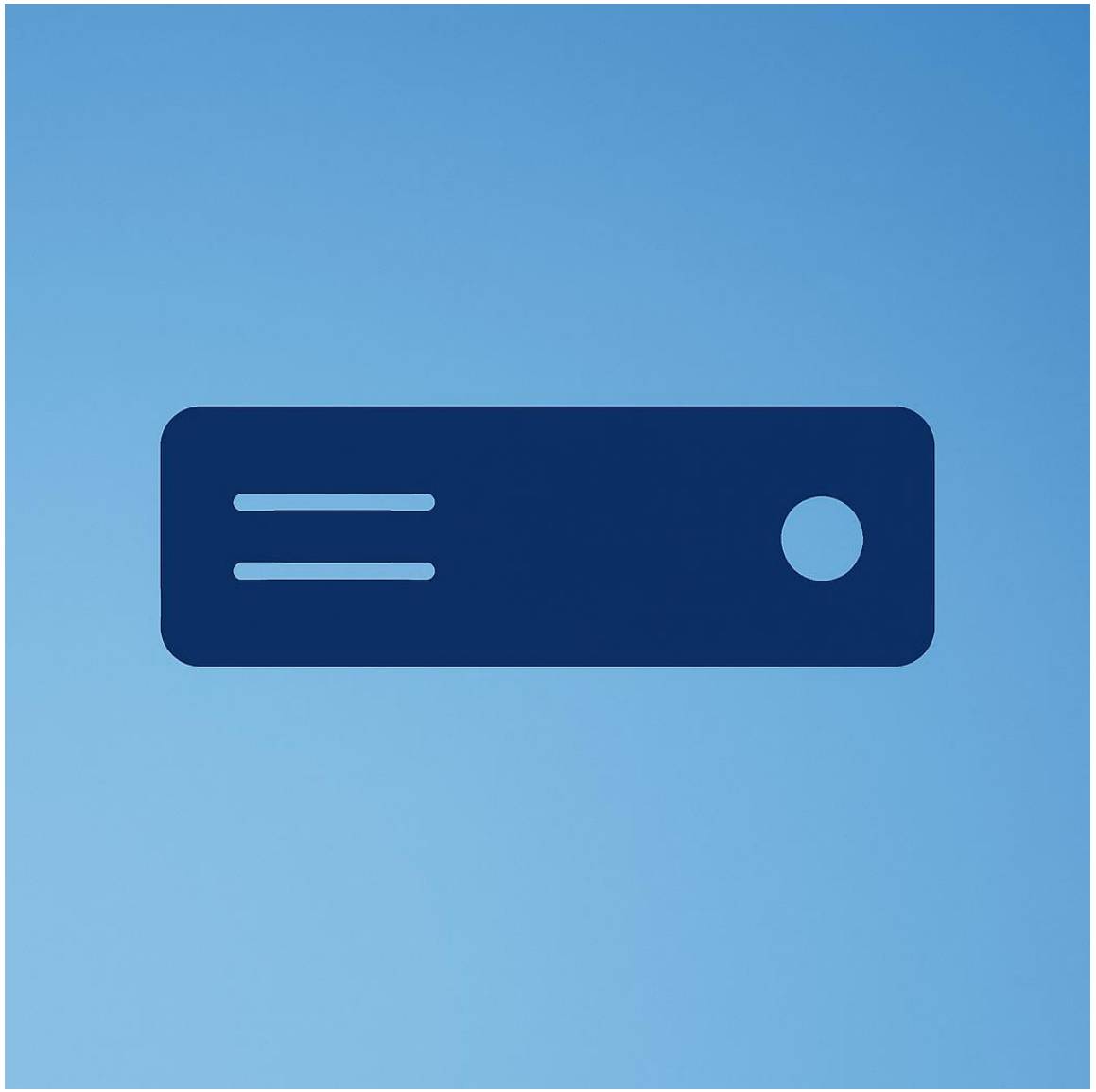
Gobert Louis
Ramia Ryan

Mise en place de Vlans et d'hyperviseur de type 1



Sommaire

Présentation de projet

Une segmentation réseau via VLANs pour isoler différents postes/VMs

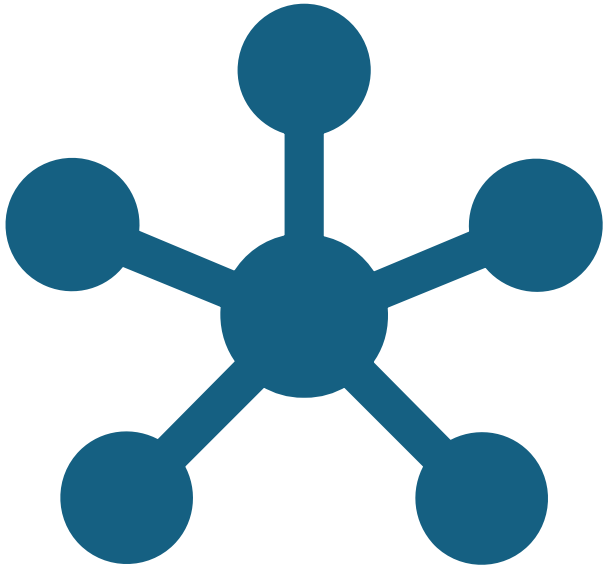
Un hyperviseur de type 1 (Proxmox) installé sur un serveur physique

Des machines virtuelles (VMs) configurées selon les rôles définis

Un plan IP clair et fonctionnel

Des tests de connectivité pour vérifier l'isolement, l'accès aux services et la cohérence du réseau

Schéma de la salle



- Plusieurs **serveurs Proxmox (18-a à 18-d)** connectés en LACP (agrégation de liens RJ45 à 3 Gb/s) à un **switch central**.

Ce switch distribue le réseau vers :

- Des **NAS (18-b & 18-e)** pour le stockage en LACP aussi
- Le **pare-feu SN510** qui gère l'accès à Internet
