

**Nom** : DOSSA

**Prénom** : Lophias

**Filière** : Master Cloud et Infrastructures 1

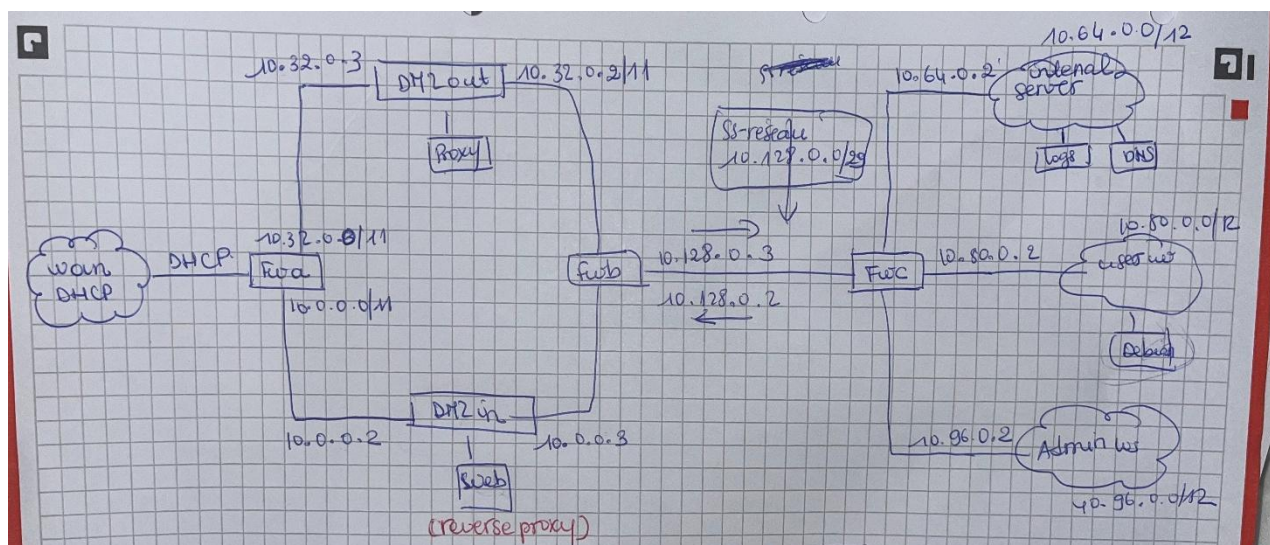
**Compte rendu**

**Sujet** : Mise en place d'une infrastructure sécurisée

Année : 2024-2025

## Introduction

Pour la réalisation de ce projet, j'ai tout d'abord personnalisé le schéma comme le montre le schéma ci-après :



J'ai donc placé le proxy (squid) au niveau du DMZ-out pour filtrer et contrôler les accès internet des utilisateurs internes. Le reverse proxy quant à lui est placé dans le DMZ-In pour la protection et le routage des requêtes vers les serveurs internes. Bien évidemment mon serveur web est placé dans le DMZ-In pour permettre aux utilisateurs d'y avoir accès sans entrer dans le réseau interne.

Pour ce TP, j'ai choisi de mettre le serveur interne dans internal servers. J'ai délibérément choisi d'y installer tous les services (DNS, Système de centralisation de Logs, Serveur web, Proxy et même reverse proxy). Ceci me permet de consommer moins de ressources au niveau de la machine principale. Nous verrons plus tard dans ce document comment la configuration a été faite.

La machine servant de poste client a été installée dans mon cas dans User WC en suivant les conventions demandées dans le document.

Résumé : Pour le déploiement de l'infrastructure, j'ai utilisé VmWare Station et au total 5 machines virtuelles. Une machine par firewall donc 3 au total ; une machine client et enfin une machine server.

## Mise en œuvre

### a- Installation des machines virtuelles

J'ai commencé à créer des VmNet sur VMware pour la connexion de chaque machine au sous-réseau correspondant.

