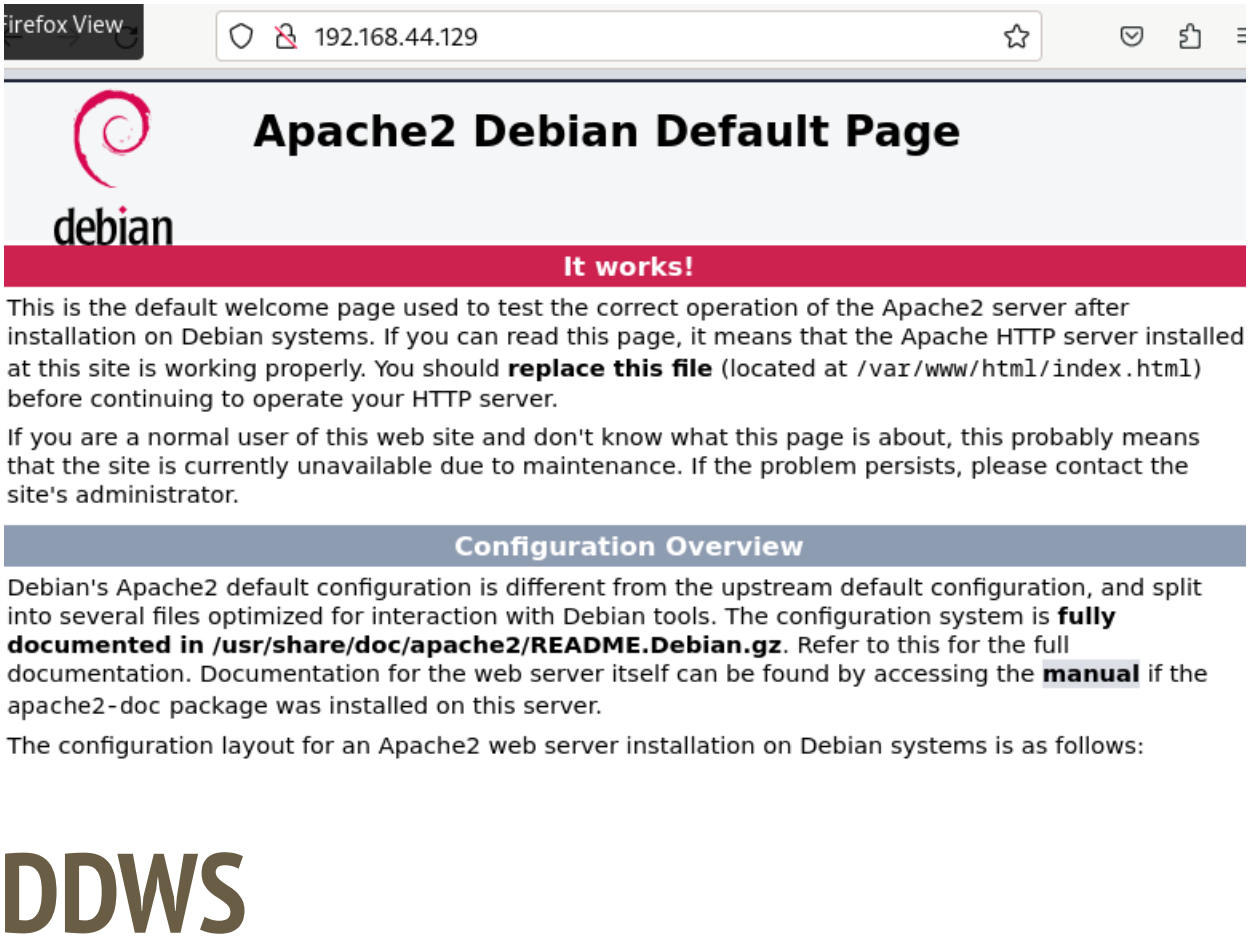


DDWS

25.10.2023

—

OroitzLR
Oroitz Lago Ramos

Vue d'ensemble

L'objectif est de mettre en place un serveur principal regroupant divers services réseau.

Job 01:

Installation VM:

I. Installation

Afin de créer un serveur **Apache2** nous allons installer une machine virtuelle debian 12 avec interface graphique.