- Un point d’accès Wi-Fi pour les clients sans fil
- Des **prises murales** pour les postes clients
- Une **baie pédagogique** connectée aux VLANs définis

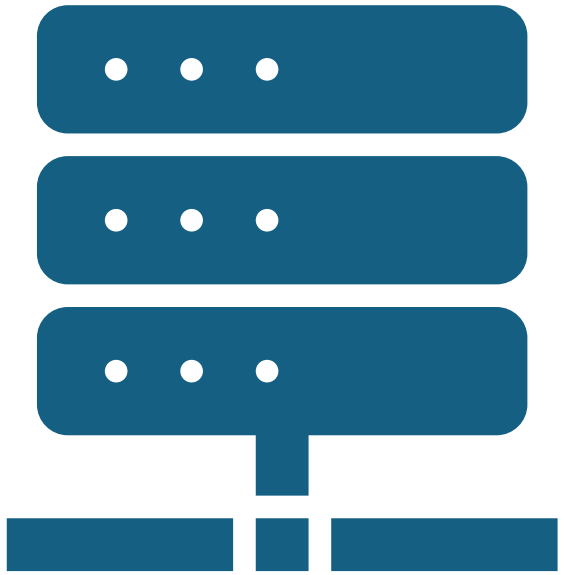
Schéma logique / Réseau & VLANs

- Le trafic est séparé en plusieurs VLANs pour isoler les différents environnements pédagogiques.

Chacun a son propre sous-réseau IP, ce qui permet de :

- Sécuriser et segmenter le trafic
- Simuler des entreprises ou structures distinctes

(notre Ipv4 sera dans la plage IP 10.140.140.0/24 dans le VLAN 140)



VLAN	NOM	SOUS-RÉSEAU IP
100	Hepturing	10.100.100.0/24
110	JL-Network	10.110.110.0/24
120	DSL-Network	10.120.120.0/24
130	Lexanil	10.130.130.0/24
140	Orion	10.140.140.0/24

Segmentation Réseau & VLANs

- **Dans notre infrastructure o**n a **plusieurs VLANs** correspondant à différents projets ou groupes :

Avantages :

