

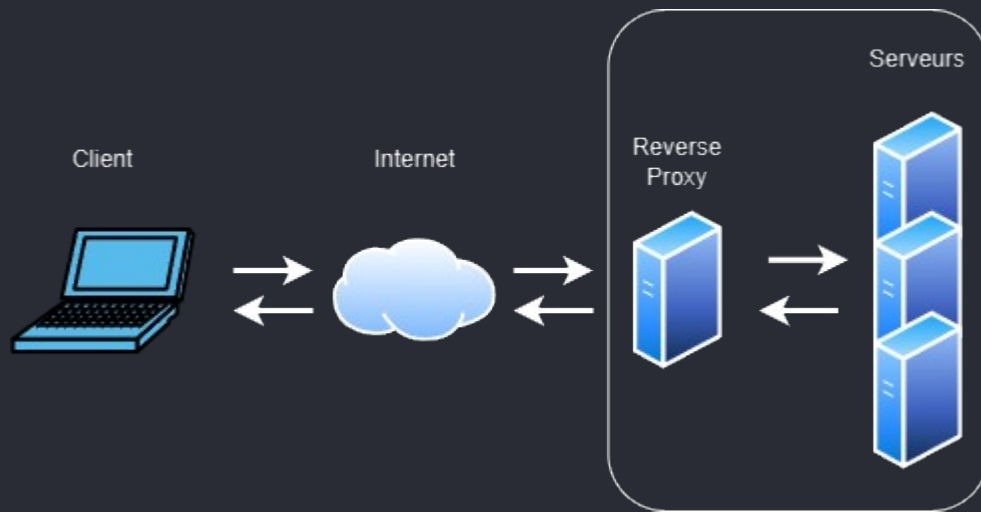
# NGINX



# Qu'est-ce que NGINX ?

C'est une option légère qui peut être utilisée comme serveur web ou proxy inverse.

Schéma d'un proxy inverse:



# Sommaire

**A.** Tutoriel

**B.** Documentation

**C.** Problèmes

**D.** Ressources

rencontrés et solutions

# A. Tutoriel

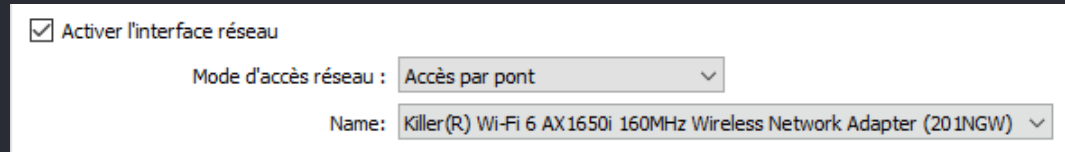
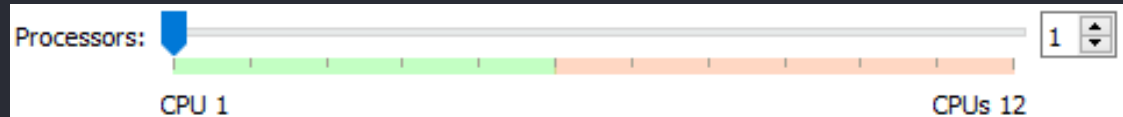
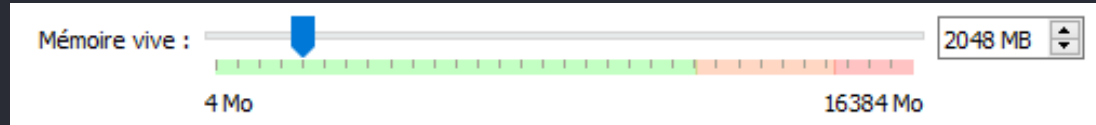
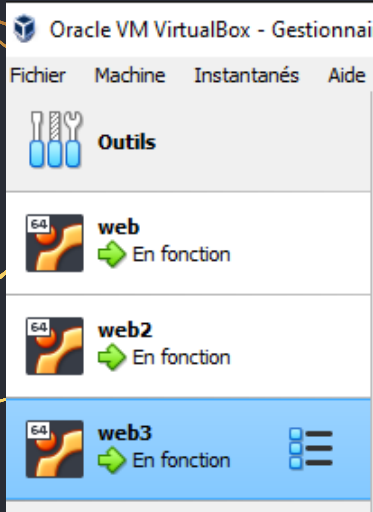
## Étapes:

1. Création des VM's
2. Installation de NGINX sur nos serveurs
3. Réglage des Pare-feux
4. Création du site web
5. Paramétrage de l'équilibrage de charge
6. Ajout du DNS sur le client

**Note:** vous devez changer le nom de domaine et les ip's dans les fichiers de configuration

# 1. Création des VM's

On crée 3 VM avec comme système d'exploitation Ubuntu Server, à chacune d'entre elles, on attribue 1 CPU et 2 Go de RAM, et on les configure toutes en mode réseau accès par pont.



