Préparé par:
Douaa Hammouchi

Résumé

Ce rapport présente la mise en œuvre d'un environnement réseau virtuel sous Linux, réalisé dans le cadre d'un travail pratique en administration système. L'objectif principal est de concevoir, configurer et sécuriser un réseau informatique simulé à l'aide de machines virtuelles Debian, en utilisant VMware comme plateforme de virtualisation.

Le projet englobe plusieurs aspects essentiels de l'administration réseau, notamment la configuration des interfaces réseau, la mise en place du routage statique, la gestion des pare-feu avec iptables et nftables, la configuration d'un serveur DNS local avec BIND9 et le déploiement d'un serveur web Apache. De plus, une attention particulière a été portée à la connectivité entre les machines virtuelles et l'accès aux services depuis la machine hôte.

Ce document détaille les étapes de configuration, les commandes utilisées, les tests effectués, ainsi que les problèmes rencontrés et les solutions apportées. Il vise à fournir une compréhension approfondie des processus impliqués dans la création et la gestion d'un réseau informatique virtuel sécurisé et fonctionnel.

Introduction

1. Contexte

Dans le cadre de la formation en administration des systèmes et réseaux, il est essentiel de maîtriser la conception, la configuration et la gestion d'infrastructures réseau. Les environnements virtuels offrent une plateforme idéale pour simuler des réseaux complexes sans nécessiter de matériel physique supplémentaire.

2. Objectifs du projet

Ce projet a pour but de :

- ✓ Configurer un réseau virtuel composé de plusieurs machines Debian sous VMware.
- ✓ Mettre en place le routage entre différents sous-réseaux.
- ✓ Implémenter des règles de pare-feu pour sécuriser les communications.
- ✓ Déployer un serveur DNS local pour la résolution de noms.
- ✓ Installer et configurer un serveur web accessible.

3. Méthodologie

La réalisation du projet s'est déroulée en plusieurs étapes :

Préparation de l'environnement : Installation de VMware et création des machines virtuelles avec Debian.

Configuration réseau : Attribution des adresses IP, mise en place des interfaces réseau et du routage.

Sécurisation : Application de règles de pare-feu adaptées aux besoins du réseau.

Services réseau : Installation et configuration des services DNS et HTTP.

Tests et validation : Vérification de la connectivité, de la résolution de noms et de l'accès aux services.

4. Présentation des outils et technologies utilisés

VMware Workstation : Plateforme de virtualisation utilisée pour créer et gérer les machines virtuelles nécessaires au projet.

Debian 12.11.0 (mode terminal uniquement) : Système d'exploitation Linux choisi pour sa stabilité et sa compatibilité avec les outils réseau.

BIND9 : Serveur DNS utilisé pour la résolution des noms de domaine au sein du réseau virtuel.

Apache2 : Serveur web déployé pour héberger des pages web accessibles depuis les machines du réseau.

nftables / iptables : Outil de filtrage de paquets utilisé pour la configuration du pare-feu sur les machines virtuelles.

tcpdump : Analyse du trafic réseau.

netstat/ss : Liste des ports ouverts.

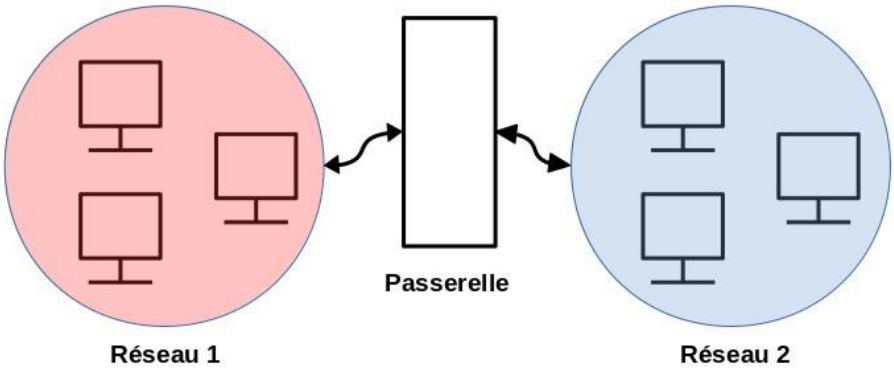
nmap : Scan des services réseau.

Partie 1 : Routage et Pare-feu

I. Routage

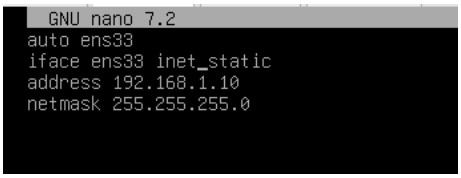
1) Création des machines virtuelles

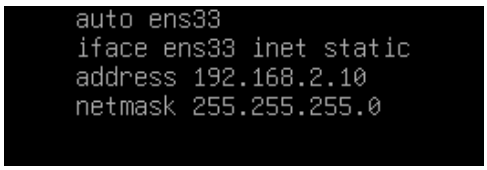
Trois machines virtuelles ont été créées sous VMware Workstation :
VM1 (client1) : Connectée au réseau VMnet2.
VM2 (client2) : Connectée au réseau VMnet3.
VM3 (router) : Connectée aux réseaux VMnet2, VMnet3, NAT et Host-Only.