Je vais utiliser **VMWare Workstation Pro**.

II. Accès SSH

Soit nous configurons notre serveur web depuis la VM soit nous pouvons nous connecter en **SSH** depuis notre machine hôte.

Dans mon cas, j'ai choisi de le faire directement depuis ma machine virtuelle.

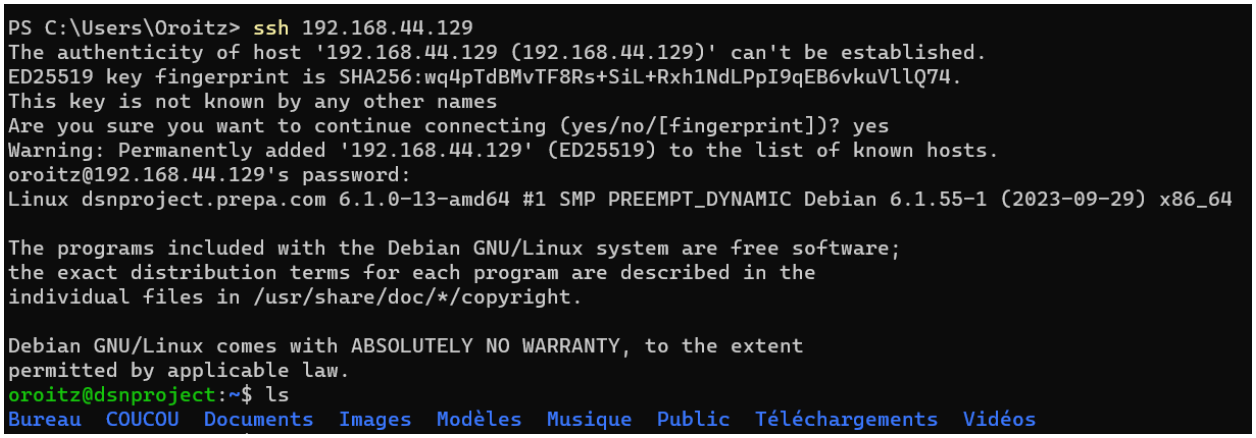
Néanmoins configurons et connectons nous en SSH.

Tout d'abord je vais vérifier que le SSH soit bien allumé et actif pour cela nous pouvons utiliser la commande **systemctl status ssh**.

```
oroitz@dsnproject:~$ systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Thu 2023-10-26 08:30:53 CEST; 24min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 791 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 817 (sshd)
      Tasks: 1 (limit: 4590)
     Memory: 3.1M
        CPU: 31ms
     CGroup: /system.slice/ssh.service
             └─817 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
```

Une fois cela vérifié sur notre machine hôte nous entrons la commande suivante **ssh 192.168.44.129**, qui est l'ip de ma machine virtuelle.

Je rentre le mot de passe et nous y sommes.



Job 02:

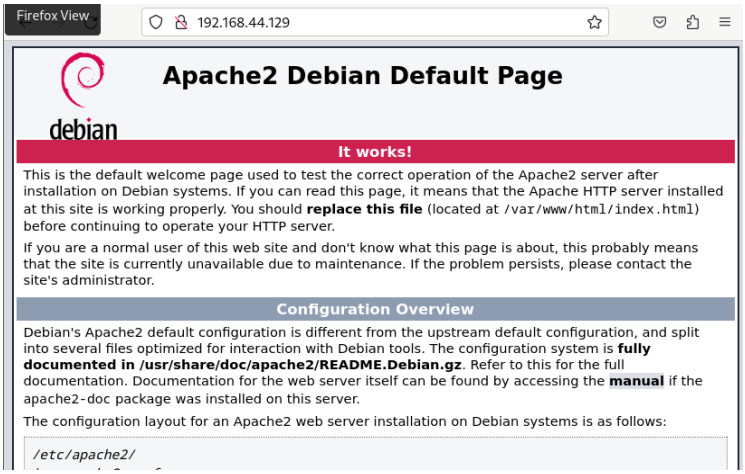
Installation serveur Apache:

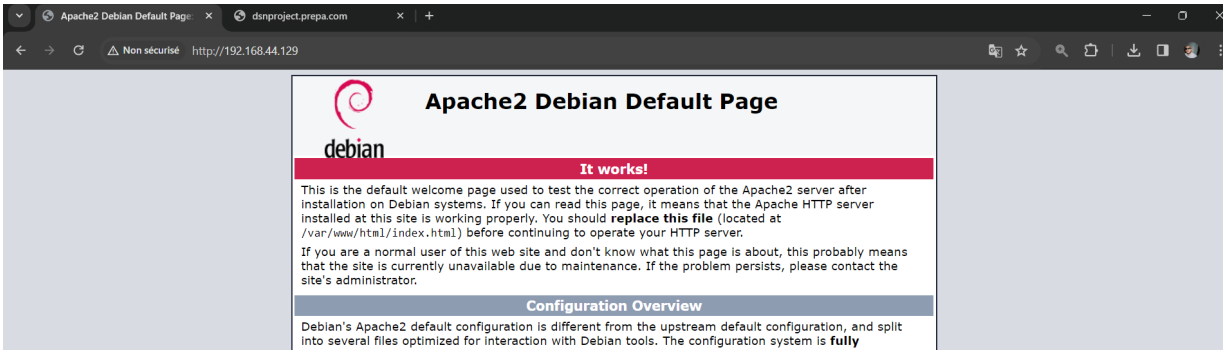
I. Apache2

Afin d'installer Apache2 nous utilisons la commande **apt install apache2**.

Une fois installé nous n'avons pas besoin de le configurer, les paramètres par défaut nous suffisent pour cet exercice.

Pour ouvrir la page web sur notre navigateur nous tapons l'adresse ip de notre vm.





Nous remarquons que nous pouvons accéder à la page web depuis notre VM et depuis notre machine hôte.

Job 03:

Serveurs web:

Il existe de nombreux serveurs web parmi lesquels les 5 les plus utilisés sont:

- **Apache HTTP Server (Apache)** :

Apache est l'un des serveurs web les plus anciens et les plus populaires. Il est open source et extensible via des modules.

Les avantages à l'utiliser sont surtout parcequ'il est stable et bien documenté. Il est assez polyvalent et possède un module de sécurité pour la protection contre des attaques.

Les inconvénients sont qu'il peut être complexe à configurer pour les débutants (surtout à sécuriser) et que la configuration necessite de redémarrages fréquents en cas de modifications.

- **Nginx** :

Nginx est réputé pour sa haute performance et sa faible utilisation de ressources. Il est souvent utilisé comme serveur proxy ou équilibrage de charge.

Les avantages à l'utiliser sont qu'il est très performant étant capable de gérer de nombreuses connexions simultanées et il équilibre la charge et le proxy inversé de manière efficace. De plus il est peu gourmand en ressources.

Les inconvénients sont qu'il n'est pas si convivial pour les débutants, il ne supporte pas nativement les langages serveur, comme PHP, et nécessite l'utilisation de proxy.

- **Microsoft Internet Information Services (IIS)** :

IIS est le serveur web de Microsoft, principalement utilisé sur les serveurs Windows.

Ces avantages sont qu'il est intégré aux systèmes Windows Server, il supporte des technologies Microsoft telles que ASP.NET et il est facile à administrer via l'interface graphique.

Ces inconvénients sont qu'il est limité aux environnements Windows et donc moins populaire en dehors de cet environnement.

- **LiteSpeed** :

LiteSpeed est connu pour sa vitesse et sa sécurité. Il est compatible avec Apache et peut remplacer ce dernier sans modification majeure de la configuration.

Il est très performant, en particulier pour les sites à fort trafic. Il a la possibilité de basculer depuis Apache sans effort majeur et possède des fonctionnalités de sécurité.

Sont inconvénient majeur et qu'il n'est pas open source et la version gratuite à des fonctionnalités limitées.

- **Caddy** :

Caddy est un serveur web moderne qui se distingue par sa simplicité d'installation et de configuration.

Sont avantage principal est donc sont installation et configuration faciles. De plus, il prend en charge le chiffrement HTTPS par défaut et possède une interface de gestion web.

Ses inconvénients sont qu'il est moins flexible que certains autres serveurs et il possède des performances légèrement inférieures dans certaines situations par rapport à des serveurs web comme Nginx.

