

RAPPORT PROJET INFRA

SOMMAIRE

I. Site Web

- 1. La création du site web**
- 2. Apache ou Nginx ?**
- 3. Hébergement du site web**

II. Sauvegarde

- 1. Serveur Cloud ou local ?**
- 2. RSYNC**
- 3. La restauration**
- 4. Automatisation de la sauvegarde**

III. Sécurité

- 1. Pare feu directement sur linux**
- 2. Pourquoi StormShield ?**

IV. Gestion

- 1. Les clients serveur**
- 2. Les interfaces**

V. Réseau

- 1. Le schéma réseau**
- 2. Emuler un réseau avec GNS3**

VI. Budget

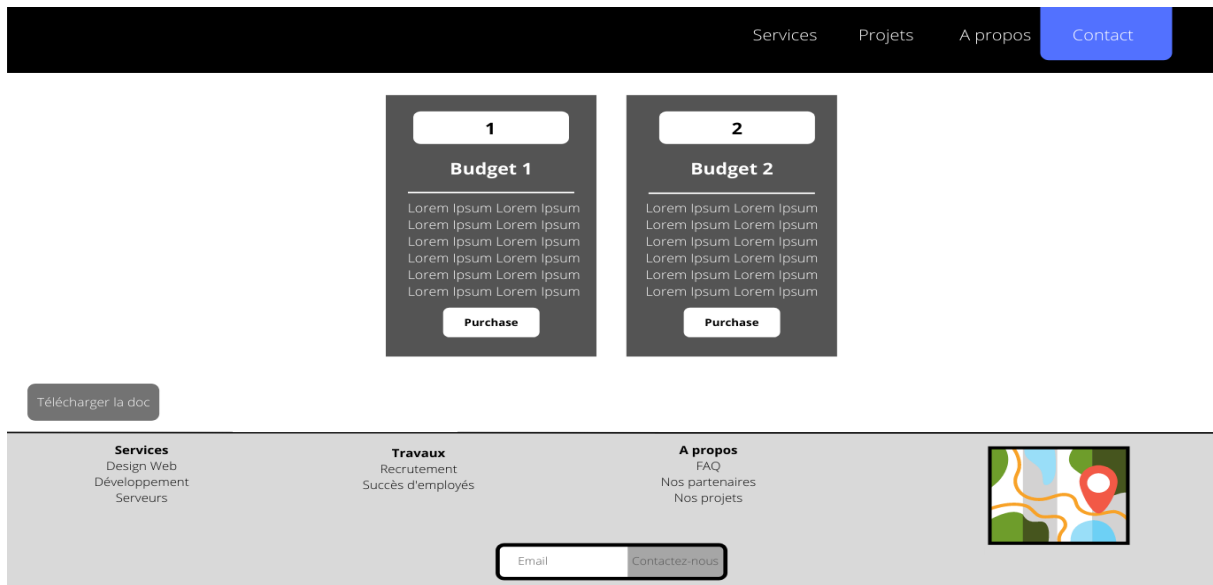
- 1. Présentation des budgets**

I. SITE WEB

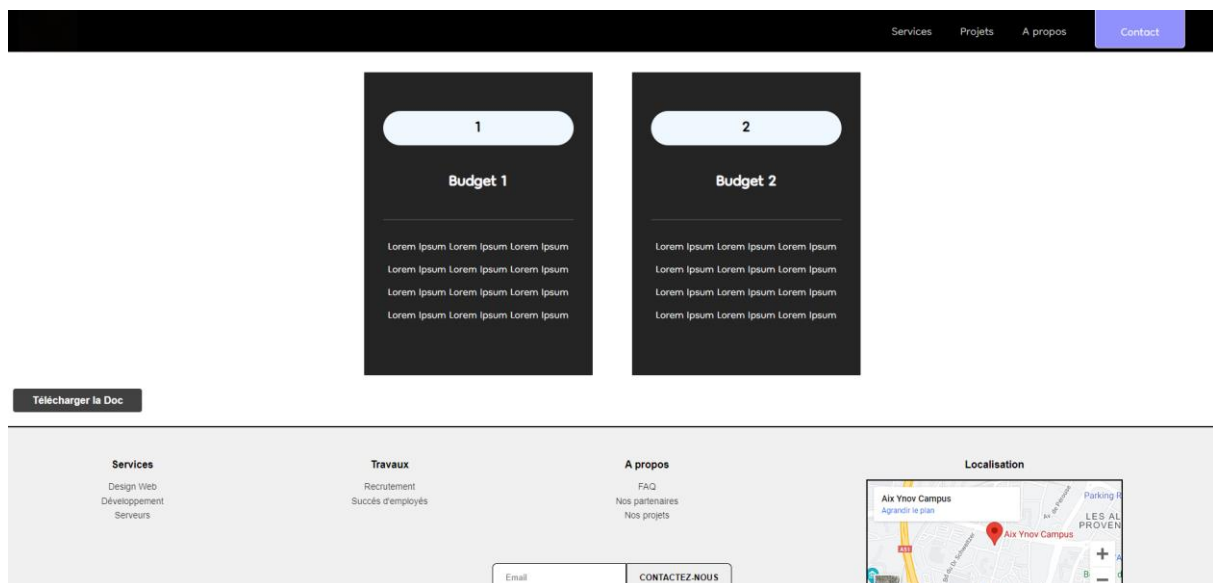
1. La création du site web

Nous avons d'abord fait une maquette sur le site Canva, pour avoir une idée de ce à quoi allait ressembler le site. Nous avons ensuite fait le site en HTML, CSS et JavaScript, en m'appuyant sur le modèle que nous avons créé.