•Meilleure **sécurité** : les groupes ne se voient pas
•Meilleure **organisation** : chaque projet est bien séparé
•**Isolation + contrôle + performance**

Segmentation Réseau & VLANs (Sécurité et SSH)

```
# configure terminal
# line console 0
# password Orionconsole20*
# login
# exit

# enable secret Orionthebestinfo20*

# no cdp run
# no lldp run
```

Sécurisation de base

Activer ssh

```
# hostname Switch_Orion
# ip domain-name monreseau.local
# crypto key generate rsa
# username admin secret Orionconsole20*
# line vty 0 4
# login local
# transport input ssh
# exit
```

Segmentation Réseau & VLANs (port Gi 1/0/2)

- **Port Gi1/0/1 → Prise murale (trunk)**
- Ce port relie le **switch à la prise murale**, donc il doit **transporter tous les VLANs** pour que la machine branchée choisisse le bon.
-
- Trunk = tous les VLANs peuvent passer
- Sécurité : 1 seule adresse MAC → évite les switchs non autorisés

Segmentation Réseau & VLANs (Port Gi 1/0/1)

- Le port Gi1/0/2 est directement connecté à **l'hyperviseur Proxmox**. Il est configuré en mode **access**, car on veut qu'il soit **dans un seul VLAN (par ex : VLAN 140)**.
- Access = VLAN fixe (VLAN 140 ici)
- MAC max : 10 adresses → permet à plusieurs VMs de sortir via ce port

Résumé

Option	Description
maximum 1/10	Nombre de MAC autorisées
violation restrict	Bloque l'excès, mais sans couper le port
mac-address sticky	Mémorise automatiquement les MAC

- Le switch **gère les VLANs** selon la machine branchée
- Le port trunk permet au serveur de "voir" tous les VLANs
- Les ports access sont **isolés et sécurisés**

Installation hyperviseur de type 1

- Préparation de Debian
- Mise à jour et prérequis
- Ajout du repo Proxmox VE
- Installer Proxmox VE
- Config réseau pour Proxmox
- Accéder à Proxmox Web UI

Préparation du Debian 12 et configuration réseau

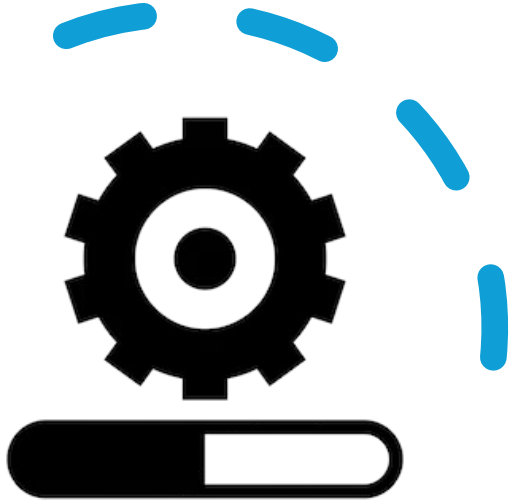
```
# sudo nano /etc/network/interfaces

auto lo
iface lo inet loopback

auto eno1
iface eno1 inet manual

auto vmbr0
iface vmbr0 inet static
    address 10.140.140.20/24
    gateway 10.140.140.254
    bridge-ports eno1
    bridge_stp off
    bridge_fd 0
    dns-nameservers 10.140.140.2

# sudo systemctl restart networking
```

- Debian installée sur la tour via une clé booté et branché sur le réseau sur le VLAN 140.
- Interface réseau configurée + reboot

+ Ajout du DNS de l'AD (voir page)

Mise à jour et prérequis

- Afin de garantir la sécurité des données, la sécurisation et le bon fonctionnement de l'appareil, ainsi que la compatibilité des programmes et du système



```
root@debian-orion:~#
    # sudo apt update && sudo apt upgrade -y
    # apt install wget curl gnupg -y
```