Name	Type	External Connection	Host Connection	DHCP	Subnet Address
VMnet1	Host-only	-	Connected	Enabled	192.168.135.0
DMZ in	Host-only	-	Connected	-	10.0.0.0
DMZ out	Host-only	-	Connected	-	10.32.0.0
VMnet4	Host-only	-	Connected	-	10.128.0.0
server	Host-only	-	Connected	-	10.64.0.0
users	Host-only	-	Connected	-	10.80.0.0
admin	Host-only	-	Connected	Enabled	10.96.0.0
VMnet8	NAT	NAT	Connected	Enabled	192.168.172.0

### 1- Firewall A (fwa)

Ce firewall a trois interfaces :

- La première a le dhcp et est connecté à internet
- La deuxième est connecté au DMZ-in
- La troisième est connecté au DMZ-out

```

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto ens33
iface ens33 inet dhcp

#Interface DMZ out VMNET3
auto ens35
iface ens35 inet static
    address 10.32.0.3
    netmask 255.224.0.0
    post-up ip route add 10.64.0.0/12 via 10.32.0.10
    post-up ip route add 10.80.0.0/12 via 10.32.0.10
#Interface DMZ in VMNET2
auto ens34
iface ens34 inet static
    address 10.0.0.2
    netmask 255.224.0.0

```

## 2- Firewall B (fwb)

Nous avons également trois interfaces. Une qui est relié au DMZ-out, une autre au DMZ-in et la dernière interface au Firewall C

```

# The primary network interface
#Interface DMZIn
auto ens33
iface ens33 inet static
    address 10.0.0.3
    netmask 255.224.0.0
#InterfaceDMZout
auto ens34
iface ens34 inet static
    address 10.32.0.2
    netmask 255.224.0.0
#Interface fwc
auto ens35
iface ens35 inet static
    address 10.128.0.3
    netmask 255.255.255.248
#route par défaut pour joindre internal default, users wc, admin ws
post-up ip route add 10.64.0.0/12 via 10.128.0.3
post-up ip route add 10.80.0.0/12 via 10.128.0.3
post-up ip route add 10.96.0.0/12 via 10.128.0.3

```

## 3- Firewall C

Nous avons 4 interfaces ici : une pour le firewall b, trois autres pour internal servers, users WC & admin ws

```

#Interface vers fwb
auto ens33
iface ens33 inet static
    address 10.128.0.2
    netmask 255.255.255.248
    post-up ip route add 10.0.0.0/11 via 10.128.0.3
    post-up ip route add 10.32.0.0/11 via 10.128.0.3

#Interface Internal servers
auto ens34
iface ens34 inet static
    address 10.64.0.2
    netmask 255.240.0.0

#Interface Users WS
auto ens35
iface ens35 inet static
    address 10.80.0.2
    netmask 255.240.0.0

#Interface Admin WS
auto ens36
iface ens36 inet static
    address 10.96.0.2
    netmask 255.240.0.0

```

NB : Des routes ont été ajoutés pour chaque configuration pour permettre d'atteindre directement le réseau interne. J'ai pris le soin d'installer sur chacune des machines les paquets iptables car c'est majoritairement à leur niveau que s'écritont les prochaines règles de filtrage.

#### 4- Configuration du serveur unique

Comme énoncé un peu plus haut, mon serveur est unique. Il a cependant une seule interface. J'ai pris le soin de configurer trois alias pour la seule interface. Chaque alias sera activé dans mon projet pour se connecter soit :

- Au DMZ-out pour la gestion du proxy ;
- Au DMZ-in pour la gestion du site web et
- Au réseau 10.64.0.0/12 de Internal Server pour la gestion du DNS et du logs.

On pourra alors utiliser la commande `ifdown ens33 && ifup ens33 :0` par exemple pour se connecter à la DMZ-in et continuellement. Voici alors le rendu :

```

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#interface dmz-in -pour le server web
iface ens33:0 inet static
    address 10.0.0.10
    netmask 255.224.0.0
    gateway 10.0.0.2
#interface dmzout pour le proxy
iface ens33:1 inet static
    address 10.32.0.10
    netmask 255.224.0.0
    gateway 10.32.0.3
#interface pour dns et syslog
iface ens33:2 inet static
    address 10.64.0.10
    netmask 255.240.0.0
    gateway 10.64.0.2
    ip route add 10.80.0.0/12 via 10.64.0.2

```

Dans ma configuration on a trois gateway. On peut se le permettre car les trois interfaces ne fonctionneront pas au même moment. Lorsque l'une est allumée, les autres sont censé être éteintes car on change de VmNet au niveau du VMWare.