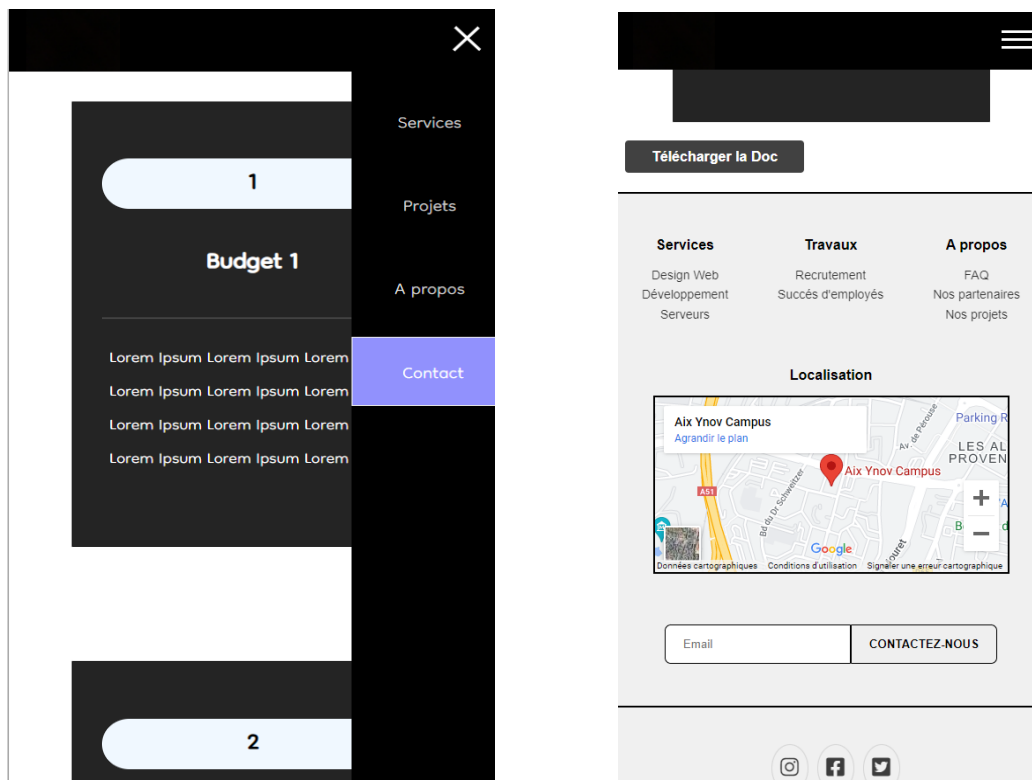
Maquette :



Site :



Nous avons aussi fait la responsive du site :

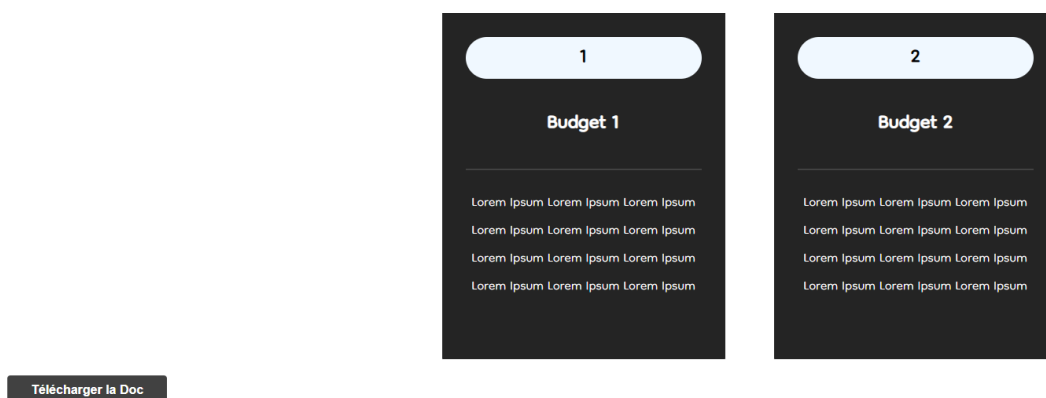


On retrouve donc tout ce qui peut composer un site de base. Nous avons donc :

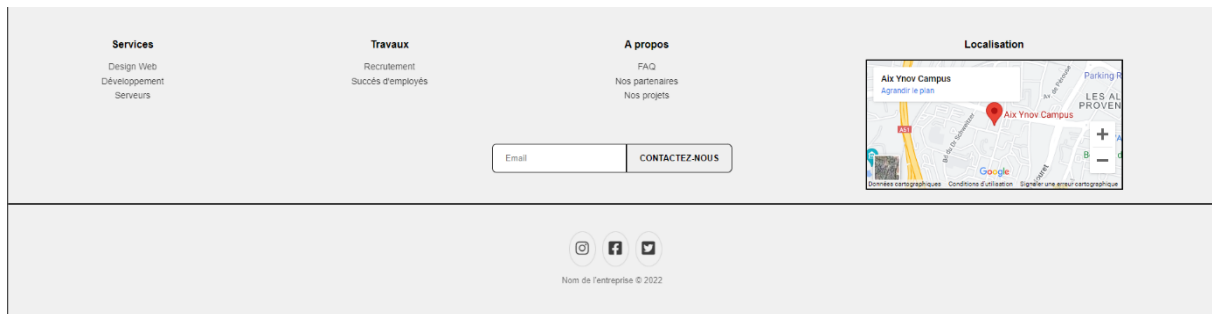
Une Navbar :



Le Content, avec les budgets, et le lien pour télécharger la doc :