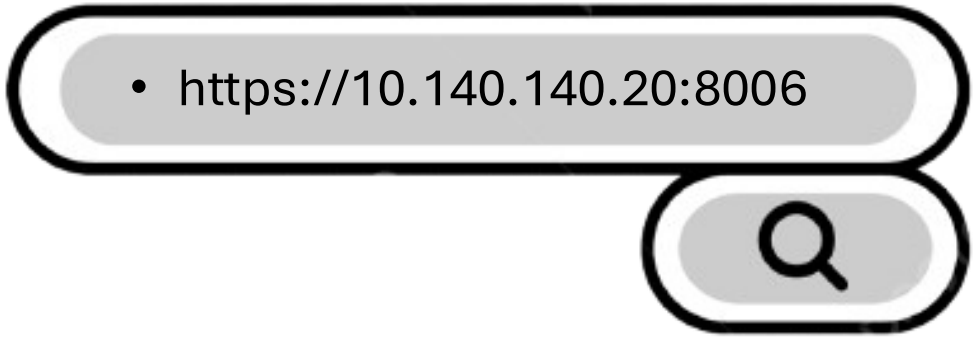
Ajout du repo et Installation de Proxmox VE

- On installe le repo de Proxmox VE, met jour etc…
- On lui donne la signature officielle (clé GPG)

```
# echo "deb http://download.proxmox.com/debian/pve bookworm pve-no-subscription" | sudo tee /etc/apt/sources.list.d/pve-install.list

# wget https://enterprise.proxmox.com/debian/proxmox-release-bookworm.gpg -O- | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/proxmox-release-bookworm.gpg

# sudo apt update
# apt install proxmox-ve postfix open-iscsi -y
```

Accéder à Proxmox Web UI

- Depuis un PC du même réseau, on entre l'ip avec le bon port dans la barre de recherche d'un navigateur web