#### 5- Configuration du poste client

```

# The loopback network interface
auto lo
iface lo inet loopback

#conf pour l'interface du posteclient
auto ens36
iface ens36 inet static
    address 10.80.0.10
    netmask 255.240.0.0
    gateway 10.80.0.2
    dns-nameservers 10.64.0.10 8.8.8.8
    up ip route add 10.64.0.0/12 via 10.80.0.2

```

#### b- Quelques tests de connectivités

- ❖ Depuis le firewall a, je teste juste la connectivité vers le DMZ-in et le DMZ out

```

root@fwa:~# ping 10.32.0.3
PING 10.32.0.3 (10.32.0.3) 56(84) bytes of data.
64 bytes from 10.32.0.3: icmp_seq=1 ttl=64 time=0.071 ms
64 bytes from 10.32.0.3: icmp_seq=2 ttl=64 time=0.057 ms
64 bytes from 10.32.0.3: icmp_seq=3 ttl=64 time=0.054 ms
^C
--- 10.32.0.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 0.054/0.060/0.071/0.007 ms
root@fwa:~# ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.056 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.093 ms
^C
--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2037ms
rtt min/avg/max/mdev = 0.056/0.074/0.093/0.015 ms
root@fwa:~# _

```

❖ Depuis le firewall b, je teste la connectivité vers le firewall c

```

root@fwb:~# ping 10.128.0.3
PING 10.128.0.3 (10.128.0.3) 56(84) bytes of data.
64 bytes from 10.128.0.3: icmp_seq=1 ttl=64 time=9.39 ms
64 bytes from 10.128.0.3: icmp_seq=2 ttl=64 time=5.15 ms
64 bytes from 10.128.0.3: icmp_seq=3 ttl=64 time=0.047 ms
64 bytes from 10.128.0.3: icmp_seq=4 ttl=64 time=0.056 ms
^C
--- 10.128.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3029ms
rtt min/avg/max/mdev = 0.047/3.659/9.387/3.906 ms
root@fwb:~# ping 10.80.0.10

```

❖ Depuis server, je teste la connectivité avec le client

```

root@server:/etc/network# ping 10.80.0.10
PING 10.80.0.10 (10.80.0.10) 56(84) bytes of data.
64 bytes from 10.80.0.10: icmp_seq=1 ttl=63 time=3.11 ms
64 bytes from 10.80.0.10: icmp_seq=2 ttl=63 time=2.80 ms
64 bytes from 10.80.0.10: icmp_seq=3 ttl=63 time=3.80 ms
64 bytes from 10.80.0.10: icmp_seq=4 ttl=63 time=2.92 ms
^X64 bytes from 10.80.0.10: icmp_seq=5 ttl=63 time=2.10 ms
64 bytes from 10.80.0.10: icmp_seq=6 ttl=63 time=2.77 ms
^C
--- 10.80.0.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5013ms
rtt min/avg/max/mdev = 2.101/2.916/3.802/0.503 ms
root@server:/etc/network# _

```

❖ Depuis le client, je teste la connectivité avec le server

```

root@client:~# ping 10.64.0.10
PING 10.64.0.10 (10.64.0.10) 56(84) bytes of data.
64 bytes from 10.64.0.10: icmp_seq=1 ttl=63 time=2.83 ms
64 bytes from 10.64.0.10: icmp_seq=2 ttl=63 time=2.34 ms
64 bytes from 10.64.0.10: icmp_seq=3 ttl=63 time=3.70 ms
64 bytes from 10.64.0.10: icmp_seq=4 ttl=63 time=4.31 ms
^C
--- 10.64.0.10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 2.337/3.292/4.305/0.761 ms
root@client:~# █

```

Il reste plusieurs connectivités à tester. Par exemple on doit tester la connectivité du server avec la DMZ-in ; la connectivité du client avec le DMZ-out puisque ce sont des zones où on va agir en fonction de si on veut tester le DNS, le proxy, ou le server. Ces tests ont été réalisés en aval de ce rapport et la connectivité est bien présente. Je saute donc ces étapes et je passe à l'installation des services sur mon server unique.