Job 04:

DNS:

I. Installation et configuration bind9

Afin de créer un **serveur DNS** nous pouvons utiliser **Bind9**. Pour l'installer nous utilisons **apt install bind9**.

Une fois installé nous devons configurer différents fichiers afin de créer notre domaine.

Tout d'abord nous allons configurer le **direct**.

Copions le fichier **db.local** avec **sudo cp /etc/bind/db.local /etc/bind/direct**.

Une zone directe (ou zone "forward") dans un serveur DNS gère la résolution des noms de domaine en adresses IP. En d'autres termes, elle associe des noms de domaine à des adresses IP correspondantes. Dans notre cas, il associera le nom **dnsproject.prepa.com** à l'adresse IP de notre machine. Les zones inverses font donc le contraire.

Éditons notre fichier direct avec sudo nano /etc/bind/direct.

```
  GNU nano 7.2                        /etc/bind/direct
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     prepa.com. dnsproject.prepa.com. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      dnsproject.prepa.com.
dnsproject      IN      A       192.168.0.15
www     IN      CNAME   dnsproject.prepa.com.
```

Une fois le direct configuré nous allons le copier dans **inverse** avec **sudo cp /etc/bind/direct /etc/bind/inverse**.

```
 GNU nano 7.2                        /etc/bind/inverse
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     prepa.com. dnsproject.prepa.com. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      dsnproject.prepa.com.
dnsproject      IN      A       192.168.0.15
10      IN      PTR     dnsproject.prepa.com.
```

Ensuite nous allons éditer le fichier **named.conf.local** situé dans **/etc/bind/**.

```
 GNU nano 7.2                 /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 "prepa.com" IN {
        type master;
        file"/etc/bind/direct";
};
zone "9.10.10.in-addr-arpa" IN {
        type master;
        file"/etc/bind/inverse";
};
```

(Veuillez noter que l'adresse à changé du fait que je bascule entre l'école et ma maison)

Une fois tout ceci réalisé nous allons configurer le fichier **resolv.conf** situé dans le dossier etc.

```
  GNU nano 7.2
# Generated by NetworkManager
search prepa.com
nameserver 192.168.0.15
```

Une fois réalisé ces étapes nous pouvons redémarrer le service **bind9**.

```
oroitz@mavm:~$ systemctl restart bind9
```

II. Test

Nous allons donc maintenant tester que notre configuration soit fonctionnelle sur notre VM et que nous puissions ping et afficher sur firefox notre site à partir de l'adresse **dnsproject.prepa.com**.

```
oroitz@mavm:~$ ping dnsproject.prepa.com
PING dnsproject.prepa.com (192.168.0.15) 56(84) bytes of data.
64 bytes from mavm (192.168.0.15): icmp_seq=1 ttl=64 time=0.015 ms
64 bytes from mavm (192.168.0.15): icmp_seq=2 ttl=64 time=0.037 ms
64 bytes from mavm (192.168.0.15): icmp_seq=3 ttl=64 time=0.046 ms
64 bytes from mavm (192.168.0.15): icmp_seq=4 ttl=64 time=0.045 ms
^C
--- dnsproject.prepa.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3054ms
rtt min/avg/max/mdev = 0.015/0.035/0.046/0.012 ms
```

Job 05:

Domaine public:

Généralement pour obtenir un nom de domaine nous devons sélectionner un **prestataire de services de nom de domaine** tels que GoDaddy, Namecheap, Google Domains, etc.

Ensuite nous devons vérifier la **disponibilité** du nom de domaine sur le site du prestataire. En effet, nous ne pouvons pas avoir deux sites différents avec le même nom de domaine.

Ensuite il faut choisir l'**extension de domaine** (Top-Level Domain), par exemple .com, .net, .org, .fr, .io, etc. Chaque extension de domaine a ses propres spécificités et restrictions (cf. fin du Job05.) et suivre les différentes étapes du site en question.

Une fois le nom de domaine enregistré, il faut configurer les **enregistrements DNS** pour pointer vers les serveurs de votre site web ou de votre service de messagerie.

Finalement il faut **payer des frais d'enregistrement** annuels pour conserver la propriété du nom de domaine.