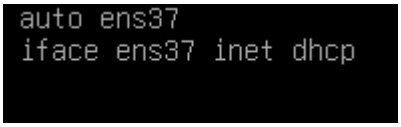


2) Configuration des interfaces réseau

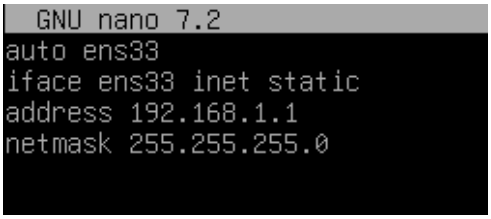
Sur chaque machine, les interfaces réseau ont été configurées avec des adresses IP statiques.
-Pour VM1 et VM2, on fait la configuration de l'interface ens33 et l'interface en37 comme suit:

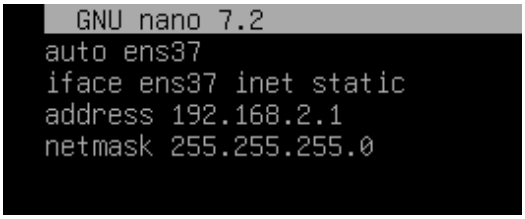






-Pour VM3, configuration des deux interfaces ens33 et ens37:





Après la configuration des interfaces réseau de chacune des machiones, on exécute la commande suivante: ***sudo service networking restart***

Pour vérifier les configurations faites sur les interfaces, on exécute la commande **ip a,** on obtient un résultat comme suit:

```
root@vm1:~# service networking restart
root@vm1:~# 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 noprefixroute
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:cd:47:7f brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.1.10/24 brd 192.168.1.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fecd:477f/64 scope link
       valid_lft forever preferred_lft forever
3: ens37: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:cd:47:89 brd ff:ff:ff:ff:ff:ff
    altname enp2s5
    inet 192.168.142.139/24 brd 192.168.142.255 scope global dynamic ens37
       valid_lft 1578sec preferred_lft 1578sec
    inet6 fe80::20c:29ff:fecd:4789/64 scope link
       valid_lft forever preferred_lft forever
```

-La VM3 agit comme un routeur entre les deux sous-réseaux 192.168.1.0/24 et192.168.2.0/24
Ses fonctions principales sont :

Relais de paquets : Transfère les données entre VM1 et VM2, qui appartiennent à des sous-réseaux différents.

Passerelle par défaut : Permet aux VMs d'accéder à des réseaux externes.

```
root@router:~# service networking restart
root@router:~# 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 noprefixroute
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:c5:ea:14 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.1.1/24 brd 192.168.1.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fec5:ea14/64 scope link
       valid_lft forever preferred_lft forever
3: ens37: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:c5:ea:1e brd ff:ff:ff:ff:ff:ff
    altname enp2s5
    inet 192.168.2.1/24 brd 192.168.2.255 scope global ens37
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fec5:ea1e/64 scope link
       valid_lft forever preferred_lft forever
4: ens38: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:c5:ea:28 brd ff:ff:ff:ff:ff:ff
    altname enp2s6
    inet 192.168.142.135/24 brd 192.168.142.255 scope global dynamic ens38
       valid_lft 1779sec preferred_lft 1779sec
    inet6 fe80::20c:29ff:fec5:ea28/64 scope link
       valid_lft forever preferred_lft forever
5: ens39: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:0c:29:c5:ea:32 brd ff:ff:ff:ff:ff:ff
    altname enp2s7
```

3) Ajouter des routes statiques :

- Ajout de la passerelle au niveau de Machine1 et Machine2:

```
root@vm1:~# ip r
default via 192.168.142.2 dev ens37
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.10
192.168.142.0/24 dev ens37 proto kernel scope link src 192.168.142.139
root@vm1:~# ip route add 192.168.2.0/24 via 192.168.1.1 dev ens33
root@vm1:~# ip r
default via 192.168.142.2 dev ens37
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.10
192.168.2.0/24 via 192.168.1.1 dev ens33
192.168.142.0/24 dev ens37 proto kernel scope link src 192.168.142.139
```

```
root@vm2:~# ip r
default via 192.168.142.2 dev ens37
192.168.2.0/24 dev ens33 proto kernel scope link src 192.168.2.10
192.168.142.0/24 dev ens37 proto kernel scope link src 192.168.142.136
root@vm2:~# ip route add 192.168.1.0/24 via 192.168.2.1 dev ens33
root@vm2:~# ip r
default via 192.168.142.2 dev ens37
192.168.1.0/24 via 192.168.2.1 dev ens33
192.168.2.0/24 dev ens33 proto kernel scope link src 192.168.2.10
192.168.142.0/24 dev ens37 proto kernel scope link src 192.168.142.136
```

4) Activation du Routage

On active l'option « ip_forwarding » sur la passerelle

```
root@router:~# sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
root@router:~# sysctl --system
* Applique /usr/lib/sysctl.d/50-pid-max.conf …
* Applique /usr/lib/sysctl.d/99-protect-links.conf …
* Applique /etc/sysctl.d/99-sysctl.conf …
sysctl: /etc/sysctl.d/99-sysctl.conf(69): erreur de syntaxe, je continue…
* Applique /etc/sysctl.conf …
sysctl: /etc/sysctl.conf(69): erreur de syntaxe, je continue…
kernel.pid_max = 4194304
fs.protected_fifos = 1
fs.protected_hardlinks = 1
fs.protected_regular = 2
fs.protected_symlinks = 1
root@router:~#
```

Puis on modifie le fichier de configuration d'interface ens33 pour vm1 et aussi vm2 pour activer la route statique: ***sudo nano /etc/network/interfaces.d/ens33***

up ip route add 192.168.2.0/24 via 192.168.1.1 dev ens33

Cette commande configure une route statique permettant à VM1 d'atteindre le réseau 192.168.2.0/24 en passant par la passerelle 192.168.1.1 via l'interface réseau ens33.

```
auto ens33
iface ens33 inet static
address 192.168.1.10
netmask 255.255.255.0
up ip route add 192.168.2.0/24 via 192.168.1.1 dev ens33
```

5) Test de connectivité:

Test de connectivité entre vm1 et vm2 à travers la commande *ping*.

```
root@vm1:~# ping 192.168.2.10
PING 192.168.2.10 (192.168.2.10) 56(84) bytes of data.
64 bytes from 192.168.2.10: icmp_seq=1 ttl=63 time=1.60 ms
64 bytes from 192.168.2.10: icmp_seq=2 ttl=63 time=0.680 ms
64 bytes from 192.168.2.10: icmp_seq=3 ttl=63 time=0.957 ms
64 bytes from 192.168.2.10: icmp_seq=4 ttl=63 time=0.735 ms
64 bytes from 192.168.2.10: icmp_seq=5 ttl=63 time=0.406 ms
^C
--- 192.168.2.10 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 0.406/0.875/1.598/0.401 ms
root@vm1:~#
```