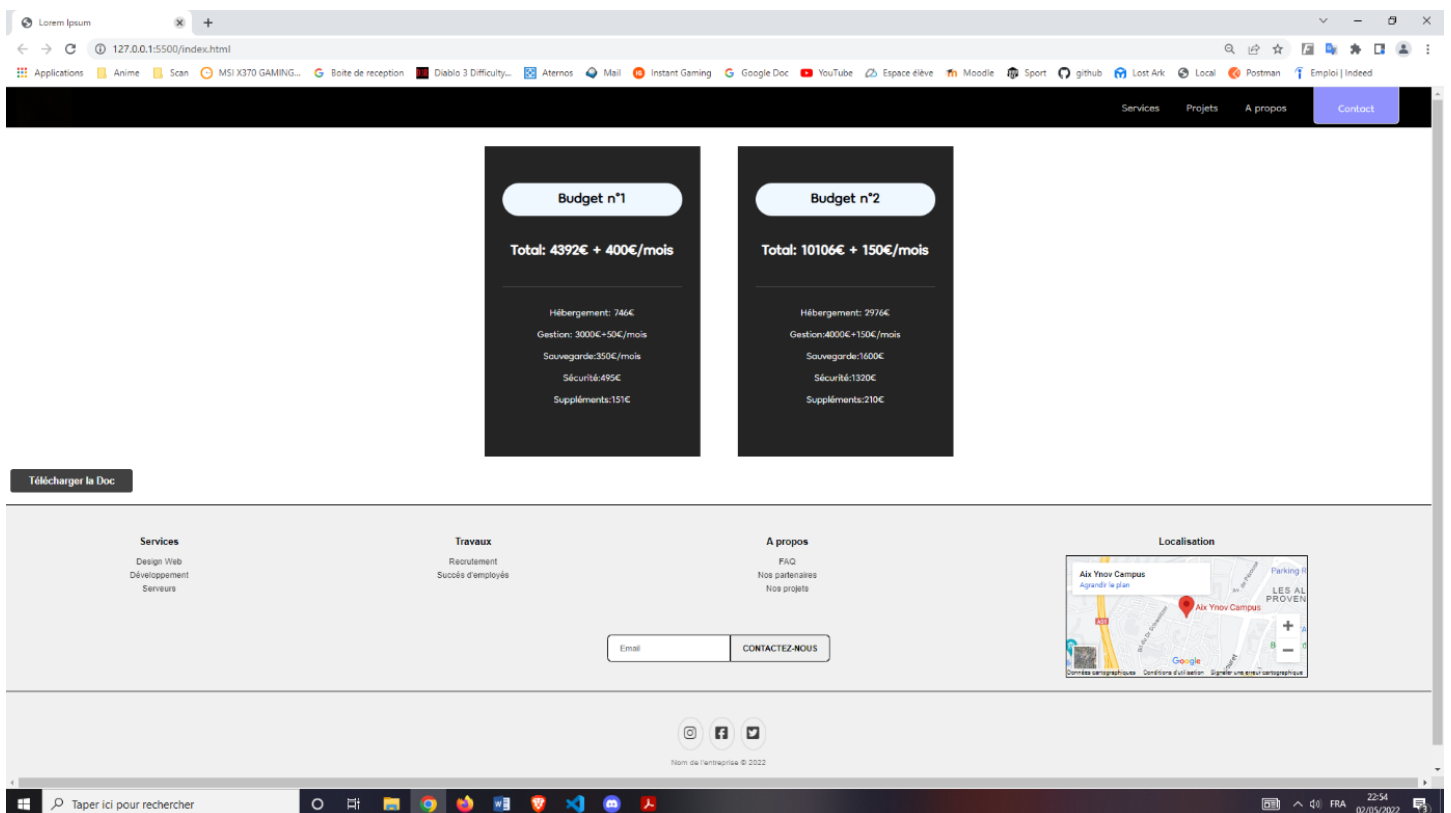


Et un Footer :



Après avoir créé le site, il faut maintenant choisir un hébergeur pour pouvoir le mettre sur le web.

Rendu final :



2. Apache ou Nginx ?



Apache et Nginx :

Apache et Nginx sont des logiciels de serveurs http. Ils desservent des requêtes respectant le protocole de communication client-serveur HTTP (HyperText Transfer Protocol).

Mais Apache et Nginx ont beau tout deux être des logiciels de serveurs http, ils ont beaucoup de différences. Nous allons en parler ici :

Apache VS Nginx :

Au niveau du traitement du contenu statique, Apache utilise la méthode basée sur les fichiers. Nginx en revanche est plus performant qu'Apache. Il est 2,5 fois plus rapide au niveau de la gestion du contenu statique selon un test qui a montré qu'il peut traiter plus de 1000 connexions simultanément.

Mais pour les contenus dynamiques, Apache peut accéder aux contenus dynamiques dans le serveur lui-même sans avoir recours à des composants externes. Nginx en revanche ne peut pas le traiter sur le serveur Web comme le fait Apache. Toutes les requêtes avec un contenu de page Web dynamique sont transmises à un processus externe pour exécution.

Le Support OS ou le Système d'Exploitation est un point important à considérer et de côté-là, Apache surpasse Nginx. Apache s'exécute sur tous les types de systèmes de type Unix, prend totalement en charge [l'hébergement Microsoft Windows](#) et le .NET Framework. Nginx fonctionne également sous tout systèmes UNIX modernes, mais supporte mal Microsoft Windows par rapport aux autres systèmes.

Apache supporte la personnalisation du serveur par le biais de modules dynamiques, ce qui est important car la flexibilité est l'une des caractéristiques les plus importantes lorsqu'on parle de serveur web. Nginx, lui, n'est pas assez flexible pour supporter des modules dynamiques et des changements.

Malgré ces informations en tête nous n'avons pas su choisir au premier abord. Notre idée a donc été de tester les deux outils en mettant en place deux serveurs http.

Après avoir tester et configurer les logiciels, notre choix s'est finalement tourné vers Apache. Nous avons rencontré plus de problèmes lors de la configuration de Apache que lors de la configuration de Nginx et il était clair que le deuxième était le plus rapide à installer mais Apache a été choisis pour sa très grande simplicité d'utilisation, en effet il possède de très nombreux raccourcis contrairement à Nginx et cela rend l'expérience d'utilisation beaucoup plus fluide, agréable et compréhensible.

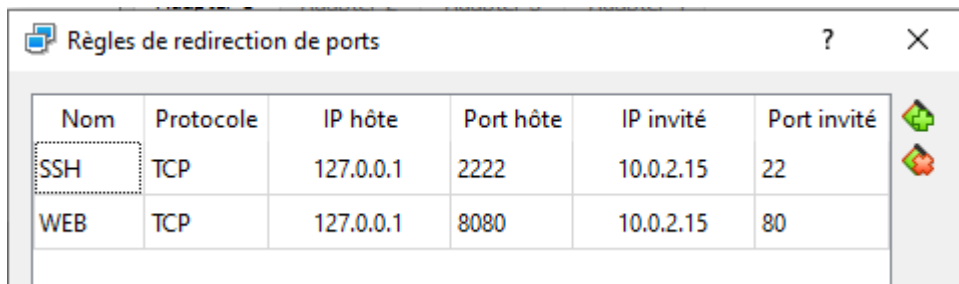
L'exemple le plus flagrant est la commande « a2ensite » qui permet de créer un lien symbolique du dossier « sites-availables » à « sites-enabled », cette manipulation est nécessaire pour Apache et Nginx, et pourtant seul Apache possède un raccourci pour le faire.

3. Hébergement du site web

Pour l'hébergement du serveur nous avons fait le choix de créer une machine virtuelle Linux sur laquelle se trouverais le serveur.

- **Machine virtuelle :**

Notre choix s'est porté sur VirtualBox pour la simplicité et le confort d'utilisation. Le logiciel est disponible gratuitement et sur tous les systèmes d'exploitation. Son principal avantage est son accessibilité, en effet sans même de documentation on en vient à maîtriser les outils qu'il propose très rapidement. Une redirection de port est tout de même nécessaire afin d'accéder au serveur depuis l'ordinateur hôte et de transférer des fichiers.



