
DDWS :

DHCP DNS Web-server Samba

Sujet 1 (Job3) : Serveurs Web

Configuration du DNS

Sujet 2 (Job 5) : Nom de domaine

Configuration du DNS accessible par l'hôte

Mise en place du sous-réseau de VMs

Paramétrage pare-feu

Dossier partagé entre VMs via Samba

Sujet 3 (Aller plus loin) : Connexion sécurisé

PELAYO Joël

Sujet 1 (Job 3) : Serveurs Web

On désigne un serveur web à la base comme étant l'appareil possédant différent contenu et services accessible à tous les clients connectés à son réseau. Il existe différent serveurs web présentant chacun des avantages comme des inconvénients et adaptées à tel ou tel utilisation.



Apache Web Server : Développé par Apache Software Foundation, il s'agit d'un des serveurs les plus populaires aujourd'hui. Etant open source, il est supporté sur presque tous les systèmes d'exploitation et est facile à installer et à personnaliser selon nos besoins. Apache traite son trafic et ses requêtes via le multi-thread ce qui peut parfois être une perte de performance sur le serveur lorsqu'il est chargé.



NGINX Web Server : Est un grand concurrent à Apache étant tout aussi populaire, mais sera moins personnalisable. Il est également supporté sur tout OS et Windows partiellement. Il traite ses requêtes sous forme d'événements, il utilisera qu'un seul thread pour plusieurs requêtes.



Lighttpd : Il s'agit d'un serveur Web open source léger pensé pour des systèmes avec peu de ressources ou qui souhaite une solution alliant performances et légèreté. La légèreté du serveur vient avec certains compromis, moins de personnalisation, moins stable, ne tiendra pas face à des sites web de grandes envergures.



LiteSpeed : Il est connu comme un serveur web très performant, relativement léger et compatible avec Apache dans le sens où passer de l'un à l'autre est simple. Traite ses requêtes comme NGINX mais reste personnalisable surtout au niveau de sa sécurité. Il possède une version open source mais qui ne sera pas compatible avec Apache, il ne supporte pas Windows également.

Il en existe encore beaucoup, mais ce qu'on retiendra c'est que le choix se fait surtout sur nos besoins et donc : performance, compatibilité avec des OSs, personnalisation/open source seront des critères auxquelles il faudra attention.

Configuration du DNS (Job 4):

1- Premièrement il faudra installer un serveur DNS. On utilisera bind9 :

sudo apt -y install bind9

2- Il faudra ensuite créer/configurer différent fichier afin d'obtenir un DNS (à noter que les configurations doivent correspondre à notre adresse ip).

```
debugger@debian:/etc$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:c9:78:38 brd ff:ff:ff:ff:ff:ff
    inet 10.10.31.114/16 brd 10.10.255.255 scope global dynamic enp0s3
        valid_lft 550sec preferred_lft 550sec
    inet6 fe80::a00:27ff:fe9:7838/64 scope link
        valid_lft forever preferred_lft forever
```

Les fichiers à configurer seront :

- **/etc/resolv.conf**

```
debugger@debian:/etc$ nano resolv.conf
# Generated by NetworkManager
nameserver 10.10.31.114
nameserver 10.10.9.1
```

- **/etc/bind/db.dnsproject** (à créer, le nom n'est pas important)

```
debugger@debian:/etc/bind$ nano db.dnsproject
; BIND data file for local loopback interface
$TTL 604800
@ IN SOA prepa.com. root.dnsproject.prepa.com. (
    2           ; Serial
    604800      ; Refresh
    86400       ; Retry
    2419200    ; Expire
    604800     ; Negative Cache TTL
)
;
; IN NS dnsproject.prepa.com.
dnsproject IN A 10.10.31.114
114 IN CNAME dnsproject.prepa.com.
```

- **/etc/bind/db.dnsproject-inverse** (à créer)

```
debugger@debian:/etc/bind$ nano db.dnsproject-inverse
; BIND data file for local loopback interface
$TTL 604800
@ IN SOA prepa.com. root.dnsproject.prepa.com. (
    2           ; Serial
    604800      ; Refresh
    86400       ; Retry
    2419200    ; Expire
    604800     ; Negative Cache TTL
)
;
; IN NS dnsproject.prepa.com.
dnsproject IN A 10.10.31.114
www IN PTR dnsproject.prepa.com.
```