### c- Installation des services sur le serveur unique.

#### 1- Squid (Proxy)

En faisant nano /etc/squid/squid.conf, j'ai rajouté juste cette ligne

```
#acl localnet src 10.0.0.0/10 # net 4291 link local (directly plugged) machines  
acl localnet src 10.64.0.0/12 10.80.0.0/12 10.96.0.0/12
```

Cette ligne demande de filtrer tout ce qui sort du réseau interne. Pour le reste des autres instructions du fichier, j'ai juste décommenté les lignes.

```
#Autorise le trafic depuis les réseaux autorisés  
http_access allow localnet  
  
#bloquer tout le reste  
http_access deny all  
  
#spécifier l'interface et le port d'écoute  
http_port 10.32.0.10:3128  
  
#configuration du cache  
cache_dir ufs /var/spool/squid 100 16 256
```

Après enregistrement du fichier, et après avoir démarrer Squid par la commande systemctl restart squid. Celui-ci est bien actif.

```
root@server:/etc/squid# systemctl status squid  
● squid.service - Squid Web Proxy Server  
   Loaded: loaded (/lib/systemd/system/squid.service; enabled; preset: enabled)  
   Active: active (running) since Mon 2025-02-10 22:23:22 CET; 10s ago  
     Docs: man:squid(8)  
  Process: 5587 ExecStartPre=/usr/sbin/squid --foreground -z (code=exited, status=0/SUCCESS)  
    Main PID: 5590 (squid)  
      Tasks: 4 (limit: 1056)  
     Memory: 16.1M  
        CPU: 5.808s  
    CGroup: /system.slice/squid.service  
            └─5590 /usr/sbin/squid --foreground -sYC  
              └─5592 "(squid-1)" --kid squid-1 --foreground -sYC  
                └─5593 "(logfile-daemon)" /var/log/squid/access.log  
                  └─5594 "(pinger)"  
  
févr. 10 22:23:21 server squid[5592]: Using Least Load store dir selection  
févr. 10 22:23:21 server squid[5592]: Set Current Directory to /var/spool/squid  
févr. 10 22:23:22 server squid[5592]: Finished loading MIME types and icons.  
févr. 10 22:23:22 server squid[5592]: HTCP Disabled.  
févr. 10 22:23:22 server squid[5592]: Pinger socket opened on FD 14  
févr. 10 22:23:22 server squid[5592]: Squid plugin modules loaded: 0  
févr. 10 22:23:22 server squid[5592]: Adaptation support is off.  
févr. 10 22:23:22 server squid[5592]: Accepting HTTP Socket connections at conn3 local=[::]:3128 remote=[::] FD 12 flags=9  
févr. 10 22:23:22 server systemd[1]: Started squid.service - Squid Web Proxy Server.  
févr. 10 22:23:22 server squid[5592]: storeLateRelease: released 0 objects  
root@server:/etc/squid#
```

#### 2- Rsyslog (pour la centralisation des logs)



RSYSLOG est la solution choisie pour centraliser les logs sur notre infras réseau. Je l'ai installé et il est temps de vérifier que l'installation s'est bien déroulée.

```
root@server:/# dpkg -l | grep rsyslog
ii rsyslog                        8.2302.0-1          amd64          reliable system and kernel logging daemon
root@server:/#
```

Maintenant que l'on sait qu'il est bien présent, nous allons effectuer quelques configurations supplémentaires.