## 2. Installation de NGINX sur nos serveurs

Serveur  
web 1



Serveur  
web 2



Serveur  
web 2



```
sudo apt update
```

```
sudo apt install nginx
```

```
sudo ufw app list
```

```
web1@web1:~$ sudo ufw app list
```

```
[sudo] password for web1:
```

```
Available applications:
```

```
  Nginx Full
```

```
  Nginx HTTP
```

```
  Nginx HTTPS
```

```
  OpenSSH
```

```
web1@web1:~$
```

### 3. Réglage des Pare-feux

On doit régler le pare-feu pour autoriser l'accès au site web seulement sur certains ports.

Il existe trois profils pour Nginx:

**Nginx Full:** Ce profil ouvre à la fois le port 80 (trafic web normal, non crypté) et le port 443 (trafic crypté TLS/SSL).

**Nginx HTTP:** Ce profil n'ouvre que le port 80 (trafic web normal, non crypté).

**Nginx HTTPS:** Ce profil n'ouvre que le port 443 (trafic crypté TLS/SSL).

On va donc autoriser le SSH et le Nginx Full.

Le serveur Nginx démarre automatiquement, il devrait déjà être opérationnel, on vérifie grâce à la commande `systemctl` que le service fonctionne bien.

Serveur  
web 1



Serveur  
web 2



Serveur  
web 2



```
sudo ufw allow "Nginx Full"
```

```
sudo ufw allow "OpenSSH"
```

```
sudo ufw enable
```

```
systemctl status nginx
```



[illegible]

## 4. Création du site web

On modifie sur tous les serveurs la page d'index en ajoutant du code html dans le fichier puis on passera sur notre serveur principal:

```
sudo nano /var/www/html/index.nginx-debian.html
```

```
web1@web1: ~  
GNU nano 6.2 /var/www/html/index.nginx-debian.html  
<html>  
<head></head>  
<body>  
<h1>Bienvenue sur le serveur 1</h1>  
<a href="page1.html">Page 1</a>  
</body>  
</html>
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify

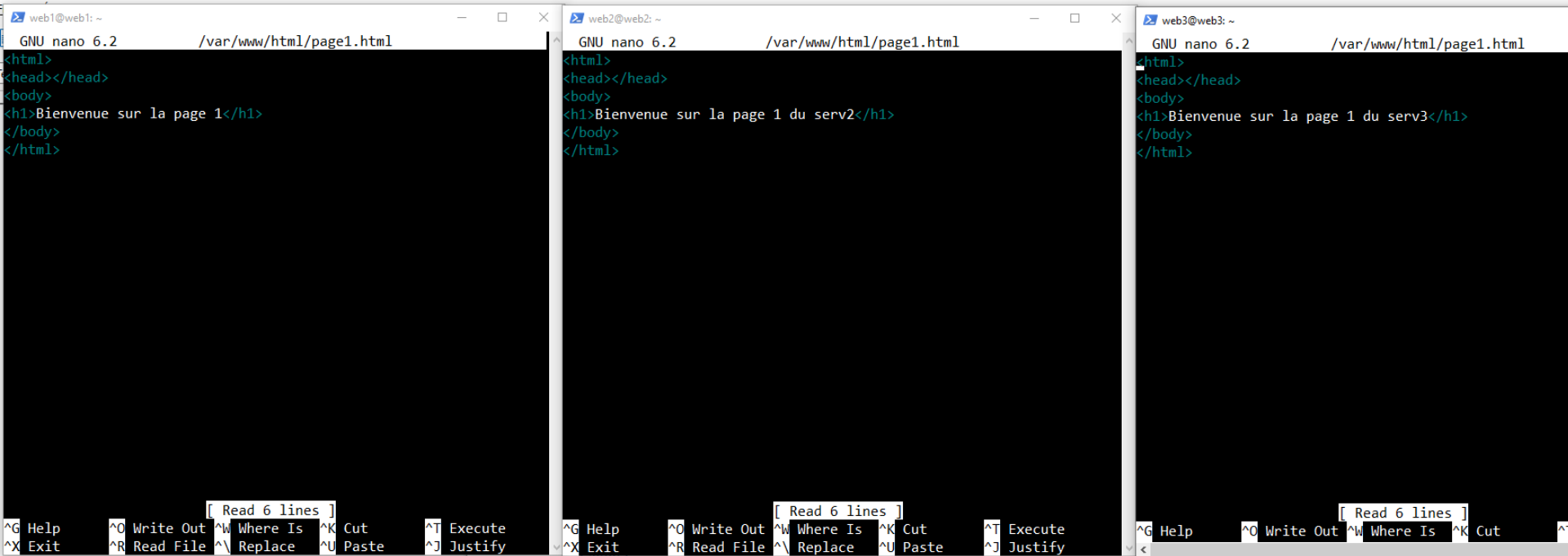
```
web2@web2: ~  
GNU nano 6.2 /var/www/html/index.nginx-debian.html  
<html>  
<head></head>  
<body>  
<h1>Bienvenue sur le serveur 2</h1>  
<a href="page1.html">Page 1</a>  
</body>  
</html>
```

[ Read 7 lines ]  
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify

```
web3@web3: ~  
GNU nano 6.2 /var/www/html/index.nginx-debian.html  
<html>  
<head></head>  
<body>  
<h1>Bienvenue sur le serveur 3</h1>  
<a href="page1.html">Page 1</a>  
</body>  
</html>
```

[ Wrote 7 lines ]  
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify

On ajoute une deuxième page sur tous nos serveurs:  
`sudo nano /var/www/html/page1.html`




The image displays three terminal windows side-by-side, each running the GNU nano 6.2 text editor to edit the file `/var/www/html/page1.html`. The windows are titled `web1@web1: ~`, `web2@web2: ~`, and `web3@web3: ~`. Each window shows the same HTML content:

```
<html>
<head></head>
<body>
<h1>Bienvenue sur la page 1</h1>
</body>
</html>
```

At the bottom of each window, a status bar shows the following options: `^G Help`, `^O Write Out`, `^W Where Is`, `^K Cut`, `^T Execute`, `^X Exit`, `^R Read File`, `^_ Replace`, `^U Paste`, and `^J Justify`. A small box labeled `[ Read 6 lines ]` is visible above the status bar in each window.

On passe sur notre serveur principal et on modifie le fichier de configuration:

`sudo nano /etc/nginx/sites-available/default`



The screenshot shows a terminal window titled "Sélection web1@web1: ~". The prompt indicates the user is editing the file `/etc/nginx/sites-available/default` using GNU nano 6.2. The configuration content is as follows:

```
server {  
    listen 443;  
    listen [::]:443;  
  
    root /var/www/html;  
    index index.nginx-debian.html page1.html;  
  
    location / {  
        try_files $uri $uri/ =404;  
    }  
}
```

On créer un lien symbolique pour activer le site, il est peut-être déjà créé de base, puis on teste la configuration:

```
web1@web1:~$ sudo ln -s /etc/nginx/sites-available/default /etc/nginx/sites-enabled/  
ln: failed to create symbolic link '/etc/nginx/sites-enabled/default': File exists  
web1@web1:~$
```

