssldnsproject2.prepa.com.conf *
<IfModule mod_ssl.c>
    <VirtualHost *:443>

        ServerName dnsproject2.prepa.com
        ServerAlias www.dnsproject2.prepa.com

        DocumentRoot /var/www/html

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

        SSLEngine on

        SSLCertificateFile      /etc/ssl/certs/certificat.crt
        SSLCertificateKeyFile   /etc/ssl/private/certificat.key

        SSLProtocol all -SSLv2 -SSLv3
        SSLHonorCipherOrder on
```

Figure 16 : fichier de configuration d'apache2 du dns2 avec SSL

Configuration des fichier `/etc/bind/`

Comme pour le Job06 le fichier qui nous intéresse est le `/etc/bind/db.empty`. Cependant seul une modification est nécessaire, donc on va copier le fichier `/etc/bind/dnsproject1.prepa.com` avec la commande : `sudo cp /etc/bind/dnsproject1.prepa.com /etc/bind/dnsproject2.prepa.com`

Puis on modifie le fichier `dnsproject2.prepa.com`, à l'aide de la commande `sudo nano /etc/bind/dnsproject2.prepa.com`. Afin d'obtenir la configuration de la **figure 17**. Une fois les modifications apportées au fichier `^O^X`, puis on entre la commande suivante : `sudo systemctl restart bind9`.

```
GNU nano 5.4 /etc/bind/dnsproject2.prepa.com *
; BIND reverse data file for empty rfc1918 zone
;
; DO NOT EDIT THIS FILE - it is used for multiple zones.
; Instead, copy it, edit named.conf, and use that copy.
;
$TTL      86400
@         IN      SOA      dnsproject2.prepa.com. root.dnsproject2.prepa.com. (
                                200          ; Serial
                                604800       ; Refresh
                                86400        ; Retry
                                2419200     ; Expire
                                86400 )     ; Negative Cache TTL
;
@         IN      NS       ns1.dnsproject2.prepa.com.
@         IN      A        10.0.0.1
ns1       IN      A        10.0.0.1
www       IN      A        10.0.0.1
```

Figure 17 : fichier de configuration de bind du dns2

Ensuite, il faut éditer le fichier `/etc/bind/named.conf.local`, ce fichier contient la configuration locale du serveur DNS, on y déclare les zones associées au domaine. Dans notre cas, il faut y déclarer une nouvelle zone, celle de `dnsproject2.prepa.com`, voir **figure 18**. Une fois toutes les configurations faites, on doit recharger chaque fichier donc on entre les commandes suivantes :

- `sudo systemctl restart apache2`, redémarrage d'`apache2`.
- `sudo systemctl restart bind9`, redémarrage de `bind9`.

```
GNU nano 5.4 /etc/bind/named.conf.local *
//
// Do any local configuration here
//
// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "dnsproject1.prepa.com"{
    type master;
    file "/etc/bind/dnsproject1.prepa.com";
};