Il est important d'activer l'écoute sur TCP, UDP en allant dans le fichier rsyslog.conf

```
#Activer l'écoute sur UDP
module(load="imudp")
input(type="imudp" port="514")

#Activer l'écoute sur TCP
module(load="imtcp")
input(type="imtcp" port="514")
```

Activation de rsyslog

```
root@server:/etc# systemctl restart rsyslog
root@server:/etc# systemctl status rsyslog
• rsyslog.service - System Logging Service
   Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-02-07 19:05:34 CET; 13s ago
     TriggeredBy: • syslog.socket
    Docs: man:rsyslogd(8)
          man:rsyslog.conf(5)
          https://www.rsyslog.com/doc/
   Main PID: 1188 (rsyslogd)
     Tasks: 10 (limit: 1056)
    Memory: 1.3M
       CPU: 19ms
   CGroup: /system.slice/rsyslog.service
           └─1188 /usr/sbin/rsyslogd -n -iNONE

févr. 07 19:05:34 server systemd[1]: Starting rsyslog.service - System Logging Service...
févr. 07 19:05:34 server systemd[1]: Started rsyslog.service - System Logging Service.
févr. 07 19:05:34 server rsyslogd[1188]: imuxsock: Acquired UNIX socket '/run/systemd/journal/syslog' (fd 3) from systemd. [v8.2302.0]
févr. 07 19:05:34 server rsyslogd[1188]: [origin software="rsyslogd" swVersion="8.2302.0" x-pid="1188" x-info="https://www.rsyslog.com"] start
root@server:/etc#
```

### 3- Bind9 (DNS)

Après installation du paquet, il faut configurer le fichier named.conf

```
zone "internal.lan" {
    type master;
    file "/etc/bind/db.internal.lan";
}
```

Configuration du fichier db.internal.lan

```
$TTL      604800
@         IN      SOA      ns.internal.lan. admin.internal.lan. (
                                1          ; Serial
                                604800     ; Refresh
                                86400      ; Retry
                                2419200    ; Expire
                                604800 )   ; Negative Cache TTL
;
@         IN      NS       ns.internal.lan.
ns        IN      A        10.64.0.10
```

Le premier fichier me permet de définir les zones où le serveur DNS doit gérer et trouver les enregistrements.

Le second fichier « **db.internal.lan** » me permet de définir les correspondances entre noms de domaine et IP pour la zone. Ici c'est 10.64.0.10 parce que notre DNS se situe dans internal servers.

Après activation du bind9, j'ai cet écran :

```
• named.service - BIND Domain Name Server
  Loaded: loaded (/lib/systemd/system/named.service; enabled; preset: enabled)
  Active: active (running) since Mon 2025-02-10 22:36:37 CET; 34s ago
    Docs: man:named(8)
  Main PID: 5607 (named)
    Status: "running"
     Tasks: 5 (limit: 1056)
  Memory: 11.3M
     CPU: 575ms
  CGroup: /system.slice/named.service
          └─5607 /usr/sbin/named -f -u bind
```

Je teste directement sur le serveur et cela semble concluant et sans erreur.

```
root@server:/etc/bind# nslookup ns.internal.lan 127.0.0.1
Server:          127.0.0.1
Address:         127.0.0.1#53

Name:   ns.internal.lan
Address: 10.64.0.10

root@server:/etc/bind# nslookup ns.internal.lan 127.0.0.1
Server:          127.0.0.1
Address:         127.0.0.1#53

Name:   ns.internal.lan
Address: 10.64.0.10
```

#### 4- Apache2

Pour la configuration d'Apache, je n'ai pas eu grand-chose à faire à part installer les paquets et à le démarrer.

```
root@server:/etc# systemctl status apache2
• apache2.service - The Apache HTTP Server
  Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
  Active: active (running) since Mon 2025-02-10 22:46:40 CET; 9s ago
    Docs: https://httpd.apache.org/docs/2.4/
  Process: 5620 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
  Main PID: 5632 (apache2)
     Tasks: 55 (limit: 1056)
  Memory: 14.9M
     CPU: 647ms
  CGroup: /system.slice/apache2.service
          └─5632 /usr/sbin/apache2 -k start
            └─5633 /usr/sbin/apache2 -k start
              └─5634 /usr/sbin/apache2 -k start

févr. 10 22:46:39 server systemd[1]: Starting apache2.service - The Apache HTTP Server...
févr. 10 22:46:40 server apachectl[5631]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message
févr. 10 22:46:40 server systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Pour être sûr que ça fonctionne comme je le souhaite, j'ai switché l'interface de mon serveur web dans le réseau du DMZ-In puis j'ai testé la commande suivante sur mon serveur curl <http://10.0.0.10>. Voici le résultat obtenu :