Pour spécificités des extensions de domaine (TLD) il y en a :

gTLD (Generic Top-Level Domain) : Ce sont des extensions de domaine génériques, telles que .com, .org, .net. Elles sont généralement utilisées de manière générale et sont largement disponibles pour tout le monde.

ccTLD (Country Code Top-Level Domain) : Ce sont des extensions de domaine liées à des pays ou à des territoires, comme .fr (France), .uk (Royaume-Uni), .de (Allemagne), etc. Les ccTLD sont souvent soumis à des restrictions géographiques, ce qui signifie que vous devrez peut-être avoir une présence ou une adresse dans ce pays pour enregistrer un tel domaine.

TLD de Second Niveau Restreints : Certains TLD de second niveau (par exemple, .gov, .edu) sont réservés pour des entités spécifiques, comme les organismes gouvernementaux ou les établissements d'enseignement.

TLD de Second Niveau Génériques : Certains pays et registres offrent des TLD de second niveau génériques, qui peuvent être utilisés par n'importe qui sans restriction particulière.

Nouvelles Extensions (gTLD) : Ces extensions de domaine sont plus récentes et plus spécifiques, comme .app, .io, .blog, etc. Elles offrent parfois des opportunités intéressantes pour des noms de domaine pertinents.

Job 06:

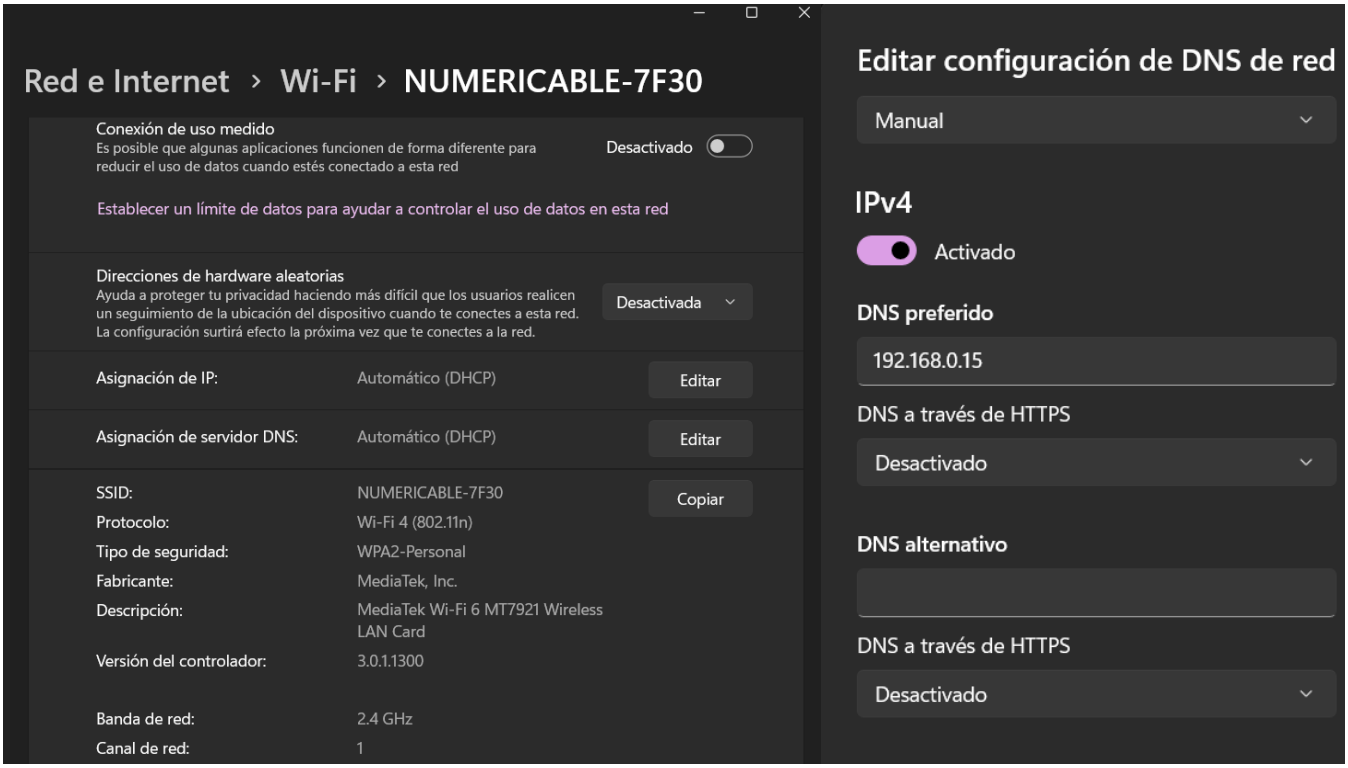
DNS machine hôte:

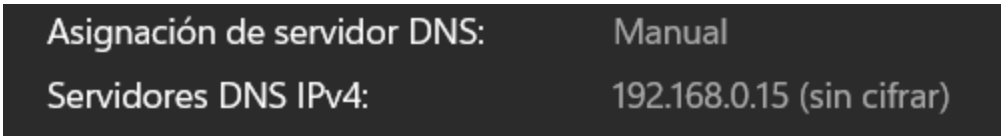
Afin de réaliser ce job, il est conseillé de configurer l'adaptateur réseau de notre VM en **bridge** ou en **host-only**. Sachant qu'avec host-only seuls notre hôte et notre VM communiqueront.

Dans mon cas, je vais relancer la VM en mode host-only.

I. Changer le dns de notre connexion wifi sur notre machine hôte

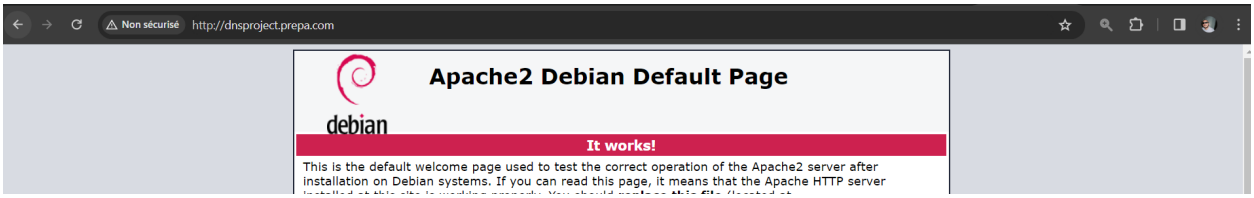
Afin de pouvoir utiliser le serveur dns de notre machine hôte (ici Windows 11) nous pouvons aller dans réseau, cliquer sur la configuration du wifi et dans la partie dns selectionner **manuel** et rentrer l'adresse ip de notre serveur dns (notre VM).





II. Test

Maintenant nous pouvons essayer d'accéder à notre serveur web à partir du navigateur web de notre machine hôte



Job 07:

Pare-feu ufw (firewall):

Bloquer les **pings (ICMP)** sur un serveur ou un pare-feu offre des avantages en matière de sécurité, notamment la **réduction de la visibilité** pour les attaquants et la **réduction de la surface d'attaque**. Il permet de se protéger des attaques **DDOS**. Cela peut également améliorer les **performances** dans certains cas. Cependant, cela peut compliquer le **dépannage**, avoir un impact sur certains services réseau et ne garantit pas une protection complète.

I. Installation et configuration de ufw (uncomplicated firewall)

Utilisons **sudo apt install ufw**, pour installer ufw. Une fois installé nous pouvons passer à la configuration de celui-ci.

Nous devons configurer le fichier nommé **before.rules** situé par défaut dans **/etc/ufw**.

Nous allons l'éditer avec nano: **sudo nano /etc/ufw/before.rules**.

```
 GNU nano 7.2                        /etc/ufw/before.rules
-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 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
-A ufw-before-forward -p icmp --icmp-type echo-request -j ACCEPT
```

Dans **# ok icmp codes for INPUT**, nous allons remplacer **ACCEPT** par **DROP**.

Ensuite nous allons permettre le **port 80** pour le protocole **HTTP** et le **port 443** pour le protocole **HTTPS**.

Nous allons utiliser : **sudo ufw allow 80/tcp** et **sudo ufw allow 443/tcp**.

Nous pouvons vérifier les règles appliquées avec **sudo ufw status**

Pour activer le firewall nous utiliserons **sudo ufw enable**.