II. Pare-feu:

1. iptables

Cette machine définit des règles pare-feu à l'aide de ***iptables***:

-On l'installe et on l'active:

```
root@vm1:~# apt install iptables-persistent -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
iptables-persistent est déjà la version la plus récente (1.0.20).
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
root@vm1:~# netfilter-persistent save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
root@vm1:~#
```

-Exécution de ces commandes:

```
root@vm1:~# iptables -F
root@vm1:~# iptables -X
root@vm1:~# iptables -P INPUT DROP
root@vm1:~# iptables -P FORWARD DROP
root@vm1:~# iptables -P OUTPUT ACCEPT
root@vm1:~# iptables -A INPUT -i  lo -j ACCEPT
root@vm1:~# iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
root@vm1:~# iptables -A INPUT -p tcp --dport 22 -j ACCEPT
root@vm1:~# iptables -A INPUT -p tcp --dport 80 -j ACCEPT
root@vm1:~# iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
root@vm1:~#
```

iptables -F: permet la réinitialisation des Règles

iptables -X pour nettoyer les anciennes règles personnalisées.

iptables -P INPUT DROP pour définir la politique par défaut (-P) de la chaîne INPUT sur DROP. Tous les paquets entrants sont bloqués sauf ceux explicitement autorisés ensuite.

iptables -P FORWARD DROP bloque le routage si la machine n'est pas configurée comme routeur.

iptables -A INPUT -i lo -j ACCEPT, autorise tout le trafic sur l'interface loopback (-i lo), ce qui permet aux applications locales de communiquer entre elles (ex: base de données, services internes).

iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

-m conntrack : Utilise le module de suivi de connexions.

--ctstate ESTABLISHED,RELATED : Autorise les paquets liés à des connexions existantes ou dérivées (ex: réponses à des requêtes sortantes).

Utilité : Évite de bloquer les réponses légitimes (ex: pages web chargées après une requête HTTP).

iptables -A INPUT -p tcp --dport 22 -j ACCEPT, autorise le trafic TCP sur le port 22 (SSH).

iptables -A INPUT -p tcp --dport 80 -j ACCEPT ce qui autorise le trafic TCP sur le port 80 (HTTP).

iptables -A INPUT -p icmp --icmp-type echo-request -j DROP

-p icmp : Cible le protocole ICMP.

--icmp-type echo-request : Bloque spécifiquement les requêtes de type "ping".

Remarque: Ces règles implémentent une politique de sécurité restrictive (tout est bloqué sauf l'essentiel).

Pour la vérification:

```
root@vm1:~# iptables -L -v -n
Chain INPUT (policy DROP 48 packets, 15744 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     0    --  lo     *       0.0.0.0/0            0.0.0.0/0
    5  1388 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0           ctstate RELATED,ESTABLISHED
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:22
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:80
    0     0 DROP       1    --  *      *       0.0.0.0/0            0.0.0.0/0           icmptype 8

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 5 packets, 1388 bytes)
 pkts bytes target     prot opt in     out     source               destination
root@vm1:~# cat /etc/iptables/rules.v4
# Generated by iptables-save v1.8.9 (nf_tables) on Fri May 23 19:40:14 2025
*filter
:INPUT DROP [17:5576]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [2:656]
-A INPUT -i lo -j ACCEPT
-A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -p icmp -m icmp --icmp-type 8 -j DROP
COMMIT
# Completed on Fri May 23 19:40:14 2025
root@vm1:~# _
```

2. nftables

En deuxième machine, on travaille avec ***nftables*** pour la tester aussi, passant de l'installation jusqu'à la définition des règles.

```
root@vm2:~# apt update
Atteint :1 http://deb.debian.org/debian bookworm InRelease
Réception de :2 http://security.debian.org/debian-security bookworm-security InRelease [48,0 kB]
Réception de :3 http://deb.debian.org/debian bookworm-updates InRelease [55,4 kB]
Réception de :4 http://security.debian.org/debian-security bookworm-security/non-free-firmware Sources [796 B]
Réception de :5 http://security.debian.org/debian-security bookworm-security/non-free-firmware amd64 Packages [688 B]
105 ko réceptionnés en 1s (150 ko/s)
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
1 paquet peut être mis à jour. Exécutez « apt list --upgradable » pour le voir.
root@vm2:~# apt install nftables -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
nftables est déjà la version la plus récente (1.0.6-2+deb12u2).
0 mis à jour, 0 nouvellement installés, 0 à enlever et 1 non mis à jour.
root@vm2:~# systemctl enable nftables
Created symlink /etc/systemd/system/sysinit.target.wants/nftables.service → /lib/systemd/system/nftables.service.
root@vm2:~# systemctl start nftables
root@vm2:~#
```

De même, nftables a été configuré pour autoriser les connexions SSH, HTTP et les pings, tout en bloquant les autres connexions entrantes. On accède au fichier de configuration **/etc/nftables.conf**

```
table inet filter {
        chain input {
                type filter hook input priority 0;
                policy drop;
      #loopback
      iif lo accept

     #coonexions établies
     ct state established,related accept

     #autoriser SSH
      tcp dport 22 accept

     #autoriser HTTP
     tcp dport 80 accept

     #bloquer ICMP echo-request
     ip protocol icmp icmp type echo-request drop
         }

         chain forward {
                type filter hook forward priority 0;
                policy drop;
         }

         chain output {
                type filter hook output priority 0;
         }
}
```