Nom	Protocole	IP hôte	Port hôte	IP invité	Port invité
SSH	TCP	127.0.0.1	2222	10.0.2.15	22
WEB	TCP	127.0.0.1	8080	10.0.2.15	80

Ainsi, on entre l'adresse IP de l'hôte et le port à utiliser d'un côté, et de l'autre l'adresse IP de la machine et le port qu'on écoute.

- **Système d'exploitation :**

Nous avons d'abord envisagé d'utiliser Linux Ubuntu mais nos recherches nous ont rapidement menés vers Debian. En effet c'est le plus adapté pour les serveurs http dû à sa stabilité. Nous l'avons donc utilisé dans sa version bullseye (11.3), la dernière en date.



- **Serveur HTTP :**

Nous avons choisis d'utiliser Apache pour le serveur car il possède de nombreux raccourcis facilitant son utilisation. Ainsi nous avons pu configurer le serveur en quelques minutes.



Nous avons notre dossier « www », dans le dossier de l'utilisateur contenant les fichiers html

```
dev@debian:~$ ls -l
total 4
drwxr-xr-x 4 dev dev 4096  2 mai  12:58 www
dev@debian:~$
```

Et notre fichier de configuration, pointant dans vers dossier « www » et écoutant sur le port 80

```
GNU nano 5.4                                000-default.conf
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /home/dev/www
```

- **Serveur SSH :**

Afin de relier la machine virtuelle et l'ordinateur qui l'héberge nous avons utilisé un serveur SSH, en l'occurrence Putty. Aucune manipulation compliquée de ce côté-là, tout ce qu'il y a à faire est d'entrer l'IP de l'ordinateur hôte et le port sur lequel on écoute.



Specify the destination you want to connect to

Host Name (or IP address)	Port
127.0.0.1	2222

Connection type:

☒ SSH ☐ Serial ☐ Other: Telnet

On accède ensuite à un terminal, comme si on était à l'intérieur de la machine virtuelle.

```
dev@debian: ~
login as: dev
dev@127.0.0.1's password:
Linux debian 5.10.0-13-686 #1 SMP Debian 5.10.106-1 (2022-03-17) i686

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 May  2 17:06:59 2022 from 10.0.2.2
dev@debian:~$
```

- **Logiciel SFTP :**

Nous avons également utiliser le logiciel Filezilla afin de transférer des fichiers de manière simple et rapide entre l'ordinateur hôte et la machine virtuelle.



Comme sur Putty, tout ce qu'il y a faire est d'entré l'IP de l'ordinateur hébergeur et le port qu'on écoute. En se connectant à l'utilisateur souhaité bien évidemment.

A screenshot of the FileZilla 'Général' (General) tab. The 'Protocole' (Protocol) is set to 'SFTP - SSH File Transfer Protocol'. The 'Hôte' (Host) is '127.0.0.1' and the 'Port' is '2222'. The 'Type d'authentification' (Authentication type) is 'Normale' (Normal). The 'Utilisateur' (Username) is 'dev' and the 'Mot de passe' (Password) is masked with three dots.

On accède ensuite à l'interface graphique des fichiers Windows où l'on peut simplement glisser et déposer les fichiers et dossier que l'on veut transférer.

Site local : C:\Users\bew14\Downloads\INFRA\

INFRA

Favorites

Games

go

IntelGraphicsProfiles

Site distant : /home/dev

/

home

dev

Nom de fichier	Taille de fic...	Type de fichier	Dernière modificat...
..			
captures		Dossier de fichiers	30/04/2022 15:00:36
img		Dossier de fichiers	01/05/2022 15:20:07
index.html	5 519	Brave HTML Docu...	30/04/2022 15:04:44
infra-script.js	72	Fichier source Java...	08/04/2022 09:45:30
infra-style.css	6 424	Fichier source CSS	30/04/2022 15:06:48
3 fichiers et 2 dossiers. Taille totale : 12 015 octets			

Nom de fichier	Taille de fi...	Type de fic...	Dernière modif...	Droits d'ac...	Propriétaire...
..					
.local		Dossier de ...	02/05/2022 17:...	drwxr-xr-x	dev dev
www		Dossier de ...	02/05/2022 12:...	drwxr-xr-x	dev dev
.bash_history	87	Fichier BAS...	02/05/2022 17:...	-rw-r-----	dev dev
.bash_logout	220	Fichier sou...	01/05/2022 20:...	-rw-r--r--	dev dev
.hashrc	3 526	Fichier sou...	01/05/2022 20:...	-rw-r--r--	dev dev
4 fichiers et 2 dossiers. Taille totale : 4 640 octets					

II. SAUVEGARDE

1. Serveur Cloud ou local ?

Nous cherchons maintenant une solution pour la sauvegarde du site.

2 solutions s'offrent à nous : La sauvegarde sur un serveur local ou un serveur cloud.

Nous allons maintenant voir les différents avantages et inconvénients de ces 2 services.

1) Le serveur cloud



Le serveur cloud est un serveur de sauvegarde hébergé dans une entreprise tierce.

Ovh propose beaucoup de services comprenant des serveurs de sauvegardes.

Le produit qui va nous intéresser particulièrement c'est le serveur Scale-1.

Le scale 1 propose toutes les caractéristiques qu'on attend d'un serveur de sauvegarde au niveau de la machine et de l'entretien de ses fonctions.

Le gros avantage d'un serveur cloud est donc qu'on peut avoir un serveur performant complètement automatisé par Ovh selon nos besoins.

Le 2^{ème} avantage c'est que le serveur est louable pour 341 € par mois ce qui permet d'alléger les coûts immédiats d'une entreprise lors de la mise en place d'une infrastructure.

Le grand inconvénient de ce service reste néanmoins son principe même : le cloud.

Dans le cadre d'une infrastructure local ce côté cloud reste un problème puisque on ne sera plus dans une infrastructure 100% local.

On devra donc configurer l'architecture de l'infrastructure en s'adaptant à ce serveur cloud pouvant engendrer finalement plus de travail et de coût que pour un serveur classique.

2) Le serveur local

On va maintenant parler de ce serveur dit plus "classique", le serveur local.