zone "dnsproject2.prepa.com"{
    type master;
    file "/etc/bind/dnsproject2.prepa.com";
};
```

Figure 18 : fichier de configuration de bind zone du dns2

Configuration de l'hôte

Il faudra se rendre sur son OS windows et taper dans le menu démarrer : *panneau de configuration - réseau et internet - centre réseau et partage - modifier les paramètre de la carte* (comme je suis sur vmware), prendre *VMware Network Adapter VMnet8 - propriété - protocole internet version 4* et mettre les adresse adapté à votre configuration, voir **figure 19**, **N.B** : cette configuration est adapté à mes paramètres.

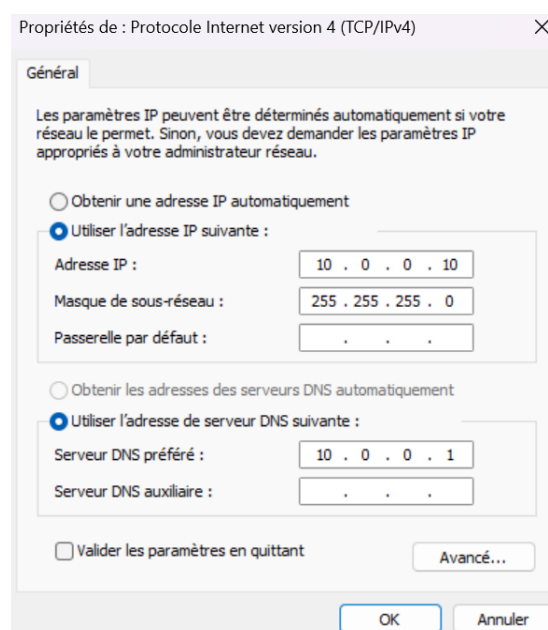


Figure 19 : configuration réseau windows

Maintenant, on a bien notre site internet certifié **HTTPS://**, avec un DNS fonctionnel, voir **figure 20**. Cependant, notre **HTTPS://** est fonctionnel mais non reconnu, car pour cela il faut payer un organisme.

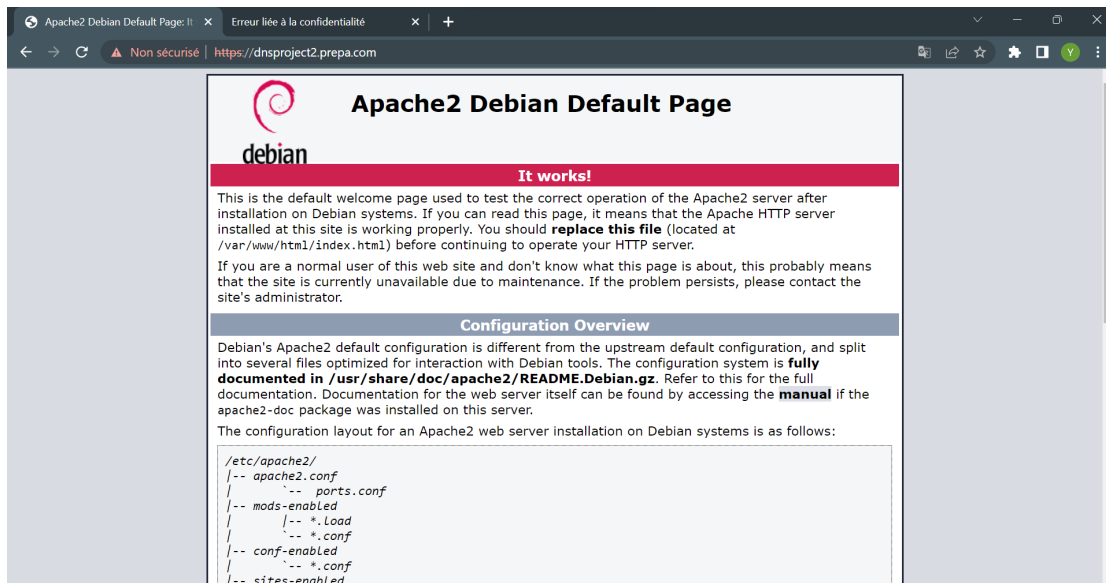


Figure 20 : page d'apache2 en HTTPS

Questions

Renseignez-vous aussi sur la différence entre les certificats SSL donnés par des organismes extérieurs et le vôtre auto-signé ? Pourquoi votre certificat apparaît-il comme non sécurisé dans votre navigateur ?

Pour être reconnu par un navigateur web, le certificat SSL doit être signé par un organisme de certification. Ces organismes font payer cette certification.

Avec un certificat auto-signé une page d'alerte apparaît dans notre navigateur, dans le cas d'un site en local, ce n'est pas grave, il suffit de passer cette alerte pour pouvoir consulter notre site en toute sécurité car la couche SSL qui crypte les données fait quand même son travail.

Par contre, si un hacker venait à prendre le contrôle de notre serveur et modifiait notre certificat SSL, il serait alors en mesure de décrypter les échanges.

Le cas d'un certificat auto-signé peut être intéressant pour la gestion d'un intranet. L'idée serait de générer un certificat maître auto-signé qui signerait ensuite lui-même tous les certificats SSL de l'ensemble des services utilisés dans l'intranet. C'est ce qu'on appelle une PKI (Public Key Infrastructure).

On peut ensuite installer le certificat maître sur tous les postes des utilisateurs de l'intranet. Ces derniers pourront alors consulter tous les sites en HTTPS sans avoir d'alerte de sécurité et l'administrateur de l'intranet garde la main pour pouvoir ajouter de nouveaux certificats valides.

Annexes

Tables des Figures

Figure 1 : page de lancement d'Apache.....	2
Figure 2 : configuration pour une IP static.....	2
Figure 3 : page de lancement d'Apache avec un DNS.....	5
Figure 4 : ping du DNS de la page d'apache2.....	5
Figure 5 : fichier de configuration d'apache2 du dns1.....	7
Figure 6 : fichier de configuration de bind du dns1.....	8
Figure 7 : fichier de configuration de bind zone du dns1.....	9
Figure 8 : configuration réseau windows.....	9
Figure 9 : fichier resolv.conf.....	10
Figure 10 : vérification nslookup www.....	10
Figure 11 : fichier dhcpd.conf.....	10
Figure 12 : status du serveur DHCP ISC.....	11
Figure 13 : requêtes de ping avant et après la modification du fichier.....	12
Figure 14 : création des certificat.key.....	14
Figure 15 : création des certificat.crt & csr.....	14
Figure 16 : fichier de configuration d'apache2 du dns2 avec SSL.....	15
Figure 17 : fichier de configuration de bind du dns2.....	15
Figure 18 : fichier de configuration de bind zone du dns2.....	16
Figure 19 : configuration réseau windows.....	16
Figure 20 : page d'apache2 en HTTPS.....	17