Puis on active le service nftables pour qu'il démarre à l'aide de ces commandes:

systemctl enable nftables
systemctl start nftables

Aussi pour vérfier les règles, on consulte *nft list ruleset.*

```
root@vm2:~# nft flush ruleset
root@vm2:~# nft -f /etc/nftables.conf
root@vm2:~# nft list ruleset
table inet filter {
        chain input {
                type filter hook input priority filter; policy drop;
                iif "lo" accept
                ct state established,related accept
                tcp dport 22 accept
                tcp dport 80 accept
                ip protocol icmp icmp type echo-request drop
        }

        chain forward {
                type filter hook forward priority filter; policy drop;
        }

        chain output {
                type filter hook output priority filter; policy accept;
        }
}
root@vm2:~#
```

3. Test de vérification

-On teste le ping (ICMP) de vm1 vers vm2 avec *ping 192.168.2.10.* Le ping échoue comme il est montré dans la figure, on remarque que 100% packet loss et 0 packet received.

```
--- 192.168.2.10 ping statistics ---
20 packets transmitted, 0 received, 100% packet loss, time 19433ms
```

-On effectue aussi le test de ssh depuis Vm1, on aura effectivement l'accès à vm2 après saisie de mot de passe:

```
root@vm1:~# ssh root@192.168.2.10
root@192.168.2.10's password:
Linux vm2 6.1.0-35-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.137-1 (2025-05-07) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@vm2:~#
```

On continue à travailler en vm2 et si on veut retourner vers notre machine, on se déconnecte:

```
root@vm2:~# exit
déconnexion
Connection to 192.168.2.10 closed.
root@vm1:~# _
```

Partie 2 : Services Réseaux

Objectifs de cette partie:

- ✓ Créer un serveur DNS local avec bind9 sur VM3.
- ✓ Héberger un site web local sur le même serveur avec apache2.
- ✓ Configurer les clients (VM1 et VM2) pour qu'ils résolvent un nom de domaine personnalisé.

Description :

La machine 3 agit comme un serveur centralisé hébergeant deux services critiques :
DNS (bind9) : Résolution de noms pour le domaine local tp-linux.local.
Web (Apache) : Hébergement d'une page HTML accessible aux clients VM1 et VM2.
Les VM1 et VM2 sont configurées comme **clients** pour interroger ces services.

1. Serveur DNS local avec bind9

-Sur le serveur installer bind9

apt update

apt install bind9 bind9utils dnsutils -y

```
Les paquets supplémentaires suivants seront installés :
  bind9-utils dns-root-data
Paquets suggérés :
  bind-doc resolvconf ufw
Les NOUVEAUX paquets suivants seront installés :
  bind9 bind9-utils dns-root-data
0 mis à jour, 3 nouvellement installés, 0 à enlever et 1 non mis à jour.
Il est nécessaire de prendre 411 ko dans les archives.
Après cette opération, 1 579 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 bind9-utils amd64 1:9.18.33-1~deb12u2 [159 kB]
Réception de :2 http://deb.debian.org/debian bookworm/main amd64 dns-root-data all 2024071801~deb12u1 [5 752 B]
Réception de :3 http://deb.debian.org/debian bookworm/main amd64 bind9 amd64 1:9.18.33-1~deb12u2 [246 kB]
411 ko réceptionnés en 2s (204 ko/s)
Sélection du paquet bind9-utils précédemment désélectionné.
(Lecture de la base de données... 34237 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../bind9-utils_1%3a9.18.33-1~deb12u2_amd64.deb ...
Dépaquetage de bind9-utils (1:9.18.33-1~deb12u2) ...
Sélection du paquet dns-root-data précédemment désélectionné.
Préparation du dépaquetage de .../dns-root-data_2024071801~deb12u1_all.deb ...
Dépaquetage de dns-root-data (2024071801~deb12u1) ...
Sélection du paquet bind9 précédemment désélectionné.
Préparation du dépaquetage de .../bind9_1%3a9.18.33-1~deb12u2_amd64.deb ...
Dépaquetage de bind9 (1:9.18.33-1~deb12u2) ...
Paramétrage de dns-root-data (2024071801~deb12u1) ...
Paramétrage de bind9-utils (1:9.18.33-1~deb12u2) ...
Paramétrage de bind9 (1:9.18.33-1~deb12u2) ...
Ajout du groupe « bind » (GID 109)...
Fait.
Ajout de l'utilisateur système « bind » (UID 101) ...
Ajout du nouvel utilisateur « bind » (UID 101) avec pour groupe d'appartenance « bind » ...
Pas de création du répertoire personnel « /var/cache/bind ».
wrote key file "/etc/bind/rndc.key"
named-resolvconf.service is a disabled or a static unit, not starting it.
Created symlink /etc/systemd/system/bind9.service → /lib/systemd/system/named.service.
Created symlink /etc/systemd/system/multi-user.target.wants/named.service → /lib/systemd/system/named.service.
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...
root@router:~#
```

- Déclaration de la zone: /etc/bind/named.conf.mes-zones

Enregistrements DNS: /etc/bind/db.tp-linux.local

-Créer une zone DNS personnalisée en définissant:

·Une zone directe pour **tp-linux.local**

·Deux zones inverses pour les réseaux **192.168.1.0/24** et **192.168.2.0/24**

-On exécute la commande:

nano /etc/bind/named.conf.mes-zones

```
//zone directe (Nom-IP)
 zone "tp-linux.local"{
  type master;
  file "/etc/bind/db.tp-linux.local";
};

//zone inverse pour 192.168.1.0/24
 zone "1.168.192.in-addr.arpa"{
  type master;
  file "/etc/bind/db.1.168.192";
};

//zone inverse pour 192.168.2.0/24
 zone "2.168.192.in-addr.arpa"{
  type master;
  file "/etc/bind/db.2.168.192";
};
```

Zone directe; On crée le fichier **db.tp-linux.local**

```
root@router:~# nano /etc/bind/db.tp-linux.local
```

cp /etc/bind/db.local /etc/bind/db.tp-linux.local
nano /etc/bind/db.tp-linux.local

```
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     tp-linux.local. root.tp-linux..local. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      localhost.
@       IN      A       127.0.0.1
@       IN      AAAA    ::1

;DNS Server
@       IN       NS      dns.tp-linux.local.

;Enregistrements A
dns       IN        A       192.168.1.1
vm1       IN        A       192.168.1.10
vm2       IN        A       192.168.2.10
```

