

SOMMAIRE

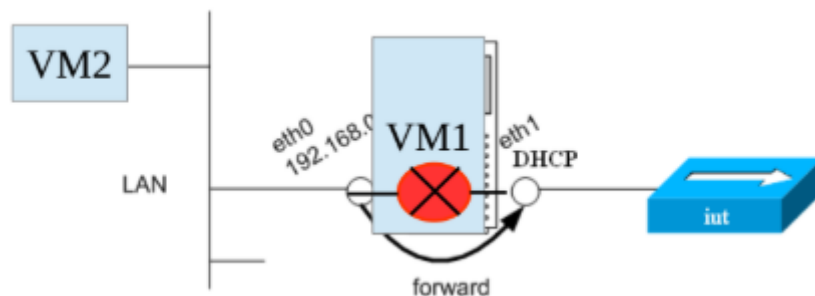
Introduction.....	1
A - Réalisation d'une infrastructure réseau privé avec DMZ et liaison Internet.....	1
a.1 VM1 : ROUTEUR / PAT.....	2
a.2 DMZ.....	4
a.3 NAT.....	5
B - VLAN.....	6
Conclusion.....	8

Introduction

L'objectif de ce TP était de mettre en place une infrastructure réseau virtualisée sous Linux (Debian) comprenant un routeur, un réseau LAN sécurisé, une DMZ pour un serveur Web et l'utilisation de VLANs (802.1Q). Nous avons également configuré la persistance des règles de routage et de pare-feu (iptables).

A - Réalisation d'une infrastructure réseau privé avec DMZ et liaison Internet

Architecture mise en place :



Ce schéma représente l'architecture logique réalisée : un routeur central Debian interconnectant le réseau IUT, le LAN privé, la DMZ Web et les VLANs.

Nous avons déployé deux machines virtuelles sous Debian :

VM1 (Passerelle) :

- Interface enp0s3 en Pont (Bridge) sur le réseau IUT/Internet (IP DHCP).
- Interface enp0s8 en LAN Interne (IP : 192.168.200.254/24).

```

VM1
root@debianVM:~#
root@debianVM:~# ip -c 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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:25:b0:29 brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.135/23 brd 192.168.51.255 scope global dynamic enp0s3
        valid_lft 6435sec preferred_lft 6435sec
    inet6 fe80::a00:27ff:fe25:b029/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:cf:3d:9f brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.254/24 brd 192.168.200.255 scope global enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fecf:3d9f/64 scope link
        valid_lft forever preferred_lft forever
    
```

Cette capture confirme la configuration réseau : l'interface WAN (enp0s3) est en DHCP, tandis que l'interface LAN (enp0s8) possède l'IP statique de passerelle 192.168.200.254.

VM2 (Client) :

→ Interface enp0s3 en LAN Interne (IP : 192.168.200.1/24).

```

VM2
root@debianVM:~# ip -c 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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:bb:d4:11 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.1/24 brd 192.168.200.255 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:febb:d411/64 scope link
        valid_lft forever preferred_lft forever
    
```

La VM2 est configurée avec l'adresse IP 192.168.200.1, ce qui la place correctement dans le même sous-réseau que l'interface LAN de la passerelle VM1.

a.1 VM1 : ROUTEUR / PAT

Activation du routage Pour permettre la communication entre les cartes, nous avons activé le forwarding IPv4 de manière persistante dans `/etc/sysctl.conf` :
`net.ipv4.ip_forward=1`

```

VM1
GNU nano 7.2 /etc/sysctl.conf
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
    
```

L'activation du paramètre `ip_forward` transforme VM1 en routeur. Elle peut désormais transmettre les paquets entre son interface WAN externe et le réseau local interne.

Activation du PAT (Masquerade).

```

iptables -t nat -A POSTROUTING -o enp0s3 -s 192.168.200.0/24 -j MASQUERADE
    
```

Nous avons configuré le Source NAT pour que les machines du LAN puissent accéder à Internet. Comme demandé, la règle ne s'applique que si le paquet provient du réseau privé.

Depuis notre VM2 nous effectuons un ping vers 8.8.8.8

```

VM2
root@debianVM:~# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=117 time=51.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=117 time=81.8 ms
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 51.498/66.660/81.822/15.162 ms
    
```

Le succès du ping vers 8.8.8.8 depuis le client LAN confirme que le routage est actif et que la règle de NAT fonctionne correctement.