Le serveur de sauvegarde local demande 2 paramètres : un serveur Nas ainsi qu'un logiciel de gestion de sauvegarde qui gère ce dernier.

Au niveau de la machine, nous avons comparé les différents produits et nous avons trouvé des serveurs que nous avons jugés assez intéressants rapport qualité/prix aux alentours des 1500€.

Au niveau du logiciel de gestion de sauvegarde nous avons choisi RSYNC, qui est un logiciel de synchronisation de fichiers très utilisé par les utilisateurs de linux et possède donc une très vaste documentation trouvable facilement sur internet.

2. RSYNC

Avant d'utiliser RSYNC, quelques prérequis sont obligatoires :

- On installe openssh-server pour autoriser la connexion ssh

```
alex@ubuntu:~$ sudo apt list --installed | grep openssh-server
WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

alex@ubuntu:~$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ncurses-term openssh-client openssh-sftp-server ssh-import-id
Suggested packages:
  keychain libpam-ssh monkeysphere ssh-askpass molly-guard
The following NEW packages will be installed:
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
The following packages will be upgraded:
  openssh-client
1 upgraded, 4 newly installed, 0 to remove and 332 not upgraded
```

- On vérifie si tout marche bien

```
alex@ubuntu:~$ sudo service ssh status
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2022-05-01 09:15:21 PDT; 4min 30s ago
     Docs: man:sshd(8)
           man:sshd_config(5)
   Main PID: 4244 (sshd)
    Tasks: 1 (limit: 2252)
   Memory: 1.0M
   CGroup: /system.slice/ssh.service
           └─4244 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups

May 01 09:15:21 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...
May 01 09:15:21 ubuntu sshd[4244]: Server listening on 0.0.0.0 port 22.
May 01 09:15:21 ubuntu sshd[4244]: Server listening on :: port 22.
May 01 09:15:21 ubuntu systemd[1]: Started OpenBSD Secure Shell server.
alex@ubuntu:~$
```

- On lance ssh-serveur

```
alex@ubuntu:~$ sudo service ssh start
alex@ubuntu:~$
```

- Certaines machines linux ont aussi besoin de plus de permissions avec des commandes comme par exemple:

```
alex@ubuntu:~$ rsync -av --chmod=Du+rwX SRC DST
sending incremental file list
```

Il est aussi important de noter la commande : **sudo apt-get install rsync** qui permet d'installer RSYNC même si ce dernier est maintenant installé de base sur les ordinateurs linux.

On peut désormais voir les commandes les plus utiles à utiliser avec RSYNC.

1) La copie de dossier :

```
rsync -e ssh -avz --delete-after /home/source user@ip_du_serveur:/dossier/destination/
```

Cette commande permet tout simplement de copier le contenu d'un dossier dans un autre en supprimant le contenu actuel du dossier.

```
alex@ubuntu:~$ rsync -e ssh -avz --delete-after /home/Old alex@192.168.182.133:/New/
alex@192.168.182.133's password:
building file list ...
rsync: link_stat "/home/Old" failed: No such file or directory (2)
done
```

En exécutant cette commande, nous avons rencontré un problème mais bizarrement la commande s'est quand même réalisée ce qui a donc dupliqué le fichier « Old » dans le fichier « New »

```
alex@ubuntu:~$ cd Old
alex@ubuntu:~/Old$ cat Test
orem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et d
, laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate v
datat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
alex@ubuntu:~/Old$ cd
alex@ubuntu:~$ cd New
alex@ubuntu:~/New$ cat Test
orem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et d
, laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate v
datat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
alex@ubuntu:~/New$
```

2) La synchronisation de serveur :

```
sudo -u www-data rsync -a --progress -e ssh --rsync-path "sudo -u www-data rsync" utilisateur@serveur_distant:/var/www/ /var/www/
```

Cette commande demande une autre machine sur le même réseau pour s'exécuter.

Nous avons donc essayé de créer une deuxième machine virtuelle sur le même pc.

On synchroniserait donc les données d'une machine sur l'autre afin de tester notre commande.

Malgré les tentatives, nous n'avons pas réussi à lier les 2 machines puisque la machine 1 ne reconnaissait pas le « hostname » de la machine 2

```
alex@ubuntu:~$ sudo -u www-data rsync -a --progress -e ssh --rsync-path "sudo -u www-data rsync" alex@alexandre:/var/www/ /var/www/
ssh: Could not resolve hostname alexandre: Temporary failure in name resolution
rsync: connection unexpectedly closed (0 bytes received so far) [Receiver]
rsync error: unexplained error (code 255) at io.c(235) [Receiver=3.1.3]
alex@ubuntu:~$
```

3) L'actualisation :

```
rsync -rtvu source_rep/ destination_rep/
```

Cette commande va faire la même chose que la 1^{ère} commande mais sur des fichiers déjà existant.

Le vrai avantage de cette 3^{ème} commande c'est qu'elle est plus rapide que la 1^{ère} commande à condition que la copie de dossier ait été effectuée au moins une fois.

```
alex@ubuntu:~/New$ cat Test
JE RAJOUTE UNE INFO |Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
d exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute ir
eur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit
alex@ubuntu:~/New$ cd
alex@ubuntu:~$ cd Old
alex@ubuntu:~/Old$ cat Test
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incip
o laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in repreh
idatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
alex@ubuntu:~/Old$ cd
alex@ubuntu:~$ rsync -rtvu New/ Old/
sending incremental file list
Test

sent 575 bytes  received 35 bytes  1,220.00 bytes/sec
total size is 467  speedup is 0.77
alex@ubuntu:~$ cd Old
alex@ubuntu:~/Old$ cat Test
JE RAJOUTE UNE INFO |Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
d exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute ir
eur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit
alex@ubuntu:~/Old$
```

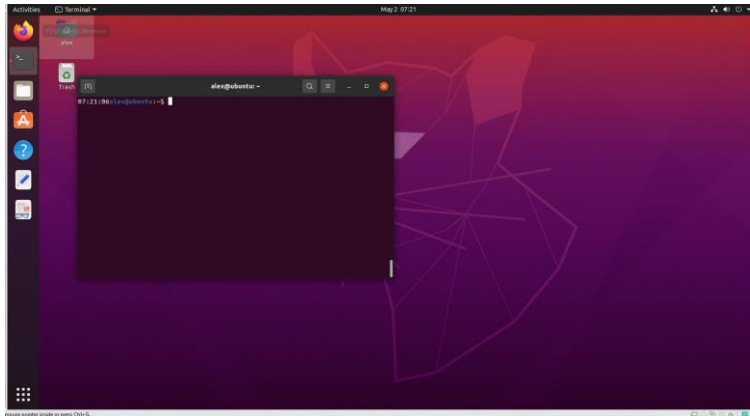
Maintenant que l'on peut copier, synchroniser et actualiser un fichier, il faut savoir comment automatiser toutes ses tâches afin d'avoir un serveur quasi autonome.