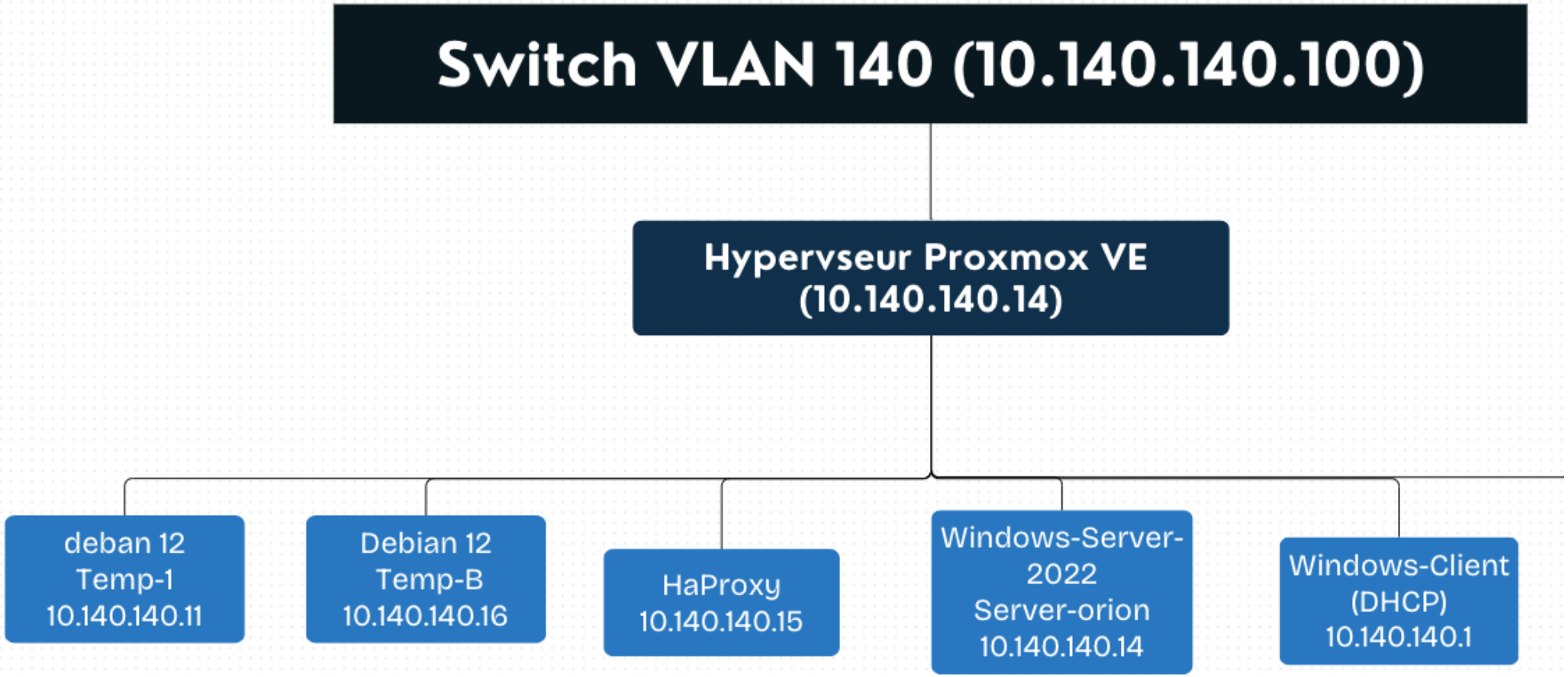


- Identifiants de connexions à sauvegarder dans un KeyPass (identification + sécurisée avec 2FA

Configuration des VMs

Nom VM	IP	Rôle/Fonction	OS / ISO à utiliser	Pourquoi ?
debian-lamp-a	10.140.140.11	Serveur applicatif/Lamp-A	Debian 12 (debian-12.iso)	Serveur pour héberger un site ou un service
debian-lamp-b	10.140.140.16	Serveur applicatif/Lamp-B	Debian 12 (debian-12.iso)	Serveur pour héberger un site ou un service
haproxy	10.140.140.15	Reverse proxy / Load balancer	Debian 12 (debian-12.iso)	Pour répartir le trafic entre plusieurs serveurs, sécuriser et centraliser l'accès, tester du load balancing ou des redirections HTTPS
win-server01	10.140.140.14	Contrôleur de domaine, DHCP et RODC	Windows Server 2022 (fr_windows_server_2022.iso)	Gère le RODC et DHCP
client-win10	DHCP (10.140.140.1)	Poste client de test	Windows 11 (fr_windows_11.iso)	Simule un poste utilisateur du réseau

Schéma logique de l'infrastructure



```
root@debian-orion:~# ping 10.140.140.3
PING 10.140.140.3 (10.140.140.3) 56(84) bytes of data.
64 bytes from 10.140.140.3: icmp_seq=1 ttl=64 time=0.205 ms
64 bytes from 10.140.140.3: icmp_seq=2 ttl=64 time=0.157 ms
```

```
root@debian-orion:~# ping 10.140.140.11
PING 10.140.140.11 (10.140.140.11) 56(84) bytes of data.
64 bytes from 10.140.140.11: icmp_seq=1 ttl=64 time=0.109 ms
64 bytes from 10.140.140.11: icmp_seq=2 ttl=64 time=0.047 ms
```

```
root@debian-orion:~# ping 10.140.140.14
PING 10.140.140.14 (10.140.140.14) 56(84) bytes of data.
64 bytes from 10.140.140.14: icmp_seq=1 ttl=128 time=0.613 ms
64 bytes from 10.140.140.14: icmp_seq=2 ttl=128 time=0.899 ms
```

```
root@debian-orion:~# ping 10.140.140.15
PING 10.140.140.15 (10.140.140.15) 56(84) bytes of data.
64 bytes from 10.140.140.15: icmp_seq=1 ttl=64 time=0.140 ms
64 bytes from 10.140.140.15: icmp_seq=2 ttl=64 time=0.051 ms
```

```
root@debian-orion:~# ping 10.140.140.16
PING 10.140.140.16 (10.140.140.16) 56(84) bytes of data.
64 bytes from 10.140.140.16: icmp_seq=1 ttl=64 time=0.126 ms
64 bytes from 10.140.140.16: icmp_seq=2 ttl=64 time=0.050 ms
```

Test Interconnection - Accès aux VMs depuis le serveur Proxmox

(10.140.140.20)