Question : *Un Hacker présent sur le réseau IUT peut définir votre passerelle (VM1) comme routeur par défaut pour rebondir via votre PAT. Quelle règle permet de bloquer cette situation ?*

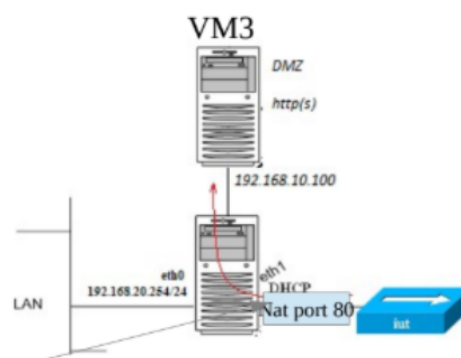
La première protection consiste à restreindre l'activation du Masquerade aux seules adresses IP sources de notre LAN (comme configuré précédemment avec l'option -s 192.168.200.0/24). Pour renforcer la sécurité et empêcher tout transit non sollicité, nous ajoutons une règle interdisant le routage (FORWARD) pour tout paquet entrant par l'interface WAN :

```
iptables -A FORWARD -i enp0s3 -j DROP
```

Cette règle bloque tout nouveau trafic initié depuis l'extérieur vers l'intérieur, tout en laissant passer les réponses grâce au suivi de connexion

a.2 DMZ

Nous avons étendu l'architecture en ajoutant une troisième interface réseau sur VM1 et une nouvelle machine VM3.



- Réseau DMZ : 192.168.10.0/24
- VM1 (Interface DMZ) : 192.168.10.254
- VM3 (Serveur Web) : 192.168.10.100

Un serveur Apache2 a été installé sur VM3 avec les commandes :

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

Pour vérifier que notre serveur apache est bien actif on exécute la commande

```
service apache2 status
```

```
root@debianVM:~# service apache2 status
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-12-04 08:30:28 +04; 57min ago
     Docs: https://httpd.apache.org/docs/2.4/
    Main PID: 1422 (apache2)
      Tasks: 55 (limit: 4635)
     Memory: 9.8M
        CPU: 309ms
    CGroup: /system.slice/apache2.service
            └─1422 /usr/sbin/apache2 -k start
              └─1424 /usr/sbin/apache2 -k start
                └─1425 /usr/sbin/apache2 -k start

déc. 04 08:30:28 debianVM systemd[1]: Starting apache2.service - The Apache HTTP Server...
déc. 04 08:30:28 debianVM apachectl[1421]: AH00558: apache2: Could not reliably determine
déc. 04 08:30:28 debianVM systemd[1]: Started apache2.service - The Apache HTTP Server.
Lines 1-16/16 (END)
```

Le service Apache2 est démarré et fonctionnel. Ce serveur Web, situé dans la DMZ, sera la cible des requêtes externes redirigées par la passerelle.

En utilisant la commande curl pour récupérer le code HTML de la page :

```
curl 192.168.10.100
```

L'affichage du code HTML via la commande curl prouve que le client du LAN accède correctement au serveur web situé dans la zone DMZ

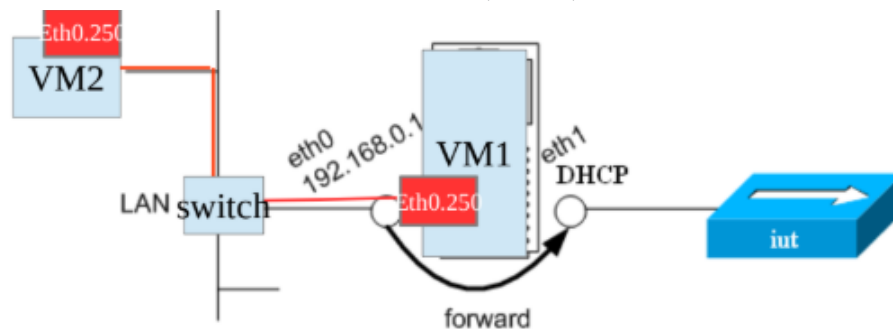
```
VM2
valid_lft forever preferred_lft forever
root@debianVM:~#
root@debianVM:~# curl 192.168.10.100

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Debian Default Page: It works</title>
    <style type="text/css" media="screen">
      *
      {
        margin: 0px 0px 0px 0px;
        padding: 0px 0px 0px 0px;
      }
    </style>
  </head>
  <body>
    <div style="text-align: center;>
      <img alt="Apache2 logo" data-bbox="484 798 514 828"/>
    </div>
  </body>
</html>
```