```

<div class="section_header">
  <div id="docroot"></div>
  Document Roots
</div>

<div class="content_section_text">
  <p>
    By default, Debian does not allow access through the web browser to
    <em>any</em> file apart of those located in <tt>/var/www</tt>,
    <a href="http://httpd.apache.org/docs/2.4/mod/mod_userdir.html" rel="nofollow">public_html</a>
    directories (when enabled) and <tt>/usr/share</tt> (for web
    applications). If your site is using a web document root
    located elsewhere (such as in <tt>/srv</tt>) you may need to whitelist your
    document root directory in <tt>/etc/apache2/apache2.conf</tt>.
  </p>
  <p>
    The default Debian document root is <tt>/var/www/html</tt>. You
    can make your own virtual hosts under /var/www. This is different
    to previous releases which provides better security out of the box.
  </p>
</div>

<div class="section_header">
  <div id="bugs"></div>
  Reporting Problems
</div>
<div class="content_section_text">
  <p>
    Please use the <tt>reportbug</tt> tool to report bugs in the
    Apache2 package with Debian. However, check <a
    href="http://bugs.debian.org/cgi-bin/pkgreport.cgi?ordering=normal;archive=0;src=apache2;repeatmerged=0"
    rel="nofollow">existing bug reports</a> before reporting a new bug.
  </p>
  <p>
    Please report bugs specific to modules (such as PHP and others)
    to respective packages, not to the web server itself.
  </p>
</div>

</div>
</div>
<div class="validator">
</div>
</body>
</html>

```

J'ai un retour du contenu de ma page web, ce qui prouve que cela fonctionne comme je l'espérait.

## 5- Configuration du reverse-proxy sur apache2

Apache2 me permet de configurer le reverse-proxy. Cette configuration me permet **cacher l'identité du serveur** aux utilisateurs externes et d'optimiser les requêtes étant donné que nous sommes dans le cadre d'une infra sécurisée.

Pour ce faire, j'ai créé un fichier nano /etc/apache2/sites-available/reverse-proxy.conf

```

<VirtualHost *:8080>

    ProxyPreserveHost On
    ProxyPass "/" http://10.0.0.10/
    ProxyPassReverse "/" http://10.0.0.10/

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

```

J'ai réutilisé l'adresse du server car c'est sur celui-ci qu'on configure le reverse-proxy.

**NB** : Pour des raisons de délais de rendu, je n'ai pas pu tester, le squid, le DNS, et le reverse-proxy. Ce rapport ne pourra donc pas en faire mention.

#### d- Les iptables

Contenu du délai imparti, je n'ai pas pu mettre en place les iptables dans mon projet. Mais ayant effectué un travail personnel avant la rédaction de ce rendu, j'ai pu écrire toutes les règles en me servant de la matrice des autorisations de trafic fournie à la page 3 du document Infrastructure réseau sécurisée.pdf.

#### **Règles WAN**

##### **Interdire WAN ↔ WAN**

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

##### **WAN → DMZ In**

```
iptables -A FORWARD -i enp0s3 -o enp0s8 -p tcp --dport 80 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s3 -o enp0s8 -p tcp --dport 443 -j ACCEPT
```

##### **WAN → DMZ Out (Autoriser accès au proxy Squid)**

```
iptables -A FORWARD -i enp0s3 -o enp0s9 -p tcp --dport 3128 -j ACCEPT
```

##### **WAN → Internal Servers (Interdire totalement)**

```
iptables -A FORWARD -i enp0s3 -o enp0s10 -j DROP
```

##### **WAN → Users WS (Interdire totalement)**

```
iptables -A FORWARD -i enp0s3 -o enp0s11 -j DROP
```

#### **Règles DMZ In**

##### **Interdire DMZ In ↔ DMZ In**

```
iptables -A FORWARD -i enp0s8 -o enp0s8 -j DROP
```

##### **DMZ In → WAN (Interdire tout sauf réponse aux connexions HTTP/HTTPS)**