Creation de zone inverse **/etc/bind/db.1.168.192** (pour VM1 + DNS):

```
root@router:~# cp /etc/bind/db.127 /etc/bind/db.1.168.192
root@router:~# nano /etc/bind/db.1.168.192
```

```
;
; BIND reverse data file for local loopback interface
;
$TTL    604800
@       IN      SOA     tp-linux.local. root.tp-linux.local. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      dns.tp-linux.local.
1.0.0   IN      PTR     dns.tp-linux.local.
10.0.0  IN      PTR     vm1.tp-linux.local.
```

Création de zone inverse **/etc/bind/db.2.168.192** pour Vm2 :

cp /etc/bind/db.127 /etc/bind/db.2.168.192
nano /etc/bind/db.2.168.192

```
; BIND reverse data file for local loopback interface
;
$TTL    604800
@       IN      SOA     tp-linux.local. roo.tp-linux.local. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      dns.tp-linux.local.
10      IN      PTR     vm2.tp-linux.local.
```

Puis on vérifie la configuration :

named-checkconf
named-checkzone tp-linux.local /etc/bind/db.tp-linux.local
named-checkzone 1.168.192.in-addr.arpa /etc/bind/db.1.168.192
named-checkzone 2.168.192.in-addr.arpa /etc/bind/db.2.168.192

Si tous les fichiers sont corrects, les commandes doivent afficher OK. Et c'est effectivement le cas:

```
root@router:~# named-checkconf
root@router:~# named-checkzone tp-linux.local /etc/bind/db.tp-linux.local
zone tp-linux.local/IN: loaded serial 2
OK
root@router:~# named-checkzone 1.168.192.in-addr.arpa /etc/bind/db.1.168.192
zone 1.168.192.in-addr.arpa/IN: loaded serial 1
OK
root@router:~# named-checkzone 2.168.192.in-addr.arpa /etc/bind/db.2.168.192
zone 2.168.192.in-addr.arpa/IN: loaded serial 1
OK
root@router:~#
```

Puis redémarrer bind9 **systemctl restart bind9**
systemctl enable bind9

Puis on fait la configuration des clients (VM1 et VM2) pour utiliser VM3 comme DNS. Sur VM1 et VM2, on édite **/etc/resolv.conf** : nameserver 192.168.1.1

-Faisant maintenant les tests, depuis machine client vm2:

Résolution directe

```
root@vm2:~# dig vm1.tp-linux.local

; <<>> DiG 9.18.33-1~deb12u2-Debian <<>> vm1.tp-linux.local
;; global options: +cmd
;; Got answer:
;; WARNING: .local is reserved for Multicast DNS
;; You are currently testing what happens when an mDNS query is leaked to DNS
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39103
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 88d7b6e31010f2ea010000006831c3a4f396bd4d14530bfe (good)
;; QUESTION SECTION:
;vm1.tp-linux.local.            IN      A

;; ANSWER SECTION:
vm1.tp-linux.local.     604800  IN      A       192.168.1.10

;; Query time: 0 msec
;; SERVER: 192.168.2.1#53(192.168.2.1) (UDP)
;; WHEN: Sat May 24 14:03:32 +01 2025
;; MSG SIZE  rcvd: 91
```

Résolution inverse:

```
root@vm2:~# dig -x 192.168.1.10

; <<>> DiG 9.18.33-1~deb12u2-Debian <<>> -x 192.168.1.10
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 61407
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 5932011e8a8decae010000006831c4877d151515af8b1844 (good)
;; QUESTION SECTION:
;10.1.168.192.in-addr.arpa.     IN      PTR

;; AUTHORITY SECTION:
1.168.192.in-addr.arpa. 604800  IN      SOA     tp-linux.local. root.tp-linux.local. 1 604800 86400 2419200 604800

;; Query time: 0 msec
;; SERVER: 192.168.2.1#53(192.168.2.1) (UDP)
;; WHEN: Sat May 24 14:07:19 +01 2025
;; MSG SIZE  rcvd: 137
```

Aussi on fait le test à l'aide de: nslookup www.tp-linux.local

2. Serveur web Apache local :

Accéder au site depuis VM1 et VM2 en utilisant un nom DNS local comme site.tp-linux.local

Vérifier que le pare-feu permet HTTP (port 80)

- Installer Apache2

```
apt update
apt install apache2 -y
```

-Vérifie que le service est actif : systemctl status apache2, on trouve **active (running)**

```
root@router:~# systemctl status apache2
• apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Sat 2025-05-24 18:24:31 +01; 1min 58s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 2058 (apache2)
      Tasks: 55 (limit: 2262)
     Memory: 9.1M
        CPU: 51ms
     CGroup: /system.slice/apache2.service
             ├─2058 /usr/sbin/apache2 -k start
             ├─2060 /usr/sbin/apache2 -k start
             └─2061 /usr/sbin/apache2 -k start

mai 24 18:24:31 router systemd[1]: Starting apache2.service - The Apache HTTP Server...
mai 24 18:24:31 router systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Ajouter un enregistrement DNS pour site.tp-linux.local, on modifie le fichier de configuration puis on recharge BIND.
Cela fait pointer **site.tp-linux.local** vers l'interface de VM3.