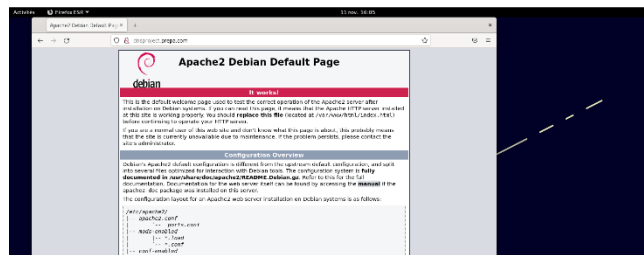
- **/etc/bind/named.conf.local**

```
debugger@debian:/etc/bind$ nano 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 "prepa.com" IN {
    type master;
    file "/etc/bind/db.dnsproject";
};

zone "31.10.10.in-addr.arpa" IN {
    type master;
    file "/etc/bind/db.dnsproject-inverse";
};
```

- 3- La page par défaut du serveur apache devrait être accessible via le dns 'dnsproject.prepa.com'



Sujet 2 (Job 5) : Nom de domaine

Le DNS est un système permettant la traduction de noms de domaine en adresses ip, étant donné que les machines se reconnaissent entre elles que via des IPs, cela facilite leurs identifications pour l'homme. On peut distinguer aujourd'hui les noms de domaine privées et publiques.

Le nom de domaine privée correspondra à un nom de domaine local ou coupé d'internet il peut prendre le nom qu'il souhaite. Le nom public lui en revanche est limitée et présent sur internet, en effet différent sites web ne peuvent pas avoir le même nom sur internet.



Afin d'obtenir un nom de domaine publique il faudra le 'réserver' et ce service est souvent (si ce n'est tout le temps) payant selon le nom qu'on veut mettre. Mais il existe toutefois de nombreuses options en ce qui concerne l'obtention d'un nom de domaine public pour moins cher, donc en incluant d'autres services web (comme du web-hosting, mail hosting, website builder, ou autres) ou via un paiement mensuel, ces services sont souvent proposés par IONOS, WordPress, Wix, et d'autres encore.

Au-delà du nom de domaine il y a aussi l'extension du nom de domaine ; .com, .net, .org, etc. Les extensions aujourd'hui n'ont plus vraiment d'importance, aujourd'hui ils ont plus un intérêt marketing et de visibilité sur un moteur de recherche. Toutefois ils peuvent tout de même être catégorisé selon l'extension utilisé (il s'agit plus d'une convention qu'autre chose) :

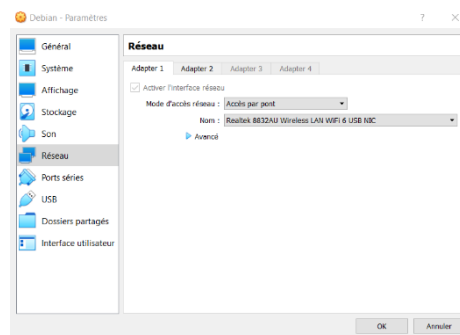
- extension les plus utilisés (ou générique) : .com, .net, .org, .us, .co
- ccTLD (Country-code top-level domain) extensions basées sur le pays : .fr, .uk, .us, .au, etc.
- sTLD (Sponsored TLD) basées sur une communauté ou un organismes particulier : .gov, .post,
- gTLD (Generic TLD) : extensions sans grande représentations, elles peuvent prendre n'importe quelle forme.



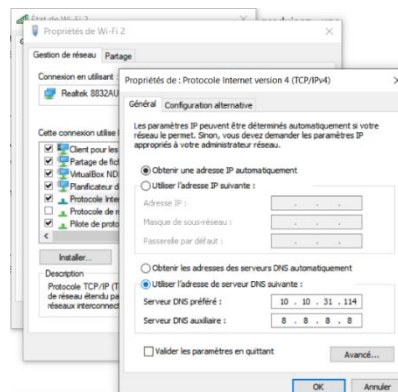
Configuration du DNS accessible par l'hôte (Job 6):

Afin que l'hôte puisse accéder à la page via le DNS, il faudra configurer notre VM serveur en accès par pont via notre hyperviseur et configurer le serveur DNS préféré de l'hôte.

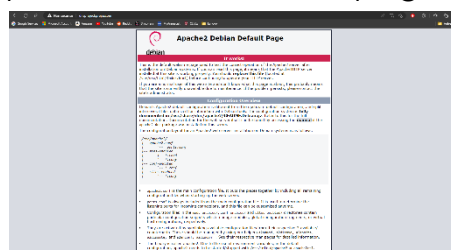
- 1- En effet la VM doit être en bridge afin que la VM soit considérée comme une machine à part entière sur le même réseau que celui de l'hôte pour qu'il puisse détecter notre serveur.



- 2- Le serveur DNS préféré doit être celui du serveur, l'hôte alors cherchera à résoudre le DNS de cette ip.