3. La restauration

En cas de problèmes amenant à une perte de fichier sources du site, il faut effectuer une restauration de ces ressources que nous allons expliquer étape par étape.

1^{ère} Etape :

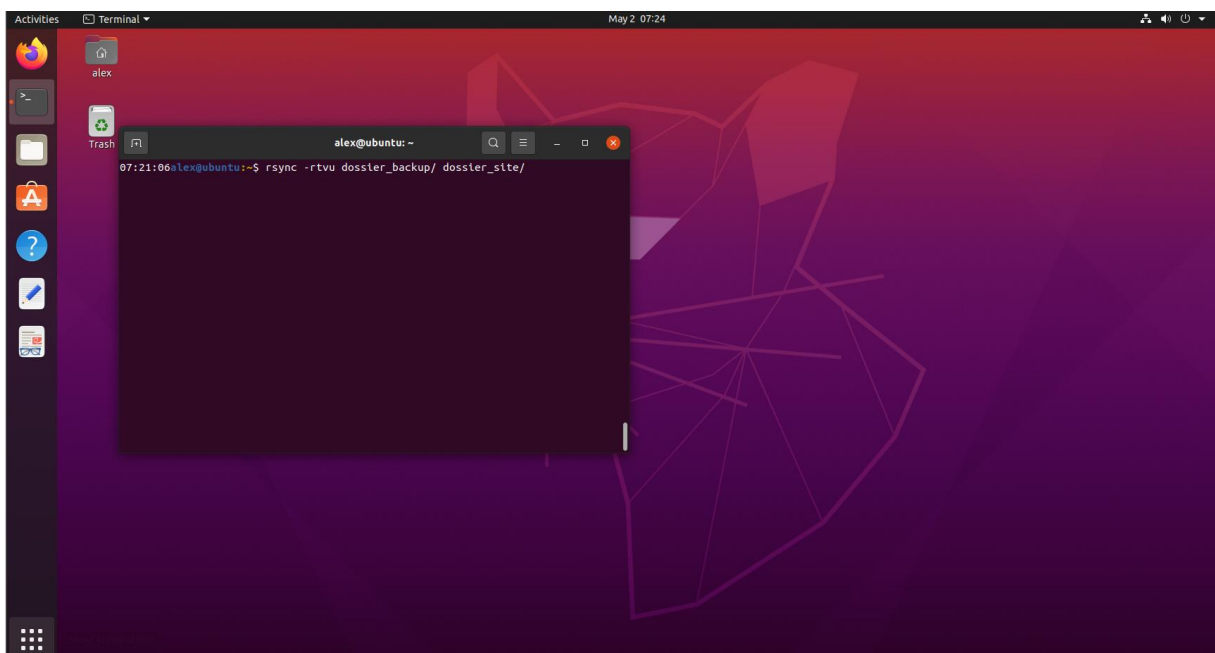
Ouvrir le terminal linux du serveur de sauvegarde



2^{ème} Etape :

Ecrire dans le terminal la commande : `rsync -rtvu source_rep/ destination_rep/` en remplaçant source_rep par le nom du dossier qui contient la sauvegarde miroir et destination_rep par le nom du dossier à restaurer.

Exemple :



Une fois la commande exécutée le logiciel va restaurer le fichier source du site.

Or cette méthode peut ne pas marcher pour 2 raisons.

La 1^{ère} raison, c'est si le fichier backup n'est pas une copie du fichier source de base. Alors il faudra utiliser des commandes plus complexes comme par exemple :

```
--backup-dir=`date +%y%m` suffix=.`date +%y%m%d-%H%M`
```

Il faudra juste rentrer les bons paramètres selon la date à laquelle on veut restaurer le fichier pour avoir la bonne version.

La 2^{ème} raison c'est qu'il n'y a pas de fichier backup. Dans ce cas-là il faut refaire complètement le site web et l'architecture du serveur selon les cas.

Pour refaire le serveur web nous avons fait un rapport détaillé de comment s'y prendre trouvable dans le dossier « PROJET INFRA » sous le nom de « RAPPORT DETAILLE SERVEUR WEB »



4. Automatisation de la sauvegarde



Pour automatiser la sauvegarde, il nous faut un logiciel de planificateur de tâche.

Nous avons fait le choix d'utiliser crontab pour 2 raisons.

La 1^{ère} raison c'est qu'il correspondait à nos attentes et que par sa popularité il possède une très bonne documentation.

La seconde raison c'est que nous nous sommes déjà familiarisés avec cron lors des cours sur linux.

Pour rentrer une tâche dans crontab il faut taper dans le terminal : **crontab -e** ce qui nous emmène dans l'interface crontab.

```
root@ubuntu: /home/alex x alex@ubuntu: ~/TP2
GNU nano 4.8 /tmp/crontab.XvLMtH/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow command
```

Dans notre exemple de planification nous avons décidé de planifier toutes les tâches rsync tous les jours à intervalles de 4h ce qui nous donne en crontab :

0 4 * * * + tâche

On va aussi prendre comme exemple un script RSYNC que je copie dans un fichier .sh ce qui nous donne :

```
#
# m h dom mon dow command
0 4 * * * home/Tp2/script.sh
```

Nous avons donc un script de sauvegarde qui se lancera toutes les 4 heures.

III. SECURITE

1. Pare feu directement sur linux

Au niveau de la sécurité il faut protéger les données circulant dans l'entreprise. Linux propose déjà de base l'outil iptable, un outil permettant de configurer Netfilter. Netfilter est un pare-feu implémenter dans la surcouche de linux.