```
web1@web1:~$ sudo nginx -t  
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok  
nginx: configuration file /etc/nginx/nginx.conf test is successful  
web1@web1:~$
```

On modifie nos fichiers hosts sur le serveur principal pour autoriser toutes les ip a accéder à notre serveur dns:

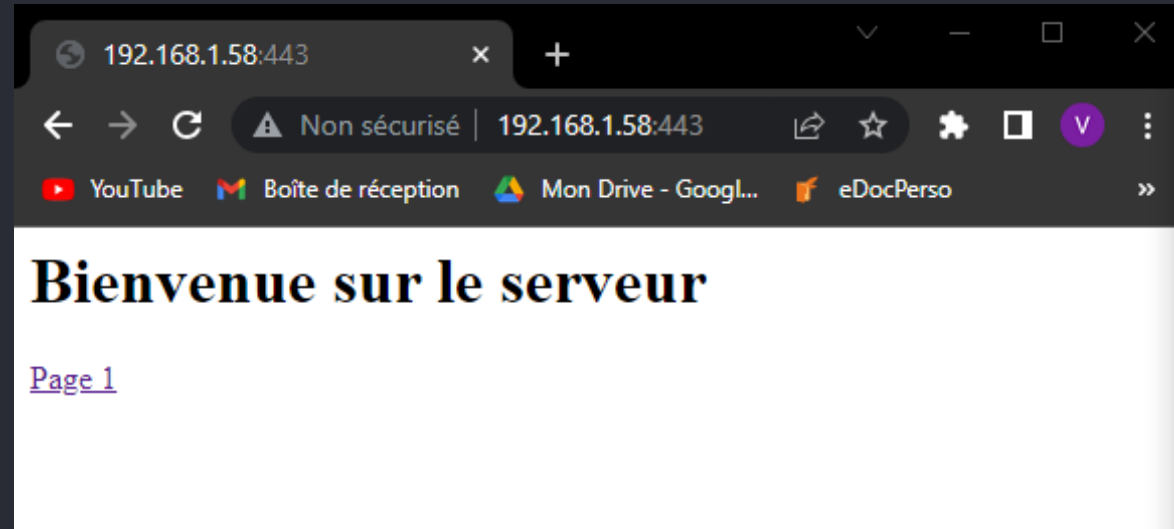
`sudo nano /etc/hosts`

```
web1@web1: ~  
GNU nano 6.2 /etc/hosts  
127.0.0.1 localhost  
127.0.1.1 web1  
0.0.0.0 nginxserv.org www.nginxserv.org
```

On redémarre le service nginx puis on tente de se connecter avec le client via l'ip du serveur principal:

```
sudo systemctl restart nginx
```

Sur le client:





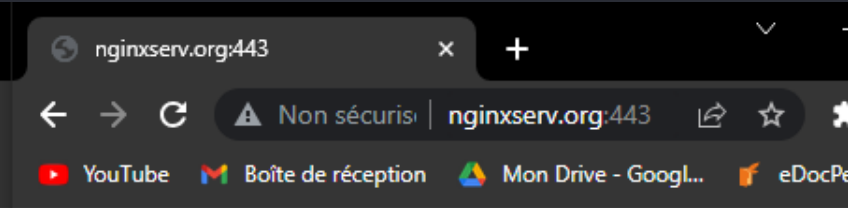
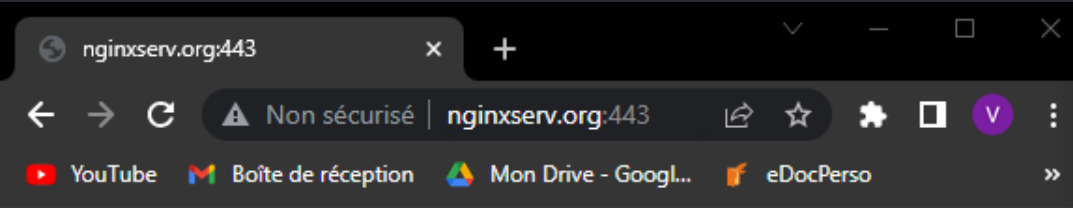
## 5. Paramétrage de l'équilibrage de charge

On crée le fichier  
d'équilibrage de charge sur  
notre serveur principal:

```
sudo nano  
/etc/nginx/conf.d/load-  
balancer.conf
```

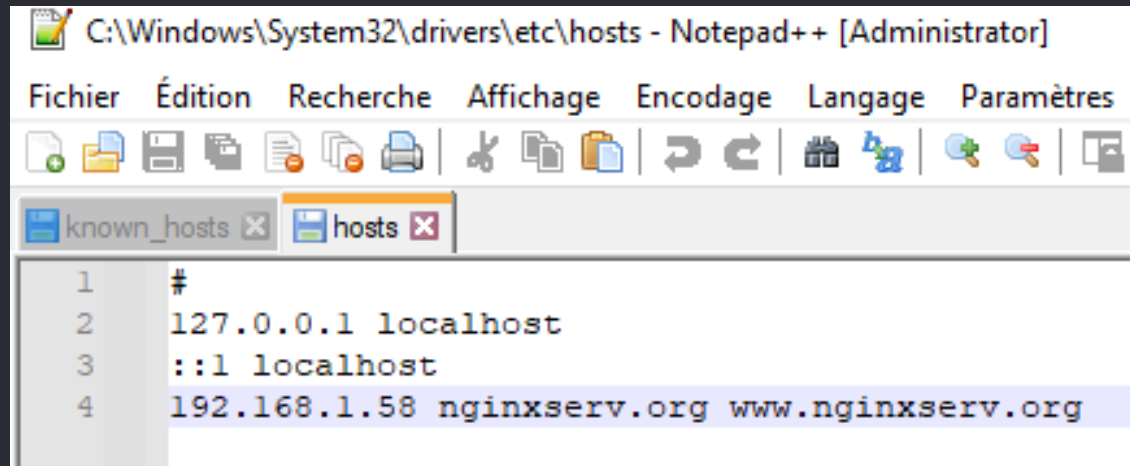
```
web1@web1: ~  
GNU nano 6.2 /etc/nginx/conf.d/load-balancer.conf  
upstream backend {  
    server 192.168.1.57 max_fails=1;  
    server 192.168.1.56 max_fails=1;  
}  
  
server {  
    listen 443;  
    server_name nginxserv.org www.nginxserv.org;  
  
    location / {  
        proxy_pass http://backend;  
    }  
}
```

Pour tester notre load balancing, on redémarre le service nginx puis on se connecte sur le site web via l'ip du serveur et on peut voir que si on rafraichît la page on tombe soit sur notre serveur 2 soit sur notre serveur 3.



## 6. Ajout du DNS sur le client

Pour se connecter via le nom de domaine on doit ajouter le dns sur notre client il suffit d'ouvrir notepad++ en administrateur et d'ouvrir le fichier hosts de notre client dans ce notepad, il se trouve sous C:\Windows\System32\drivers\etc\hosts

A screenshot of the Notepad++ application running as Administrator, editing the hosts file located at C:\Windows\System32\drivers\etc\hosts. The window title is "C:\Windows\System32\drivers\etc\hosts - Notepad++ [Administrator]". The menu bar includes "Fichier", "Édition", "Recherche", "Affichage", "Encodage", "Langage", and "Paramètres". The toolbar contains various icons for file operations and editing. The tab bar shows two tabs: "known\_hosts" and "hosts", with "hosts" being the active tab. The text in the editor is as follows:

```
1 #
2 127.0.0.1 localhost
3 ::1 localhost
4 192.168.1.58 nginxserv.org www.nginxserv.org
```

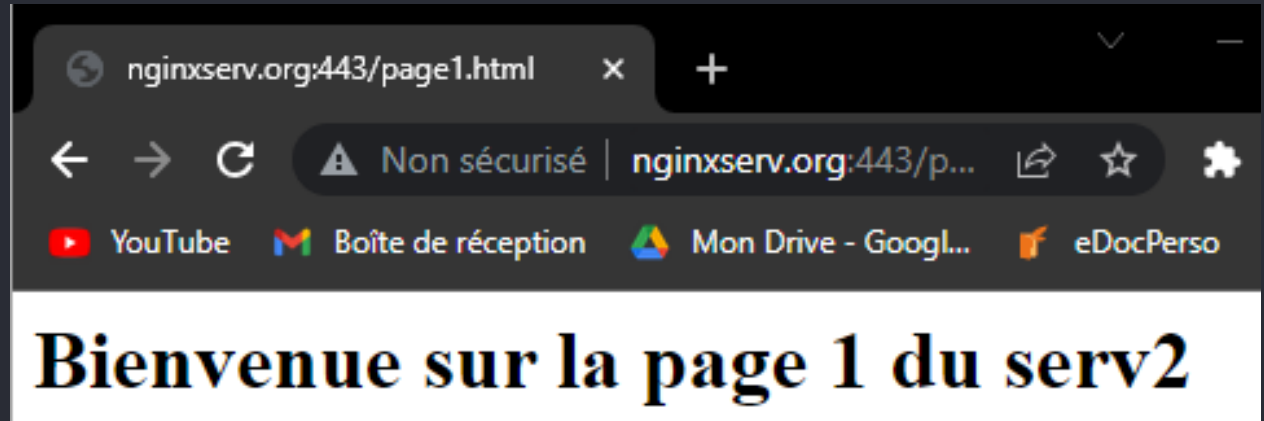
The fourth line is highlighted in blue.

On peut maintenant aller sur

<http://nginxserv.org:443>

et

<http://nginxserv.org:443/page1.html>



## B. Documentation

1. `/etc/nginx/sites-available/default`
2. `/var/www/html/index.html`
3. L'équilibrage de charge / Load Balancing

# 1. /etc/nginx/sites-available/default

C'est notre  
fichier de  
configuration  
pour notre  
site.