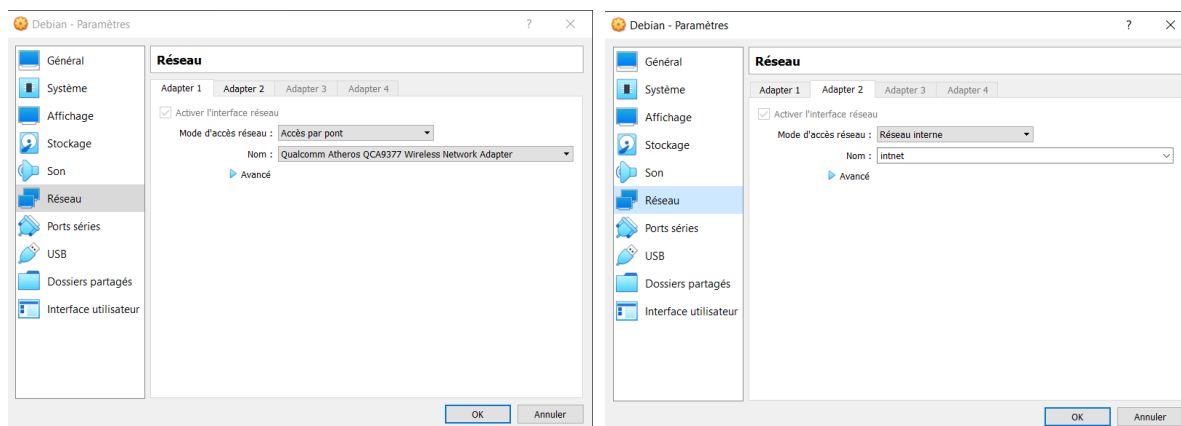
L'hôte devrait être capable d'accéder à la page via le DNS.



Mise en place du sous-réseau de VMs (Job 7-8) :

La VM possède différentes options pour créer un sous-réseau de VMs, une des manières qui sera du coup celle que j'ai choisi est celle de devoir ajouté une interface réseau à la VM pour connecter le serveur au sous-réseau.

Donc via Vbox on pourra posséder jusqu'à 4 interfaces pour une VM :



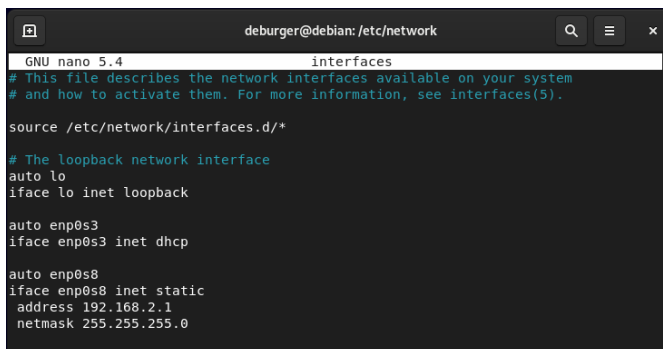
L'idée serait d'avoir une interface en bridge pour l'accès à internet et la deuxième en interne qui connectera le serveur au sous-réseau (le Job 7-8 à été fait à domicile pour cause de problème d'accès à internet en bridge à l'école, cela justifiera les IPs différentes des précédentes sur les screens).

DHCP : Pour faire fonctionner le DHCP on se servira du paquet présent par défaut et donc à installer étant '**isc-dhcp-server**'

- 1- Installation du paquet avec : **sudo apt -y install isc-dhcp-server**
- 2- On va configurer une adresse statique sur l'interface paramétré en réseau interne (enp0s8 ci-dessous), cela sera utile également pour la Gateway.

```
deburger@debian: /
deburger@debian:/etc/network$ cd /
deburger@debian:/$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
    link/ether 08:00:27:c9:78:38 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.14/24 brd 192.168.1.255 scope global dynamic enp0s3
        valid_lft 40959sec preferred_lft 40959sec
    inet6 2a01:e0a:8ae:2330:a00:27ff:fec9:7838/64 scope global dynamic mngtmpadd
        valid_lft 86160sec preferred_lft 86160sec
    inet6 fe80::a00:27ff:fec9:7838/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
    link/ether 08:00:27:ac:47:c2 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.1/24 brd 192.168.2.255 scope global enp0s8
        valid_lft forever preferred_lft forever
```

L'adressage d'une ip statique se fait dans le fichier **/etc/network/interfaces**



```
deburer@debian: /etc/network
GNU nano 5.4 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 enp0s3
iface enp0s3 inet dhcp

auto enp0s8
iface enp0s8 inet static
    address 192.168.2.1
    netmask 255.255.255.0
```