On va maintenant configurer Netfilter grâce à quelque commandes iptable :

On commencer par taper la commande = « iptables -L --line-numbers »

```
root@ubuntu:/home/alex# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target      prot opt source                destination            tcp dpt:http

Chain FORWARD (policy ACCEPT)
num  target      prot opt source                destination

Chain OUTPUT (policy ACCEPT)
num  target      prot opt source                destination
root@ubuntu:/home/alex#
```

On va désactiver l'élément 1 « tcp dpt : http » pour interdire le Traffic web externe afin de vraiment rendre le réseau local avec la commande = « iptables -D OUTPUT 2 »

```
root@ubuntu:/home/alex# iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target      prot opt source                destination

Chain FORWARD (policy ACCEPT)
num  target      prot opt source                destination

Chain OUTPUT (policy ACCEPT)
num  target      prot opt source                destination
root@ubuntu:/home/alex#
```

On se sert de la commande « service iptables-persistent » pour que les modifications soient conservées. On peut aussi instaurer des scripts grâce à Systemd compatible avec iptable. (Exemple de script fourni par « doc.ubuntu-fr.org »)

```
[Unit]
Description=Firewall
Requires=network-online.target
After=network-online.target

[Service]
User=root
Type=oneshot
RemainAfterExit=yes
ExecStart=/etc/init.d/nom_de_votre_script_iptables(modifier_le_nom!) start
ExecStop=/etc/init.d/nom_de_votre_script_iptables(modifier_le_nom!) stop

[Install]
WantedBy=multi-user.target
```

Toutes ces configurations peuvent suffirent pour une utilisation lambda mais pas pour une entreprise. Il faut donc compléter avec un pare feu professionnel.

2. Pourquoi StormShield ?



La question désormais est de savoir quelle pare feu choisir pour une entreprise.

Parmi tous les marques de pare feus sur le marché nous avons décidé d'en garder une : StormShield.

Il y a plusieurs raisons pour laquelle nous avons décidé de choisir StormShield.

La première raison c'est son accompagnement chez l'entreprise pour trouver le pare feu idéal.

Dans l'exemple du petite entreprise le prix d'un pare-feu serait entre 600 et 1600 € qui peuvent grimper jusqu'à plus de 4000 € pour une entreprise moyenne.

On retrouve chez StormShield le SN 160 et le SN 310 qui se situent tous deux dans cette gamme de prix.

Ses deux boitiers de pare-feu proposent des connexions vpn sécurisées, plusieurs port Ethernet pour la connexion au réseau local et une protection très solide protègent de la plupart des cybers attaques courantes comme les attaques DoS ou les injections SQL.

Donc en combinant la configuration de Netfilter et un pare-feu StormShield, on obtient une sécurité optimale pour une petite entreprise.



IV. GESTION

1. Les clients serveurs

L'architecture client-serveur désigne un mode de communication entre plusieurs ordinateurs d'un réseau.

Le rôle des clients est d'envoyer des requêtes au serveur et d'attendre leurs réponses, on dit de lui qu'il est actif.

Le rôle du serveur quant à lui est d'attendre les requêtes du client en étant à l'écoute et d'y répondre, on dit de lui qu'il est passif.

Dans notre situation, le client peut être représenté de deux manières différentes :

- Un utilisateur voulant accéder au site depuis chez lui
- Un salarié voulant accéder à l'interface graphique depuis le réseau local de l'entreprise

L'utilisateur du site, fait une requête en entrant un nom de domaine ou une url dans son navigateur, tandis que le salarié fait sa requête en communiquant directement avec le serveur et en lui donnant des modifications à s'apporter.

Le serveur de son côté se met en position d'écoute avant de recevoir n'importe quelle requête, il la traite ensuite lui-même et envoie sa réponse à l'auteur de la demande. Dans le cadre du site il renvoie l'affichage du contenu html, tandis que dans le cas de l'interface graphique, il déploie le contenu visuel et s'apprête à modifier le contenu auquel accède le salarié.



2. Les interfaces

Faire une interface graphique :

Une interface graphique est utile lors de la création d'une infrastructure. Elle permet à une entreprise ou une personne d'interagir plus simplement avec le serveur.

L'interface graphique est l'équivalent graphique de l'interface de ligne de commande réseau.

L'interface graphique d'administration réseau vous permet de visualiser et de contrôler l'état de votre réseau sur le bureau, mais aussi d'interagir avec les profils réseau pour gérer la configuration Ethernet et sans fil.

On peut appeler cela aussi une « Page Admin ». Elle est donc très importante dans le processus de création d'un site Web, car elle permet d'avoir le contrôle sur ce qu'il se passe sur notre site.

Une interface graphique peut être codée en beaucoup de langages. En Python, en Java, en C++... Une entreprise va généralement payer un développeur pour leur page Admin.

Une **interface** intègre vos images et les mises en page.

Le développeur va faire des modifications en fonction de vos retours pour arriver au résultat souhaité.

Pour **un site**, pour lequel il faudra probablement créer un template original, les coûts peuvent varier de 3 000 € à 15 000 €.

On va maintenant voir combien d'interface graphiques nous allons avoir besoin pour la mise en place de l'infrastructure.

Web :

Etant donné que le site web n'est qu'un site vitrine, alors il n'y a aucun besoin de le contrôler

Sauvegarde :

Dans le cas de la sauvegarde par cloud, pas besoin d'interface car le cloud est géré par Ovh.

Dans le cas de RSYNC, on se servira de GRSYNC. GRSYNC est une interface gratuite et facile d'utilisation créée pour gérer RSYNC.

Donc dans tous les cas il n'y aura pas de coût pour les interfaces de sauvegardes.

Sécurité :

On peut utiliser Firewall Builder qui est gratuit et compatible avec Netfilter.

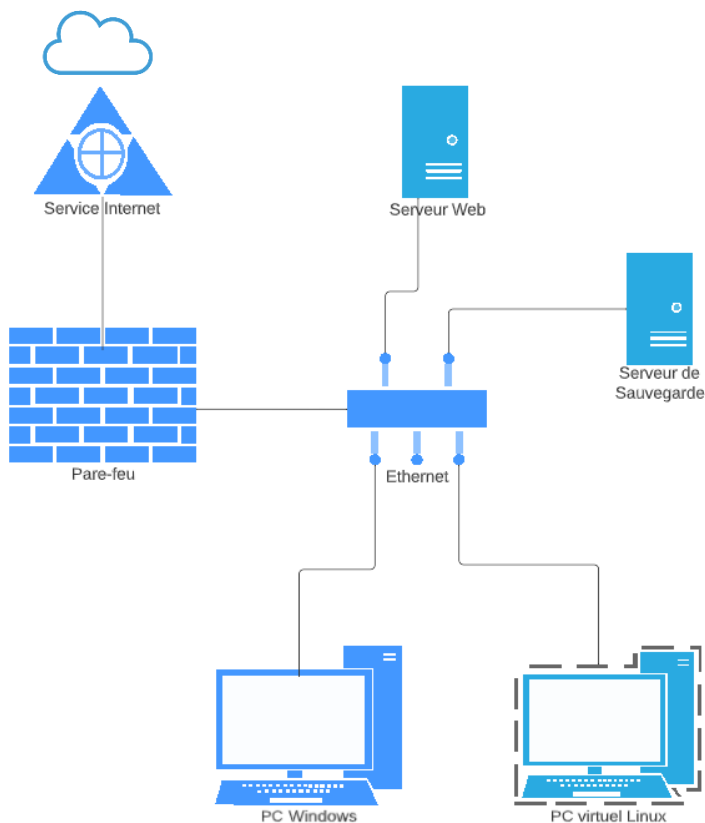
Réseau :