Accès VM Debian (SSH)

```
root@debian-orion:~# ssh root@10.140.140.11
root@10.140.140.11's password:
Linux Serveur-Lamp-A 6.8.12-11-pve #1 SMP PREEMPT_DYNAMIC PMX 6.8.12-11 (2025-05-22T09:39Z
) 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.
You have new mail.
Last login: Mon Jun 30 10:19:00 2025
root@Serveur-Lamp-A:~#
```

```
root@debian-orion:~# ssh root@10.140.140.16
root@10.140.140.16's password:
Linux Serveur-Lamp-B 6.8.12-11-pve #1 SMP PREEMPT_DYNAMIC PMX 6.8.12-11 (2025-05-22T09:39Z
) 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.
Last login: Mon Jun 30 10:15:20 2025
root@Serveur-Lamp-B:~#
```

Test isolement VLAN 140 depuis un Windows client

Windows client vers Proxmox salle-18 (hors VLAN 140)

```
C:\Users\Client>ping 10.0.85.231

Envoi d'une requête 'Ping'  10.0.85.231 avec 32 octets de données :
Délai d'attente de la demande dépassé.
Délai d'attente de la demande dépassé.
```

Windows client vers Windows-server principal

```
C:\Users\Client>ping 10.140.140.2

Envoi d'une requête 'Ping'  10.140.140.2 avec 32 octets de données :
Réponse de 10.140.140.2 : octets=32 temps<1ms TTL=128
Réponse de 10.140.140.2 : octets=32 temps<1ms TTL=128
```

En utilisant curl → redirige automatiquement vers temp-a ou temp-b (en l'occurrence, temp-a)

```
root@HAProxy-Orion:~# curl http://10.140.140.15
<!DOCTYPE html>
<html lang="fr-FR">
<head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1" />
<meta name='robots' content='noindex, nofollow' />
        <style>img:is([sizes="auto" i], [sizes^="auto," i]) { contain-intrinsic-size: 3000
px 1500px }</style>
        <title>Site Web Orion</title>
<link rel="alternate" type="application/rss+xml" title="Site Web Orion &raquo; Flux" href=
"http://10.140.140.15/index.php/feed/" />
<link rel="alternate" type="application/rss+xml" title="Site Web Orion &raquo; Flux des co
mmentaires" href="http://10.140.140.15/index.php/comments/feed/" />
<script>
```

On peut aussi tester avec un navigateur : http://10.140.140.15

Installation serveur web

Installation LAMP (Linux + Apache + MariaDB + PHP)

```
apt update && apt install apache2 mariadb-server php php-mysql libapache2-mod-php -y
```

INSTALLATION DE WORDPRESS (A & B)

```
cd /var/www/html
rm index.html
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz
mv wordpress/* .
```

```
chown -R www-data:www-data /var/www/html
```

Crée MariaDB sur A

```
mysql -u root
CREATE DATABASE wordpress;
CREATE USER 'wpuser'@'%' IDENTIFIED BY
'WordpressOrion22*';
GRANT ALL PRIVILEGES ON wordpress.* TO
'wpuser'@'%';
FLUSH PRIVILEGES;
EXIT;
```

SYNCHRONISATION ET BASE DE DONNÉES COMMUNE

Solution simple :

`lamp-b` monte un dossier NFS exporté depuis `lamp-a`

- `wp-content` synchronisé pour que les fichiers (uploads) soient communs
- Sinon : `rsync` toutes les 10 secondes via cron (si t'as pas NFS)

- DB sur un seul serveur (ex : lamp-a)
- temp-b se connecte à lamp-a via IP interne :
- php

Sur **A et B**, dans /var/www/html/wp-config.php

```
define('DB_HOST', '10.140.140.11'); // ou .16 selon où est la base
```

Synchronisation des fichiers Wordpress

Sur Lamp-A, fais un rsync vers Lamp-B

```
rsync -az --delete /var/www/html/ root@10.140.140.16:/var/www/html/
```

Ensuite, crée un **cron** sur Temp A pour le faire auto

```
crontab -e
```

```
*/2 * * * * rsync -az --delete /var/www/html/ root@10.140.140.16:/var/www/html/
```

Ça synchronise toutes les 2 minutes.