- 3- Il faut mentionner sur quel interface (quel réseau du coup) le serveur DHCP doit attribuer les IPs qu'il générera dans le fichier **/etc/default/isc-dhcp-server**.



```
deburer@debian: /etc/default
GNU nano 5.4 isc-dhcp-server
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp-server)

# Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

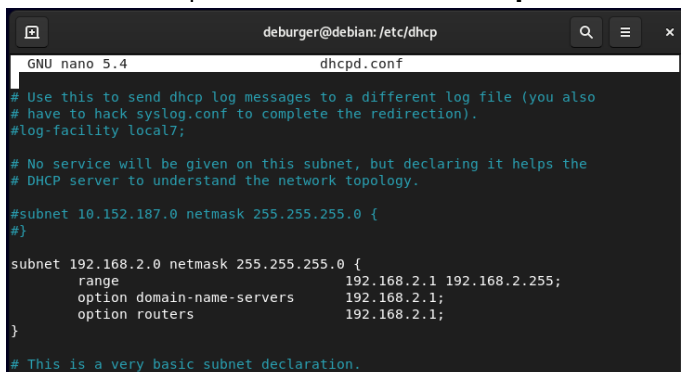
# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPDv4_PID=/var/run/dhcpd.pid
#DHCPDv6_PID=/var/run/dhcpd6.pid

# Additional options to start dhcpd with.
# Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
# Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="enp0s8"
INTERFACESv6=""
```

- On décommente **DHCPDv4_CONF=/etc/dhcp/dhcpd.conf**
- On mentionne du coup dans **INTERFACESv4** l'interface où le dhcp sera effectué (enp0s8 dans mon cas).

- 4- Configurons maintenant le fichier de configuration du dhcp mentionné précédemment, **dhcpd.conf**.



```
deburer@debian: /etc/dhcp
GNU nano 5.4 dhcpd.conf
# Use this to send dhcp log messages to a different log file (you also
# have to hack syslog.conf to complete the redirection).
#log-facility local7;

# No service will be given on this subnet, but declaring it helps the
# DHCP server to understand the network topology.

#subnet 10.152.187.0 netmask 255.255.255.0 {
#}

subnet 192.168.2.0 netmask 255.255.255.0 {
    range                  192.168.2.1 192.168.2.255;
    option domain-name-servers 192.168.2.1;
    option routers          192.168.2.1;
}

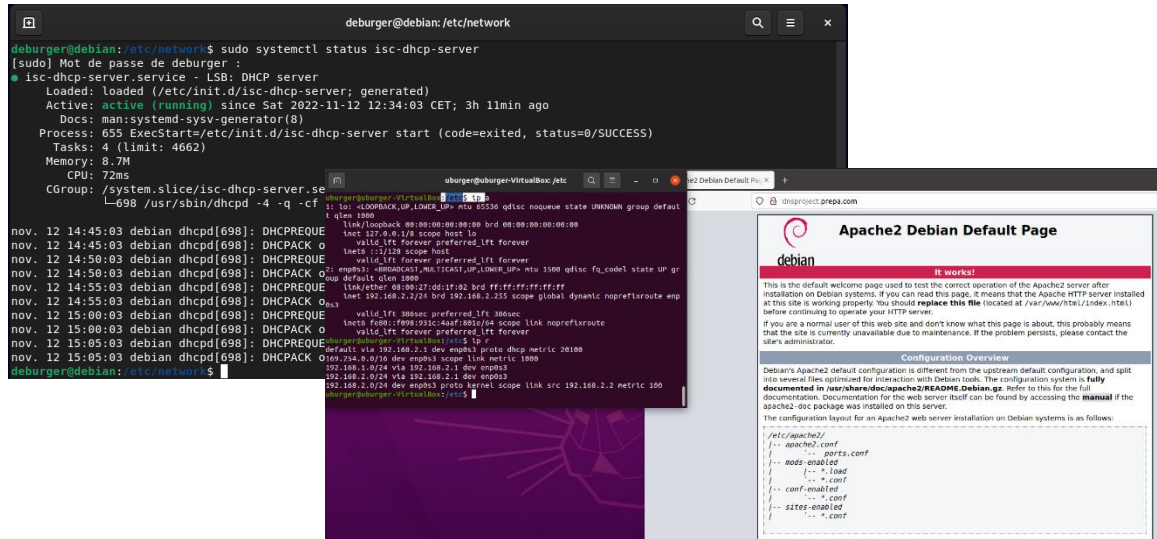
# This is a very basic subnet declaration.
```