Pour l'administration réseau il faut une interface graphique adaptée à l'entreprise

Il y aura donc une interface graphique à créer si l'on se sert d'interfaces existantes.

V. RESEAU

1. Le schéma réseau



Voici le schéma de l'infrastructure que nous avons réalisé.

Il comporte tous ce qui était demandé dans la consigne : un serveur web, un serveur de sauvegarde 2 clients serveurs (Linux et Windows) ainsi qu'un pare-feu.

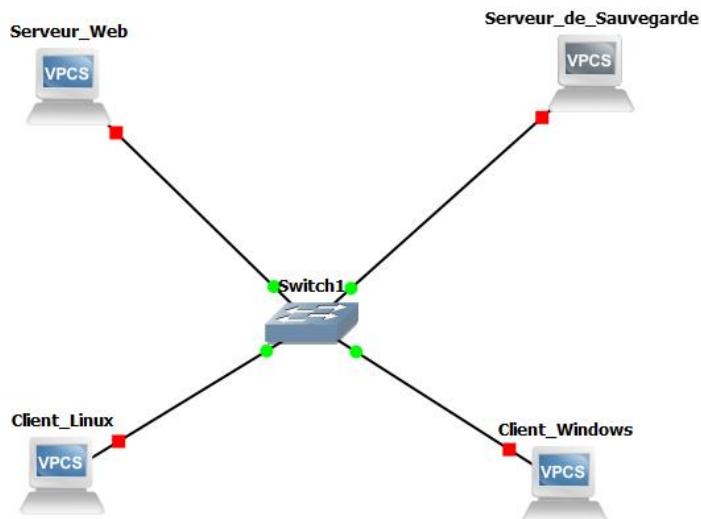
L'interconnexion s'effectue avec un switch Ethernet. On utilise 5 câbles RJ 45.

Dans le cas d'un serveur de sauvegarde cloud on remplace le serveur par le nuage sur le réseau et on le place du côté du service internet.

On va maintenant émuler ce réseau sur le logiciel GNS3.

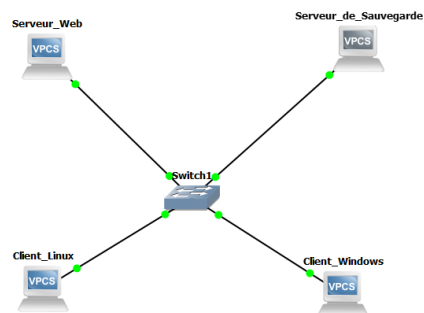
2. Emuler un réseau avec GNS3

Avec les outils de bases de GNS3 on peut se rapprocher du réseau ci-dessus :



La première étape est de connecter tous les serveurs entre eux à l'aide du switch.

On commence donc par lancer le processus pour allumer les 4 serveurs.



Une fois allumer on va vérifier que les machines soient bien sur le même réseau.

Pour cela on va essayer de Ping le serveur web du client linux grâce à son adresse ip :

```
Client_Linux> ping 192.168.1.20
84 bytes from 192.168.1.20 icmp_seq=1 ttl=64 time=0.386 ms
84 bytes from 192.168.1.20 icmp_seq=2 ttl=64 time=0.527 ms
84 bytes from 192.168.1.20 icmp_seq=3 ttl=64 time=0.429 ms
84 bytes from 192.168.1.20 icmp_seq=4 ttl=64 time=0.536 ms
84 bytes from 192.168.1.20 icmp_seq=5 ttl=64 time=0.411 ms

Client_Linux> █
```

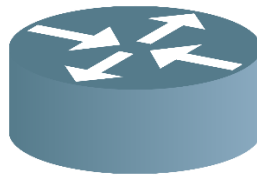
On a donc bien le client linux qui communique avec le serveur web.

```
Serveur_Web> show ip

NAME          : Serveur_Web[1]
IP/MASK       : 192.168.1.20/24
```

(On peut donc observer que le client linux ping bien le serveur web)

Dans le schéma actuel il n'y a pas de routeur mais dans le cas où un routeur serait présent dans l'infrastructure, il faudrait effectuer un routage statique.



On utiliserait ip route comme vu en cours de réseau cette année avec de préférence un routeur Cisco.

Maintenant que nous avons vu toutes les démarches techniques pour la mise en place d'une infrastructure, nous allons parler du côté économique.

Les budgets que nous avons établis se basent sur tous les objets nécessaires à la mise en place de l'infrastructure.

VI. BUDGET

1. Présentation des budgets

(Les prix sont présentés hors taxes)

Budget n°1 :

- Web : Fujitsu PRIMERGY TX1310 M3 **746€**
- Interface graphique **3000€**
- Infogérance de serveur (heures ouvrées) : **50 € /mois**
- Sécurité : Stormshield SN160 **495€**
- Sauvegarde : OVH cloud **350€/mois**
- Switch : TP-LINK TL-SG1024DE **119€**
- 8 câbles RJ45 de 1m **32€**

Total n°1 : **4392€ + 400/mois € HT**

Budget n°2 :

- Web : Lenovo ThinkSystem ST250 **2976€**
- Interface graphique **4000€**
- Infogérance de serveur 24h/24 : **150 € /mois**
- Sécurité : Stormshield SN310 **1320€**
- Sauvegarde : My Cloud Expert Series **1600€**
- Switch : D-Link DGS-1100-24V2 **154€**
- 8 câbles RJ45 de 1m50 **56€**

Total n°2 : **10106€ + 150/mois € HT**