Ceci prouve que le client de la VM2 (192.168.200.1) a pu accéder au serveur web via le routage grâce à la commande `curl 192.168.10.100`

a.3 NAT

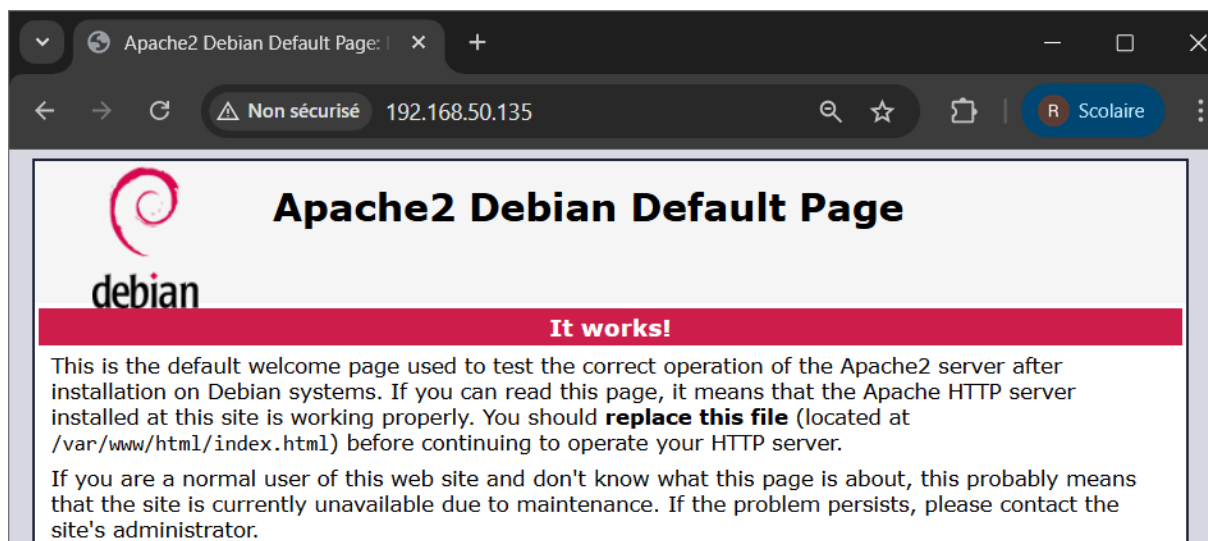
Pour rendre le serveur Web accessible depuis le réseau de l'IUT (Extérieur), nous avons mis en place une redirection de port (DNAT).



Règle de NAT sur la passerelle (VM1) : La règle redirige le trafic TCP entrant sur le port 80 de l'interface publique vers l'IP privée du serveur DMZ.

```
iptables -t nat -A PREROUTING -i enp0s3 -p tcp --dport 80 -j DNAT --to-destination 192.168.10.100:80
```

Le test a été effectué depuis le navigateur de la machine hôte (Windows) physique en utilisant l'adresse IP publique de VM1 qui est en 192.168.50.135.



L'accès au site depuis le navigateur de l'hôte physique, via l'IP publique du routeur, valide le fonctionnement de la redirection de port DNAT.

B - VLAN

Nous avons créé un Trunk entre VM1 et VM2 pour faire transiter un réseau virtuel tagué (VLAN 250) sur le câble existant.

Configuration réalisée :

- Installation du paquet vlan.
- Création de l'interface virtuelle enp0sX.250 sur les deux VMs.
- Attribution des adresses IP dans le sous-réseau 192.168.250.0/24.

```
5: enp0s8.250@enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noque
ue state UP group default qlen 1000
    link/ether 08:00:27:cf:3d:9f brd ff:ff:ff:ff:ff:ff
    inet 192.168.250.254/24 scope global enp0s8.250
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fecf:3d9f/64 scope link
        valid_lft forever preferred_lft forever
```

L'interface virtuelle pour le VLAN 250 est créée et active avec l'IP 192.168.250.254. VM1 est prête à router le trafic de ce segment réseau isolé.