```
# on appelle la commande:
server {
# on ouvre les ports 443 en ipv4 et en ipv6:
    listen 443;
    listen [::]:443;
# on sélectionne notre répertoire où sont situés les pages web:
    root /var/www/html;
# on note que dans nos pages il y a la page "index.html":
    index index.html;
# on note le nom de domaine du serveur web
    server_name nginxserv.org www.nginxserv.org;
# si les url ne sont pas bonnes on retourne un code d'erreur 404:
    location / {
        try_files $uri $uri/ =404;
    }
}
```

## 2. /var/www/html/index.html

C'est un fichier qui servira à notre site web pour afficher une page codée en html, cette page web est nommée "index".

Exemple de  
codage  
en html:

```
<!DOCTYPE>
```

```
<html>
```

```
<head>
```

```
<meta charset="utf-8">
```

```
<title> Mon titre </title>
```

```
</head>
```

```
<body>
```

```
</body>
```

```
</html>
```

# 3. L'équilibrage de charge / Load Balancing

Exemple de configuration load balancer avec explications:

```
GNU nano 2.7.4                               Fichier : /etc/nginx/conf.d/load-balancer.conf
# Define which servers to include in the load balancing scheme.
# It's best to use the servers' private IPs for better performance and security.
# You can find the private IPs at your UpCloud Control Panel Network section.

upstream backend {
    server 192.168.42.195;
    server 192.168.42.166;
}

# This server accepts all traffic to port 80 and passes it to the upstream.
# Notice that the upstream name and the proxy_pass need to match.

server {
    listen 80;

    location / {
        proxy_pass http://backend;
    }
}
```



# Choisir une méthode d'équilibrage de charge

Il y a plusieurs méthodes de load balancing:

C'est un code à ajouter dans la commande upstream:

Exemple:

```
upstream backend {  
    least_conn;  
    server1;  
    server2;  
}
```

1 – **Round Robin**: Les requêtes se font sur chaque serveur, chacun leur tour.  
C'est la configuration de base, il n'y a pas de commande à écrire.

2 – **Least Connections**: Une requête est envoyée au serveur avec le moins de connexions actives, encore une fois en tenant compte des poids du serveur.

```
least_conn;
```

3 – **IP Hash**: Cette méthode garantit que les requêtes provenant de la même adresse parviennent au même serveur, sauf si celui-ci n'est pas disponible.

```
ip_hash;
```

# C. Problèmes rencontrés et solutions

Problème 1: L'extension de domaine ne marche pas et ramène sur une recherche google


Solution:

- accéder via l'ip
- changer le .lan en .org sur le fichier hosts et le fichier sites-available

Problème 2: Passer en HTTPS

Solution:

- Il faut posséder le nom de domaine



Problème 3: nginx: [emerg] "upstream" directive is not allowed here in /etc/nginx/sites-enabled/default:20

Solution:

- Ne pas mettre upstream sur le fichier de configuration du serveur mais sur le fichier de configuration du load balancer



Problème 4: Le SSH ne marche plus

Solution:

- sudo ufw allow ssh



Problème 5: `DNS_PROBE_FINISHED_NXDOMAIN`

Solution:

- Ajoute l'ip avec le nom de domaine dans le fichier hosts du client



Problème 6: `La page1 ne s'affiche pas`

Solution:

- Au lieu de:

`nginxserv.org:443/page1`

écrire:

`nginxserv.org:443/page1.html`

## D. Ressources

Ce sont les ressources utilisés pour ma recherche d'informations:

1 - Recherches google / forums

2 - <https://www.youtube.com/watch?v=gZ6uwd2ki4s>

3 -  
<https://www.digitalocean.com/community/tutorials/how-to-install-nginx-on-ubuntu-20-04-fr>

4 -  
<https://www.it-connect.fr/debian-comment-installer-nginx-en-tant-que-serveur-web/>

5 - Google Bard

6 - ChatGPT

7 - <https://mathisthuault.wordpress.com/2018/09/24/load-balancer-avec-nginx/>