Notre sous-réseau sera défini en 192.168.2.0/24 avec une range total de 1 à 255, excluant bien sûr le 1 qui sera le routeur, en

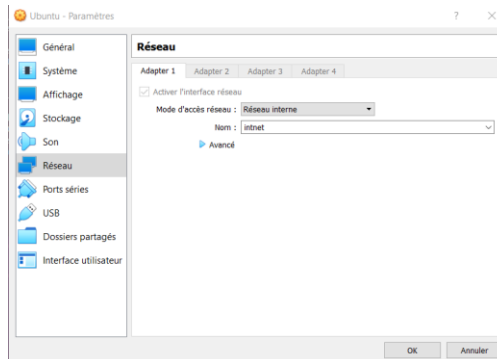
l'occurrence notre serveur qui servira Gateway pour le sous-réseau, et le 255 étant le broadcast.

- 5- Enfin, le dhcp devrait être fonctionnelle après un démarrage ou redémarrage du serveur et il devrait être actif :

sudo systemctl restart isc-dhcp-server



Comme on peut le voir, le serveur dhcp est bien actif et le client (qui est sur ubuntu) a bien une ip correspondant à la range défini sur le serveur et à accès à la page apache via le dns. A noter que le client doit être configuré en réseau interne sur le même réseau (voir image ci-dessous).



Gateway : Il s'agira de configurer un routage entre le sous-réseau le vrai réseau donc dans notre cas :

- **sudo ip route add 192.168.1.0/24 via 192.168.2.1 dev enp0s8**
- **sudo ip route add 192.168.2.0/24 via 192.168.1.14 dev enp0s8**

Paramétrage du pare-feu (Job 9) :

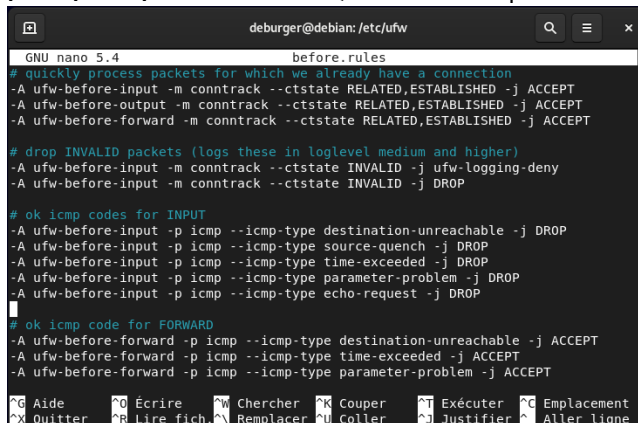
L'utilisation d'un pare-feu nous permettra de sécuriser l'accès à notre serveur, l'idée ici serait de limité l'accès au serveur qu'à la page web.

- 1- On va se servir d'ufw qui devrait normalement déjà être installé, si ce n'est pas le cas on peut toujours l'installer via apt.

On peut aussi si ce n'est pas fait, activer le pare-feu via :

sudo ufw enable

- 2- Il va falloir configurer un fichier de configuration d'ufw étant **/etc/ufw/before.rules**, afin d'empêcher le ping sur notre serveur.



```
deburger@debian: /etc/ufw
GNU nano 5.4 before.rules
# quickly process packets for which we already have a connection
-A ufw-before-input -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-output -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-forward -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

# drop INVALID packets (logs these in loglevel medium and higher)
-A ufw-before-input -m conntrack --ctstate INVALID -j ufw-logging-deny
-A ufw-before-input -m conntrack --ctstate INVALID -j DROP

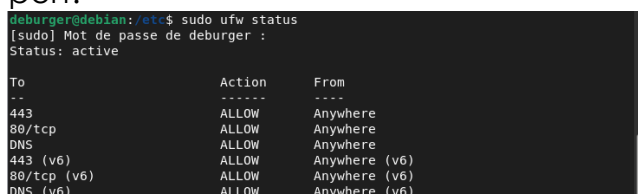
# ok icmp codes for INPUT
-A ufw-before-input -p icmp --icmp-type destination-unreachable -j DROP
-A ufw-before-input -p icmp --icmp-type source-quench -j DROP
-A ufw-before-input -p icmp --icmp-type time-exceeded -j DROP
-A ufw-before-input -p icmp --icmp-type parameter-problem -j DROP
-A ufw-before-input -p icmp --icmp-type echo-request -j DROP

# ok icmp code for FORWARD
-A ufw-before-forward -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type parameter-problem -j ACCEPT

^G Aide      ^O Écrire   ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^N Remplacer  ^U Coller    ^J Justifier  ^_ Aller ligne
```