Rendre Temp B fonctionnel (avec la même base)

Sur Temp A :

```
mysqldump -u root -p wordpress > /tmp/wp.sql
```

Sur Temp B :

```
scp root@10.140.140.11:/tmp/wp.sql /tmp/
mysql -u root -p
```

```
mysql -u root
CREATE DATABASE wordpress;
CREATE USER 'wpuser'@'%' IDENTIFIED BY
'WordpressOrion22*';
GRANT ALL PRIVILEGES ON wordpress.* TO
'wpuser'@'%';
FLUSH PRIVILEGES;
EXIT;
```

```
mysql -u root -p wordpress < /tmp/wp.sql
```

Installation HaProxy

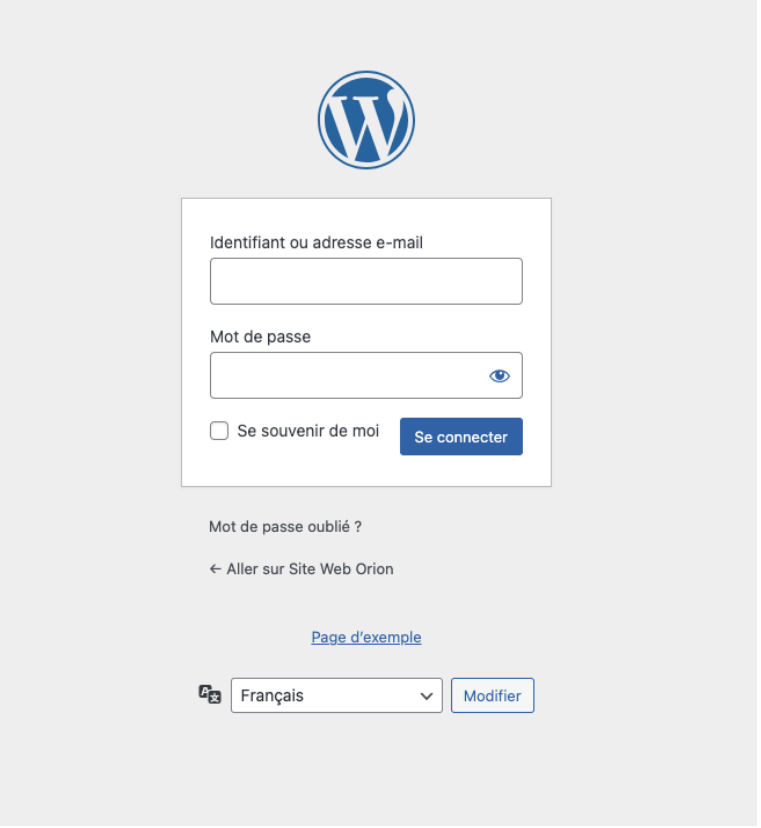
Dans le fichier /etc/haproxy/haproxy.cfg

```
frontend http-in
    bind *:80
    default_backend wordpress-nodes

backend wordpress-nodes
    balance roundrobin
    option httpchk GET /
    server temp-a 10.140.140.11:80 check
    server temp-b 10.140.140.16:80 check
```

```
systemctl restart haproxy
```

ACCÈS ADMIN

- HAProxy redirige tout vers /wp-admin
- Mais pour accéder directement à temp-a ou temp-b :
- Utilise IP directe : http://10.140.140.11/wp-admin ou ...16/wp-admin
- Désactive redirection WordPress dans wp-config.php si besoin



RODC et DHCP

Objectif:

- Avoir un **contrôleur de domaine secondaire** en lecture seule (RODC) sur 10.140.140.14
- Servir des adresses IP avec un **serveur DHCP local** sur 10.140.140.14.
- Améliorer la sécurité et la résilience dans le VLAN 140.