```
;DNS Server
@       IN       NS      dns.tp-linux.local.


dns        IN         A       192.168.1.1
vm1        IN         A       192.168.1.10
vm2        IN         A       192.168.2.10
site       IN         A       192.168.1.1
```

```
root@router:~# rndc reload
server reload successful
```

-Maintenant, il est temps pour créer une page web (HTML) personnalisée, on va juste éditer la page d'accueil par défaut.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html lang="fr" >
  <head>
     <title>Bienvenue chez Douaa Hammouchi</title>
 </head>
    <body style= background: blue; color: white; text-align: center; padding 50px >
 <h1>Bienvenue en page test</h1>
 <p>Page d'acceuil du TP Administration reseau</p>
</body>
</html>
```

Test de connectivité:

-On effectue donc sur VM1 et VM2 un test d'accès pour tester la résolution DNS:

```
root@vm2:~# dig site.tp-linux.local

; <<>> DiG 9.18.33-1~deb12u2-Debian <<>> site.tp-linux.local
;; global options: +cmd
;; Got answer:
;; WARNING: .local is reserved for Multicast DNS
;; You are currently testing what happens when an mDNS query is leaked to DNS
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 7858
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 97a259d79b5822ac0100000068320ebc3f476454db012214 (good)
;; QUESTION SECTION:
;site.tp-linux.local.           IN      A

;; ANSWER SECTION:
site.tp-linux.local.    604800  IN      A       192.168.1.1

;; Query time: 4 msec
;; SERVER: 192.168.2.1#53(192.168.2.1) (UDP)
;; WHEN: Sat May 24 19:23:56 +01 2025
;; MSG SIZE  rcvd: 92
```

Tester l'accès au site web : **curl http://site.tp-linux.local**

```
root@vm2:~# curl http://site.tp-linux.local
-bash: curl : commande introuvable
root@vm2:~# apt install curl -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  libcurl4
Les NOUVEAUX paquets suivants seront installés :
  curl libcurl4
0 mis à jour, 2 nouvellement installés, 0 à enlever et 1 non mis à jour.
Il est nécessaire de prendre 707 ko dans les archives.
Après cette opération, 1 361 ko d'espace disque supplémentaires seront utilisé
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 libcurl4 amd6
Réception de :2 http://deb.debian.org/debian bookworm/main amd64 curl amd64 7.
707 ko réceptionnés en 5s (131 ko/s)
Sélection du paquet libcurl4:amd64 précédemment désélectionné.
(Lecture de la base de données... 34300 fichiers et répertoires déjà installés
Préparation du dépaquetage de .../libcurl4_7.88.1-10+deb12u12_amd64.deb ...
Dépaquetage de libcurl4:amd64 (7.88.1-10+deb12u12) ...
Sélection du paquet curl précédemment désélectionné.
Préparation du dépaquetage de .../curl_7.88.1-10+deb12u12_amd64.deb ...
Dépaquetage de curl (7.88.1-10+deb12u12) ...
Paramétrage de libcurl4:amd64 (7.88.1-10+deb12u12) ...
Paramétrage de curl (7.88.1-10+deb12u12) ...
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.36-9+deb12u10
Progression : [ 89%] [#################################################################
```

-Configurer le pare-feu sur VM3 pour permettre les connexions entrantes sur le port 80 (HTTP), assurant ainsi l'accessibilité du serveur web Apache depuis les clients du réseau.

 # Créer une table et une chaîne (si elles n'existent pas)
sudo nft add table ip filter
sudo nft add chain ip filter input { type filter hook input priority 0 \; policy drop \; }

 # Autoriser HTTP
sudo nft add rule ip filter input tcp dport 80 accept

 # Sauvegarder
sudo nft list ruleset > /etc/nftables.conf

Ou en accède au fichier de configuration **/etc/nftables.conf** et entrer les commandes directement comme suit:

```
flush ruleset

table inet filter {
        chain input {
                type filter hook input priority 0;
         iif "lo" accept
         ct state established,related accept
         ip protocol icmp accept
         tcp dport 80 accept
         tcp dpoet 22 accept

#refuser autre
reject with icmp type port-unreachable
        }
        chain forward {
                type filter hook forward priority 0;
        }
        chain output {
                type filter hook output priority 0;
        }
}
```

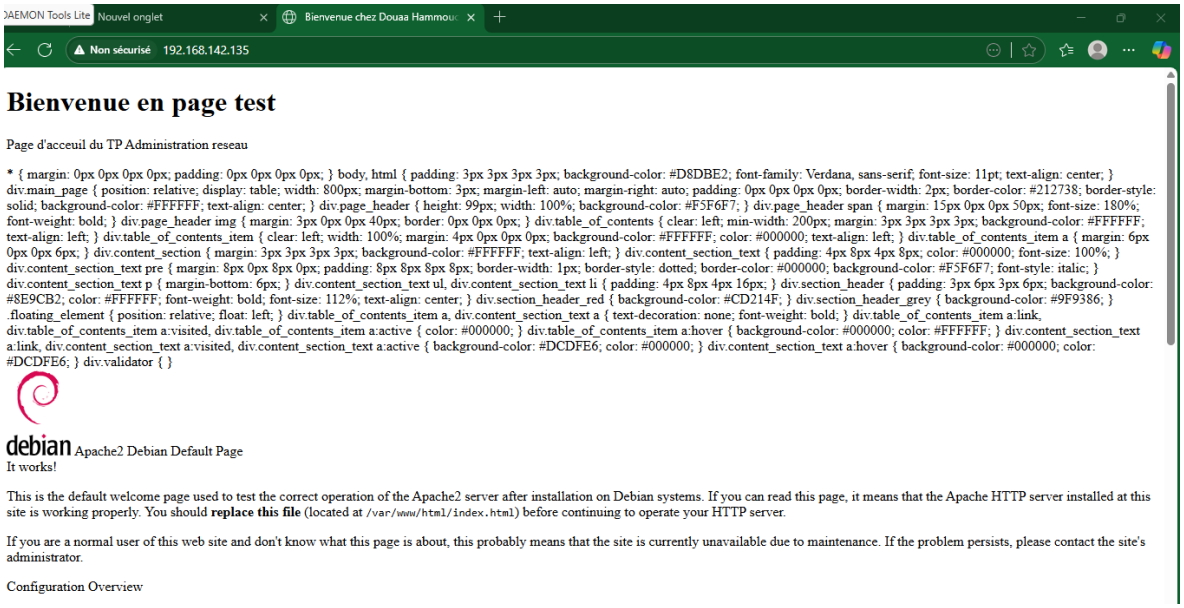
Pour vérifier la liste des règles: **sudo nft list ruleset**

```
root@router:~# nft -c -f /etc/nftables.conf
root@router:~# systemctl restart nftables
root@router:~# nft list ruleset
table inet filter {
        chain input {
                type filter hook input priority filter; policy accept;
                iif "lo" accept
                ct state established,related accept
                ip protocol icmp accept
                tcp dport 80 accept
                tcp dport 22 accept
                reject with icmp port-unreachable
        }

        chain forward {
                type filter hook forward priority filter; policy accept;
        }

        chain output {
                type filter hook output priority filter; policy accept;
        }
}
```

Effectivement le test montre bien que le processus de configuration du serveur web est bien fait. On peut faire un test sur la machine virtuelle ou aussi sur localhost puisqu'on a ajouté une interface host-only et NAT pour avoir accès à Internet.