Les 5 lignes après '# ok icmp codes for INPUT' était de base en 'ACCEPT', il faut les remplacés par 'DROP'

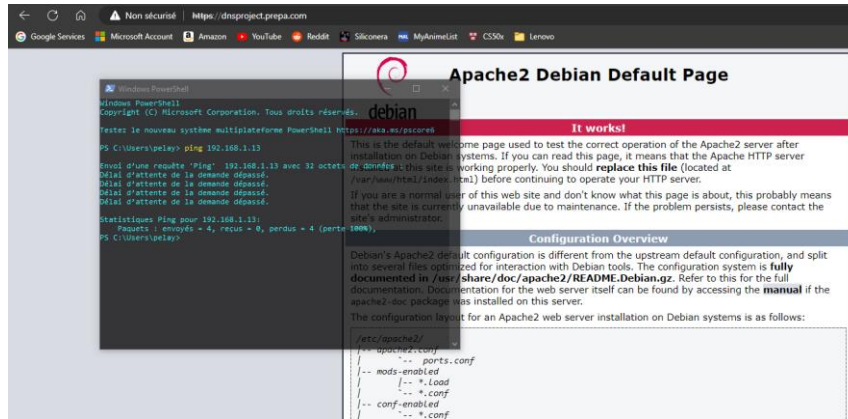
- 3- Il est possible aussi d'ouvrir/fermer certains ports via la commande **ufw allow <port>**, il est possible aussi de spécifier seulement le nom du port.



```
deburger@debian: /etc$ sudo ufw status
[sudo] Mot de passe de deburger :
Status: active

To Action From
--
443 ALLOW Anywhere
80/tcp ALLOW Anywhere
DNS ALLOW Anywhere
443 (v6) ALLOW Anywhere (v6)
80/tcp (v6) ALLOW Anywhere (v6)
DNS (v6) ALLOW Anywhere (v6)
```

Après un **sudo ufw reload**, le pare-feu devrait être actif et empêcher le ping vers notre serveur mais pas l'accès à la page apache.



Dossier partagé entre VMs via Samba (Job 10)

- 1- Installation de samba : **sudo apt -y install samba smbclient cifs-utils**
- 2- On créera les dossiers partagés où on le souhaitera, puis on modifiera le fichier de conf de samba pour leurs définir des spécificités.

```
deburger@debian: /etc/samba
GNU nano 5.4      smb.conf

[public]
comment = Public Folder
path = /public
writable = yes
guest ok = yes
guest only = yes
force create mode = 775
force directory mode = 775

[private]
comment = Private Folder
path = /private
writable = yes
guest ok = yes
valid users = @smbshare
force create mode = 770
force directory mode = 770
inherit permissions = yes
```

On rajoutera ces deux segments (un pour chaque dossier créer) parmi les valeurs les plus intéressantes on a :

- comment : est juste un commentaire
- path : il faudra mentionner le chemin du dossier partagé
- writable : défini si le dossier peut être altéré (rajouter/supprimer du contenu)
- valid users : défini les utilisateurs qui auront accès au dossier

- 3- Créons un groupe qui aura accès aux dossiers qui seront partagés :

- **sudo groupadd smbshare**
- **sudo chgrp -R smbshare 'chemin du dossier'**
- **sudo chmod 2775 'chemin du dossier'**

On restreint l'accès à ce groupe

- **sudo useradd -M -s /sbin/nologin sambauser**
- **sudo useradd -aG smbshare sambauser**

On crée un utilisateur local qui n'aura pas besoin de login pour les dossiers en **force create/directory mode 770** et on le rajoute dans le groupe