Et sur la VM2 :

Créer l'interface VLAN : (Assure-toi que ton interface principale est bien enp0s3, sinon adapte le nom)

```
ip link add link enp0s3 name enp0s3.250 type vlan id 250
```

Lui donner une IP sur ce nouveau réseau :

```
ip addr add 192.168.250.1/24 dev enp0s3.250
```

Allumer l'interface :

```
ip link set enp0s3.250 up
```

```

VM2
root@debianVM:~# ip link add link enp0s3 name enp0s3.250 type vlan id 250
root@debianVM:~# ip addr add 192.168.250.1/24 dev enp0s3.250
root@debianVM:~# ip link set enp0s3.250 up
root@debianVM:~# ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group de
fault 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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state U
P group default qlen 1000
    link/ether 08:00:27:bb:d4:11 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.1/24 brd 192.168.200.255 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:febb:d411/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s3.250@enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noque
ue state UP group default qlen 1000
    link/ether 08:00:27:bb:d4:11 brd ff:ff:ff:ff:ff:ff
    inet 192.168.250.1/24 scope global enp0s3.250
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:febb:d411/64 scope link
        valid_lft forever preferred_lft forever
root@debianVM:~#

```

L'interface virtuelle VLAN 250 est bien créée sur VM2 avec l'IP 192.168.250.1, permettant la communication avec la passerelle sur ce segment réseau isolé.

Nous avons effectué un ping entre les deux machines sur le réseau VLAN de VM1 vers VM2 et capturé le trafic sur VM1 pour vérifier la présence du tag 802.1Q

```

VM2
root@debianVM:~# ping 192.168.250.254
PING 192.168.250.254 (192.168.250.254) 56(84) bytes of data.
64 bytes from 192.168.250.254: icmp_seq=1 ttl=64 time=1.38 ms
64 bytes from 192.168.250.254: icmp_seq=2 ttl=64 time=1.33 ms
64 bytes from 192.168.250.254: icmp_seq=3 ttl=64 time=1.32 ms
64 bytes from 192.168.250.254: icmp_seq=4 ttl=64 time=1.77 ms
64 bytes from 192.168.250.254: icmp_seq=5 ttl=64 time=2.15 ms

```

Le succès du ping vers 192.168.250.254 confirme la connectivité au sein du VLAN 250. Les trames taguées circulent correctement entre le client et la passerelle.

La commande `tcpdump -en -i enp0s8` depuis la VM1 a permis de capturer les paquets avec le texte vlan 250.

```
VM1
root@debianVM:~# tcpdump -en -i enp0s8 vlan
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
09:38:43.376879 08:00:27:bb:d4:11 > 08:00:27:cf:3d:9f, ethertype 802.1Q (0x8100), length 102: vlan 250, p 0, ethertype IPv4 (0x0800), 192.168.250.1 > 192.168.250.254: ICMP echo request, id 605, seq 28, length 64
09:38:43.376907 08:00:27:cf:3d:9f > 08:00:27:bb:d4:11, ethertype 802.1Q (0x8100), length 102: vlan 250, p 0, ethertype IPv4 (0x0800), 192.168.250.254 > 192.168.250.1: ICMP echo reply, id 605, seq 28, length 64
09:38:44.384031 08:00:27:bb:d4:11 > 08:00:27:cf:3d:9f, ethertype 802.1Q (0x8100), length 102: vlan 250, p 0, ethertype IPv4 (0x0800), 192.168.250.1 > 192.168.250.254: ICMP echo request, id 605, seq 29, length 64
09:38:44.384147 08:00:27:cf:3d:9f > 08:00:27:bb:d4:11, ethertype 802.1Q (0x8100), length 102: vlan 250, p 0, ethertype IPv4 (0x0800), 192.168.250.254 > 192.168.250.1: ICMP echo reply, id 605, seq 29, length 64
```

La capture `tcpdump` sur le routeur révèle l'étiquette 802.1Q ID 250 lors du ping, confirmant que le lien Trunk entre les VM est opérationnel.

Conclusion

Ce TD a permis de valider la configuration complète d'un routeur Linux. Nous avons réussi à segmenter le réseau en trois zones distinctes (LAN, WAN, DMZ) et à implémenter des VLANs. L'utilisation des fichiers `/etc/network/interfaces` garantit que notre infrastructure est robuste et redémarre automatiquement avec les bons paramètres.