Partie 3 : Dépannage

L'objectif de cette section est de :

- Simuler des pannes réseau courantes (interface down, DNS non répondu, service web inaccessible).
- Utiliser des outils d'analyse (tcpdump, netstat, ss, nmap) pour diagnostiquer les problèmes.
- Résoudre les pannes en appliquant des correctifs appropriés.

1) Préparation de l'environnement de test:

Avant de commencer le dépannage on prépare les éléments suivants :

sudo apt install tcpdump nmap net-tools

Scénarios de panne à simuler :

Panne 1 : Désactivation d'une interface réseau c'est à dire perte de connectivité entre sous-réseaux

sudo ip link set ens37 down

Panne 2 : Arrêt du service DNS et le rendre inaccessible

systemctl stop bind9

Panne 3 : Blocage du port HTTP par le pare-feu pour rendre Serveur web injoignable

nft add rule filter input tcp dport 80 drop

```
root@router:~# ip link set ens37 down
root@router:~# systemctl stop bind9
root@router:~# nft add rule inet filter input tcp dport 80 drop
root@router:~#
```

Pour vérifier si les problèmes sont vraiement créés:

```
root@router:~# ip a show ens37
3: ens37: <BROADCAST,MULTICAST> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 00:0c:29:c5:ea:1e brd ff:ff:ff:ff:ff:ff
    altname enp2s5
    inet 192.168.2.1/24 brd 192.168.2.255 scope global ens37
       valid_lft forever preferred_lft forever
root@router:~# nslookup www.tp-linux.local 127.0.0.1
;; communications error to 127.0.0.1#53: connection refused
;; communications error to 127.0.0.1#53: connection refused
;; communications error to 127.0.0.1#53: connection refused
;; no servers could be reached
```

2) Étapes de dépannage et Outils de diagnostic :

Pour l'interface réseau, on la réactive :

```
root@router:~# ip link set ens37 up
root@router:~# ip route show
default via 192.168.142.2 dev ens38
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168.1.1
192.168.2.0/24 dev ens37 proto kernel scope link src 192.168.2.1
192.168.56.0/24 dev ens39 proto kernel scope link src 192.168.56.10
192.168.142.0/24 dev ens38 proto kernel scope link src 192.168.142.135
```

-Pour le service DNS en panne, on le redémarre via ces commandes:

systemctl start bind9
named-checkconf

-Rétablir le HTTP

```
root@router:~# nft --handle --numeric list chain inet filter input
table inet filter {
        chain input { # handle 1
                type filter hook input priority 0; policy accept;
                iif "lo" accept # handle 4
                ct state 0x2,0x4 accept # handle 5
                ip protocol 1 accept # handle 6
                tcp dport 22 accept # handle 8
                tcp dport 80 accept # handle 9
                tcp dport 80 drop # handle 10
        }
}
root@router:~# nft delete rule inet filter input handle 10
root@router:~# nft list ruleset | grep 80
                tcp dport 80 accept
root@router:~# _
```

Utilisation des outils de diagnostic:

-On peut **tcpdump** analyser le trafic réseau: **sudo tcpdump -i ens37 -n**

-On peut utiliser **netstat** pour lister tous les ports en écoute **netstat -tuln** et aussi lister tous les ports ouverts: **sudo netstat -tulnp.**

```
Connexions Internet actives (seulement serveurs)
Proto Recv-Q Send-Q Adresse locale          Adresse distante        Etat        PID/Program name
tcp        0      0 192.168.1.1:53          0.0.0.0:*               LISTEN      3626/named
tcp        0      0 192.168.1.1:53          0.0.0.0:*               LISTEN      3626/named
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      3626/named
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN      3626/named
tcp        0      0 192.168.2.1:53          0.0.0.0:*               LISTEN      3626/named
tcp        0      0 192.168.2.1:53          0.0.0.0:*               LISTEN      3626/named
tcp        0      0 192.168.56.10:53        0.0.0.0:*               LISTEN      3626/named
tcp        0      0 192.168.56.10:53        0.0.0.0:*               LISTEN      3626/named
tcp        0      0 192.168.142.135:53      0.0.0.0:*               LISTEN      3626/named
tcp        0      0 192.168.142.135:53      0.0.0.0:*               LISTEN      3626/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      3626/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      3626/named
tcp6       0      0 ::1:953                 :::*                    LISTEN      3626/named
tcp6       0      0 ::1:953                 :::*                    LISTEN      3626/named
tcp6       0      0 :::80                   :::*                    LISTEN      3354/apache2
tcp6       0      0 fe80::20c:29ff:fec5::53 :::*                    LISTEN      3626/named
tcp6       0      0 fe80::20c:29ff:fec5::53 :::*                    LISTEN      3626/named
tcp6       0      0 fe80::20c:29ff:fec5::53 :::*                    LISTEN      3626/named
tcp6       0      0 fe80::20c:29ff:fec5::53 :::*                    LISTEN      3626/named
tcp6       0      0 fe80::20c:29ff:fec5::53 :::*                    LISTEN      3626/named
tcp6       0      0 fe80::20c:29ff:fec5::53 :::*                    LISTEN      3626/named
tcp6       0      0 fe80::20c:29ff:fec5::53 :::*                    LISTEN      3626/named
tcp6       0      0 fe80::20c:29ff:fec5::53 :::*                    LISTEN      3626/named
tcp6       0      0 ::1:53                  :::*                    LISTEN      3626/named
tcp6       0      0 ::1:53                  :::*                    LISTEN      3626/named
udp        0      0 192.168.56.10:53        0.0.0.0:*                           3626/named
udp        0      0 192.168.56.10:53        0.0.0.0:*                           3626/named
udp        0      0 192.168.142.135:53      0.0.0.0:*                           3626/named
udp        0      0 192.168.142.135:53      0.0.0.0:*                           3626/named
udp        0      0 192.168.2.1:53          0.0.0.0:*                           3626/named
udp        0      0 192.168.2.1:53          0.0.0.0:*                           3626/named
udp        0      0 192.168.1.1:53          0.0.0.0:*                           3626/named
udp        0      0 192.168.1.1:53          0.0.0.0:*                           3626/named
udp        0      0 127.0.0.1:53            0.0.0.0:*                           3626/named
```