```
iptables -A FORWARD -i enp0s8 -o enp0s3 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
iptables -A FORWARD -i enp0s8 -o enp0s3 -j DROP
```

##### **DMZ In → DMZ Out (Autoriser le serveur web à contacter le proxy)**

```
iptables -A FORWARD -i enp0s8 -o enp0s9 -p tcp --dport 3128 -j ACCEPT
```

##### **DMZ In → Internal Servers (Autoriser DNS et logs)**

```
iptables -A FORWARD -i enp0s8 -o enp0s10 -p udp --dport 53 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s8 -o enp0s10 -p udp --dport 514 -j ACCEPT
```

### **DMZ In → Users WS (Interdire totalement)**

```
iptables -A FORWARD -i enp0s8 -o enp0s11 -j DROP
```

## **Règles DMZ-out**

### **Interdire DMZ Out ↔ DMZ Out**

```
iptables -A FORWARD -i enp0s9 -o enp0s9 -j DROP
```

### **DMZ Out → WAN (Interdire tout sauf le trafic HTTP via le proxy)**

```
iptables -A FORWARD -i enp0s9 -o enp0s3 -p tcp --dport 80 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s9 -o enp0s3 -p tcp --dport 443 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s9 -o enp0s3 -j DROP
```

### **DMZ Out → DMZ In (Interdire sauf communication Proxy → Web Server)**

```
iptables -A FORWARD -i enp0s9 -o enp0s8 -p tcp --dport 80 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s9 -o enp0s8 -p tcp --dport 443 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s9 -o enp0s8 -j DROP
```

### **DMZ Out → Internal Servers (Autoriser accès DNS uniquement)**

```
iptables -A FORWARD -i enp0s9 -o enp0s10 -p udp --dport 53 -j ACCEPT
```

### **DMZ Out → Users WS (Interdire totalement)**

```
iptables -A FORWARD -i enp0s9 -o enp0s11 -j DROP
```

## **Règles Internal Servers**

### **Interdire Internal Servers ↔ Internal Servers**

```
iptables -A FORWARD -i enp0s10 -o enp0s10 -j DROP
```

### **Internal Servers → WAN (Interdire tout accès direct à Internet)**

```
iptables -A FORWARD -i enp0s10 -o enp0s3 -j DROP
```

### **Internal Servers → DMZ In (Autoriser DNS et logs)**

```
iptables -A FORWARD -i enp0s10 -o enp0s8 -p udp --dport 53 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s10 -o enp0s8 -p udp --dport 514 -j ACCEPT
```

### **Internal Servers → DMZ Out (Autoriser requêtes DNS vers le proxy)**

```
iptables -A FORWARD -i enp0s10 -o enp0s9 -p udp --dport 53 -j ACCEPT
```

### **Internal Servers → Users WS (Autoriser communication des logs)**

```
iptables -A FORWARD -i enp0s10 -o enp0s11 -p udp --dport 514 -j ACCEPT
```

## **Règles Admin WS**

### **Interdire Admin WS ↔ Admin WS**

```
iptables -A FORWARD -i enp0s12 -o enp0s12 -j DROP
```

### **Admin WS → WAN (Interdire sauf mises à jour Debian)**

```
iptables -A FORWARD -i enp0s12 -o enp0s3 -p tcp --dport 80 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s12 -o enp0s3 -p tcp --dport 443 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s12 -o enp0s3 -j DROP
```

### **Admin WS → Internal Servers**

```
iptables -A FORWARD -i enp0s12 -o enp0s10 -p tcp --dport 22 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s12 -o enp0s10 -p udp --dport 514 -j ACCEPT
```

### **Admin WS → DMZ In**

```
iptables -A FORWARD -i enp0s12 -o enp0s8 -p tcp --dport 22 -j ACCEPT
```

### **Admin WS → Users WS**

```
iptables -A FORWARD -i enp0s12 -o enp0s11 -p tcp --dport 3389 -j ACCEPT
```

```
iptables -A FORWARD -i enp0s12 -o enp0s11 -p tcp --dport 5900 -j ACCEPT
```