II. Test

Nous pouvons remarquer que les **ping** ne fonctionnent plus. Tous les paquets envoyés sont **perdus**. Néanmoins la page web reste **accessible**.

```
C:\Users\Oroitz>ping 10.10.11.178

Haciendo ping a 10.10.11.178 con 32 bytes de datos:
Tiempo de espera agotado para esta solicitud.
Tiempo de espera agotado para esta solicitud.
Tiempo de espera agotado para esta solicitud.
Tiempo de espera agotado para esta solicitud.

Estadísticas de ping para 10.10.11.178:
    Paquetes: enviados = 4, recibidos = 0, perdidos = 4
    (100% perdidos),
```

Job 08:

SMB:

Pour mettre en place un dossier partagé sur notre serveur, nous pouvons utiliser le protocole **SMB (Server Message Block)** pour créer un partage réseau. Cela permettra aux autres membres de votre réseau d'accéder aux fichiers et de partager des fichiers dans ce dossier.

I. Installation et configuration de Samba

Nous allons utiliser **Samba**, nous utiliserons alors **sudo apt install samba**.

Nous allons le configurer un peu plus tard.

Maintenant nous allons créer un dossier qui va être celui du partage, dans mon cas, j'ai créé ce dossier dans le dossier home. Nous utiliserons **sudo mkdir /home/dossier-partage**.

Ensuite nous lui accordions tous les droits à tous les utilisateurs. Ceci **n'est pas une bonne pratique** car ça suppose un **problème de sécurité** très gros. Par simplicité j'ai choisi de donner tous les droits pour cet exercice. Nous utiliserons **sudo chmod -R 777 /home/dossier-partage**.

Éditons dès à présent la configuration de **samba**.

Utilisons **sudo nano /etc/samba/smb.conf**.

```
[Partage]
comment = Dossier de partage
path = /home/dossier-partage
valid users = @users
force group = users
create mask = 0660
directory mask = 0771
writable = yes
```

Nous allons ajouter à la fin du fichier ce bloc. Nous n'avons pas besoin de tous ces attributs, il s'agit de la configuration des règles spécifiques à ce partage. Le nom entre les crochets est le nom pour pouvoir accéder au dossier partagé.

Il ne nous reste plus qu'à configurer un mot de passe pour l'utilisateur avec **sudo smbpasswd -a oroitz**.

Nous pouvons alors restart le service, **systemctl restart samba**.

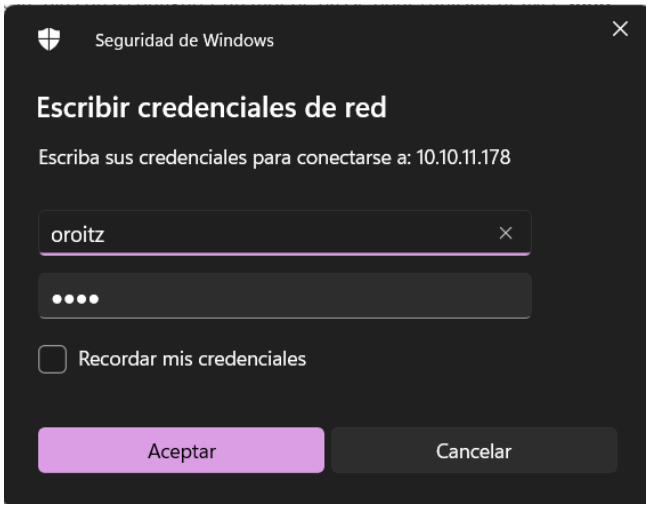
II. Configuration ufw

Notre nouveau **firewall** empêche la connexion par le protocole **smb**, pour permettre la connexion nous pouvons ouvrir le **port 139** et le **port 445**. Nous utiliserons les commandes **sudo ufw allow 139/tcp** et **sudo ufw allow 445/tcp**.

III. Test

Nous pouvons dès à présent tester depuis le navigateur de fichier de notre machine hôte.

Utilisons **\\10.10.11.178\Partage** dans l'explorateur de fichiers.



Il nous demandera alors de nous connecter. Nous utilisons l'utilisateur de ma vm, **oroitz**, et le mot de passe celui défini dans samba, dans notre cas, **root**.

Une fois connectés nous pouvons voir que les fichiers se partagent correctement.

Pour aller plus loin… :

Certification SSL:

II. A