Filtrer apache:

```
root@router:~# netstat -tulnp | grep :80
tcp6       0      0 :::80                   :::*                    LISTEN      3354/apache2
```

-Ou utilise aussi **ss** qui est un alternative moderne à netstat. On l'utlise pour vérifier les connexions actives à Apache.

```
root@router:~# ss -tnp state established '( dport = :80 or sport = :80 )'
Recv-Q          Send-Q          Local Address:Port          Peer Address:Port          Process
root@router:~#
```

On utilise souvent **nmap** qui est un outil pour détecter les hôtes actifs sur un réseau, identifier les ports ouverts et les services associés et déterminer le système d'exploitation des machines distantes.

```
root@router:~# nmap -sV -p 22,80 192.168.1.1
Starting Nmap 7.93 ( https://nmap.org ) at 2025-05-24 23:31 +01
Nmap scan report for 192.168.1.1
Host is up (0.00010s latency).

PORT   STATE  SERVICE VERSION
22/tcp closed ssh
80/tcp open   http    Apache httpd 2.4.62 ((Debian))
```

-sV : détection des versions
-p : ports spécifiques

Problèmes et difficultés rencontrés:

Au cours du projet, plusieurs défis ont été identifiés et résolus :
Conflits d'adresses IP : Assurer que chaque machine possède une adresse IP unique pour éviter les conflits.

Erreurs de configuration DNS : S'assurer que les fichiers de zone DNS sont correctement configurés et que le service BIND9 fonctionne sans erreur.

```
domain localdomain
search localdomain
nameserver 192.168.142.2
```

on a changé le nameserver.

Problèmes de Résolution DNS
Impossible de résoudre **tp-linux.local** depuis les machines clientes et des erreurs de type "Name or service not known" lors des tests avec **dig** ou **nslookup**
Causes identifiées :
Configuration incorrecte du fichier /etc/resolv.conf :
Le fichier pointait vers le mauvais serveur DNS (127.0.0.53 par défaut au lieu de 192.168.1.10)
-Les fichiers de configurations ne font pas partie donc voici les commandes que j'ai saisi:

```
//
// Please read /usr/share/doc/bind9/README.Debian for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";
```

Problèmes de connectivité HTTP, malgré la configuration du serveur Apache, le service n'était pas accessible depuis le navigateur.

Solution : Vérification des règles de pare-feu avec **nftables** et ajout d'une règle pour autoriser le trafic entrant sur le port 80.

Connexion SSH échouée: faire un diagnostic puis résoudre le problème avec:
sudo systemctl start ssh
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
sudo sed -i 's/PermitRootLogin no/PermitRootLogin yes/' /etc/ssh/sshd_config

Il y avait problème de permissions (fichier en lecture seule)
PermitRootLogin yes
PasswordAuthentication yes
#Port 22
#PermitEmptyPasswords no
#PubkeyAuthentication yes
#ChallengeResponseAuthentication no

Puis on le redémarre: systemctl restart ssh

Leçons Apprises
Validation systématique :
Toujours vérifier les configurations avec **dig**, **ss**, et **ping** après modification
Approche méthodique :
Isoler chaque problème (DNS, puis SSH, puis routage)
Utiliser les outils de diagnostic réseau tel que **tcpdump** pour analyser le trafic problématique.

Problème	Cause Probable	Solution	Commande de Vérification
DNS inaccessible	Mauvais résolveur	Correction de /etc/resolv.conf	dig +short tp-linux.local
SSH bloqué	Pare-feu/Service arrêté	Ouvrir port 22 + démarrer sshd	systemctl status ssh
Routage instable	Conflit DHCP/Static	Configuration manuelle IP	ip route show
Règles pare-feu perdues	Outil non persistant	Installation iptables-persistent	iptables-save

Conclusion

Ce projet de mise en place d'une infrastructure réseau virtuelle a permis de concevoir et de configurer un environnement complet, intégrant des services essentiels tels que le DNS, le serveur web Apache2 et un pare-feu sécurisé. Grâce à l'utilisation de VMware, nous avons simulé un réseau d'entreprise réaliste, facilitant ainsi l'apprentissage pratique des concepts d'administration système et réseau.

Au cours de ce projet, plusieurs défis ont été rencontrés, notamment des problèmes de connectivité et de configuration des services. Ces obstacles ont été surmontés grâce à une analyse approfondie, à l'application de solutions adaptées et à une documentation rigoureuse, aussi le découvert des outils de diagnostic. Cette expérience a renforcé notre capacité à diagnostiquer et à résoudre des problèmes techniques complexes.

En conclusion, ce projet a non seulement consolidé nos compétences techniques, mais a également mis en évidence l'importance d'une planification minutieuse, d'une documentation détaillée et d'une approche méthodique dans la gestion de projets informatiques. Les connaissances acquises serviront de base solide pour des projets futurs et pour une carrière professionnelle dans le domaine des technologies de l'information.