- **sudo smbpasswd -a sambauser**
- **sudo smbpasswd -e sambauser**

On crée un mdp samba pour ensuite activer l'utilisateur

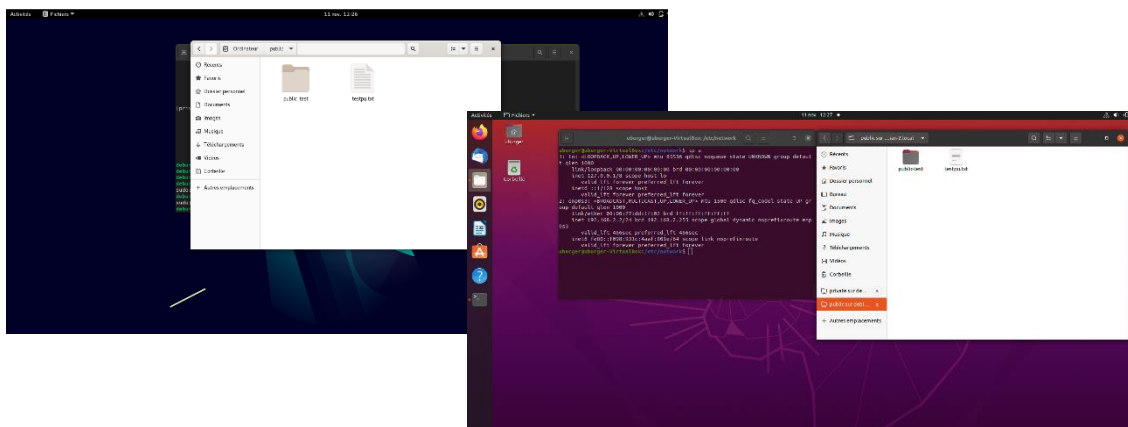
- 4- Si on possède un pare-feu il faudra autoriser l'accès à distance au moins sur notre sous-réseau de VM, dans notre cas ce sera le réseau 192.168.2.0/24 : **sudo ufw allow from 192.168.2.0/24 to any app Samba**
Il sera alors temps de redémarrer Samba et donc de configurer les clients pour y avoir accès.

- **sudo testparm** (permet une vérification des paramètres samba)
- **sudo systemctl restart nmbd**

- 5- Mon client sur mon sous-réseau est un linux (ubuntu), il faudra alors lui installer premièrement le client samba :

sudo apt -y install samba-client cifs-utils

A partir de là, les dossiers créés et configurés devraient être accessibles au client via le gestionnaire de fichiers dans 'Autres emplacements', puis en se connectant avec l'utilisateur créé précédemment si on a un dossier en **force create/directory mode 770**, Sinon il devrait être possible d'y accéder en anonyme sans problème.



Sujet 3 (Aller Plus Loin) : Connexion sécurisé

Afin d'obtenir une connexion à notre page apache en https il faudra configurer une connexion en SSL/TLS, pour cela on devra générer une clé/certificat et faire en sorte que l'échange entre un client du serveur et le serveur lui-même soit sécurisé par cette certification.

Pour y parvenir on va se servir d'openssl afin de générer notre certificat gratuitement puis configurer apache2 pour qu'il se serve de notre certificat pour permettre une connexion sécurisée au serveur et donc à la page.

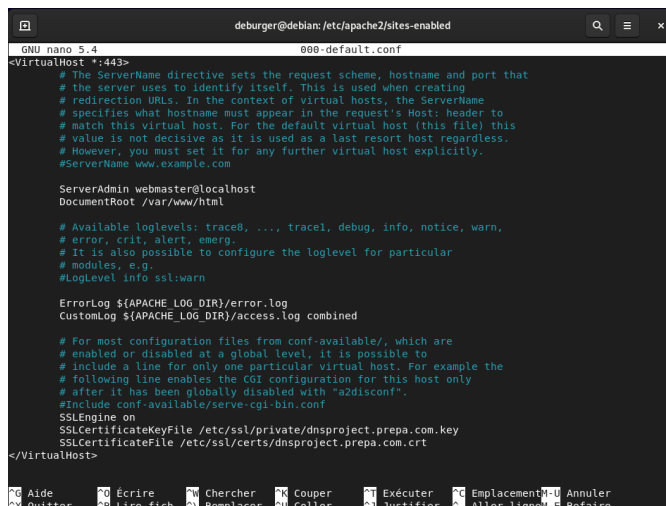
- 1- Premièrement on se connecte à notre serveur sur le terminal via ssh :
sudo ssh 'utilisateur@<ip> -p 22'

On active ensuite le module ssl d'apache2 :
sudo a2enmod ssl

- 2- On génère la clé et le certificat via openssl (le nom de la clé et du certificat n'est pas important mais il sera conseillé de les faire tout de même correspondre avec le nom du dns) :

**sudo openssl req req -x509 -nodes -days 365 -newkey rsa:2048 **
**-keyout /etc/ssl/private/dnsproject.prepa.com.key **
-out /etc/ssl/certs/dnsproject.prepa.com.crt

- 3- On configure apache2 afin que l'accès à la page puisse se faire via une connexion sécurisé grâce au certificat qu'on vient de générer. Donc dans le dossier /etc/apache2/sites-available on peut configurer un fichier existant ou alors en créer un (j'ai modifié un fichier existant '000-default.conf').



```
debuger@debian: /etc/apache2/sites-enabled
GNU nano 5.4 000-default.conf
<VirtualHost *:443>
# 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
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

SSLEngine on
SSLCertificateKeyFile /etc/ssl/private/dnsproject.prepa.com.key
SSLCertificateFile /etc/ssl/certs/dnsproject.prepa.com.crt
</VirtualHost>

Aide  Écrire  Chercher  Couper  Exécuter  Emplacement  Annuler
Quitter Lire fich. Remplacer Coller Justifier Aller ligne Refaire
```