- 10.140.140.2 → Windows Server avec AD principal + DNS (domaine : oasis.local)
- 10.140.140.14 → Windows Server secondaire, sans AD à l'origine

Test DHCP sur Windows Client

Bonus

OBJECTIF :

- Simuler des utilisateurs qui accèdent à ton site (via HAProxy)
- Vérifier si HAProxy **bascule / répartit** bien le trafic entre les deux serveurs
- **Visualiser** sur les logs ou sur l'interface qui répond (A ou B)

OUTILS UTILISÉS :

1. Apache Benchmark (ab) pour tester la charge
2. Logs Apache ou Nginx sur les serveurs lamp-a et lamp-b
3. Stats HAProxy (optionnelle mais stylée)

```
root@Serveur-Lamp-B:~# ab -n 1000 -c 10 http://10.140.140.15/
This is ApacheBench, Version 2.3 <$Revision: 1913912 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking 10.140.140.15 (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
```

Test charge Temp-B

Temp-A logs

```
10.140.140.15 - - [01/Jul/2025:09:31:17 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:17 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:17 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:17 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:18 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:18 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:18 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:18 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:18 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
10.140.140.15 - - [01/Jul/2025:09:31:18 +0000] "GET / HTTP/1.0" 200 50540 "-" "ApacheBench/2.3"
```

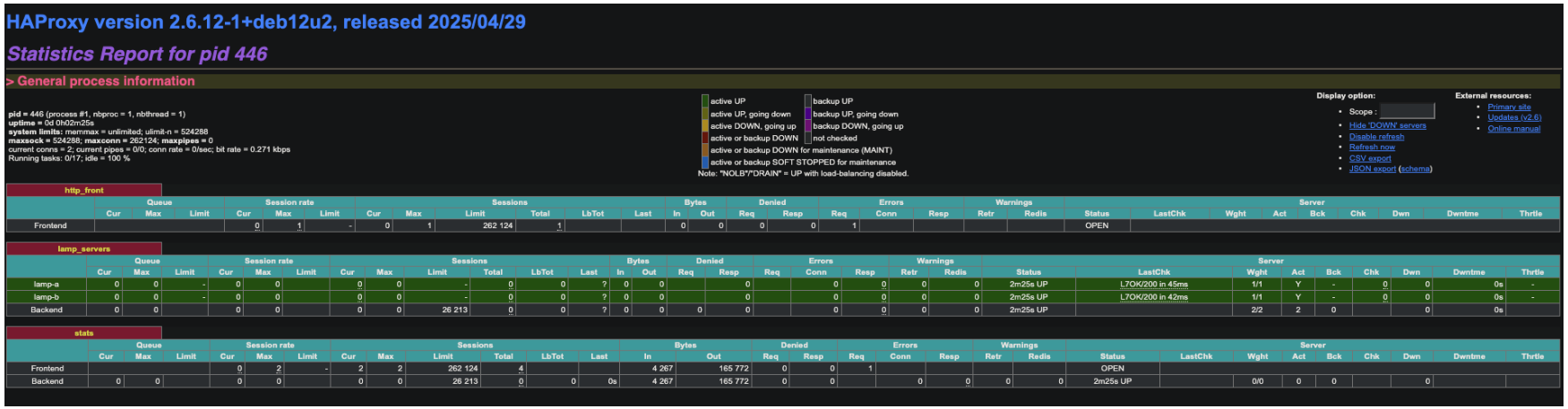
Activer la page de stats HAProxy

```
listen stats
    bind *:8080
    stats enable
    stats uri /haproxy?stats
    stats refresh 10s
    stats auth admin:admin
```

Dans le fichier de config HAProxy (/etc/haproxy/haproxy.cfg) à ajouter à la fin :

On y voit :

- Les serveurs en ligne
- Qui prend le trafic
- Combien de connexions



Résultat attendu

- Cela permet de :
- - Voir si `lamp-a` et `lamp-b` prennent bien chacun une partie de la charge
- - Vérifier que le **failover** (si un tombe, l’autre prend tout) fonctionne
- - Tester des scénarios en coupant `apache2` sur un des nœuds pour voir si HAProxy bascule bien

Conclusion