Il faudra alors comme on le voit sur l'image ci-dessus :

- Changer le port à utiliser : **<VirtualHost *:443>**
- Rajouter le fait qu'on se servira du certificat créé précédemment :

- **SSLEngine on**
- **SSLCertificateKeyFile 'chemin vers la clé SSL'**
- **SSLCertificateFile 'chemin vers le certificat'**

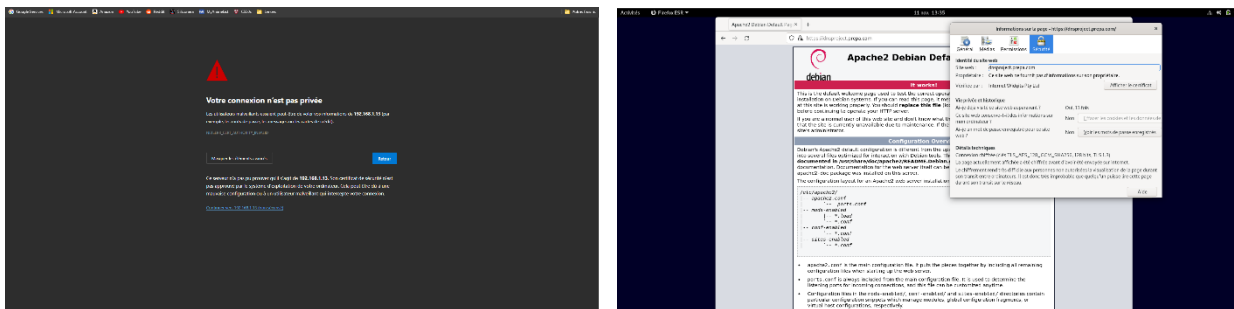
- 4- On peut maintenant activer la configuration (si ce n'est pas déjà fait, notamment si c'est un nouveau fichier) avec :

sudo a2ensite 'nom du fichier configuré (sans le chemin)'

Il faudra alors redémarrer le service apache2 avec :

sudo service apache2 restart

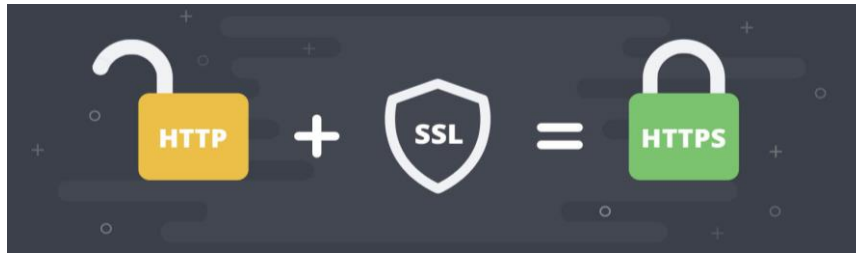
A partir de là, il devrait être possible d'accéder à la page en https, le moteur de recherche nous avertira pour faire simple que le certificat pour la connexion en https n'est pas fiable, mais il sera tout de même possible d'accéder à la page en continuant.



Connexion https / certificats SSL :

La connexion https se sert du protocole SSL/TLS pour crypter les échanges entre le serveur et les clients, le protocole se sert de certificat attribué aux clients qui sert de clé publique et d'une clé privée gardé par le serveur.

Or on a pu voir qu'il était possible de fabriquer son propre certificat/clé, néanmoins ils existent des organisations spécialisées dans la création et attribution de certificats SSL à plusieurs sites web alors qu'est-ce qui les différencie de nos certificats ?



La différence vient principalement du fait que le certificat n'est reconnu par aucune autorité de certificats, en effet ils existent des organismes (comme Cloudflare, DigiCert, etc.) chargées de valider les certificats produits par différentes entités, leurs rôles est de tester la sécurité des ces certificats (ex : méthode de cryptographie, ou autres), le type de certificat et de s'assurer qu'ils soient peu vulnérables.

La plupart des organisations produisant ces certificats font payer leurs certificats selon leurs niveaux de validation.

La raison pour laquelle notre certificat apparaît comme non-sécurisé vient tout simplement du fait que le moteur de recherche ne reconnaît aucun niveau de validation sur notre certificat et donc nous signal que c'est une connexion non-sécurisé car le certificat n'est pas reconnu ou validé donc non-viable.