# Mise en place d'un serveur et d'un réseau

AVELINE Baptiste, OUDAR Nicolas

---

## Objectif

> L'objectif est de déployer un serveur Proxmox afin de créer des machines virtuelles (VM) qui nous permettront d'installer divers services tels qu'un agent DVR, un routeur virtuel, un VPN et un réseau avec des VLAN. Ce serveur sera essentiel pour la réalisation de nos travaux pratiques (TP). De plus, grâce au VPN, nous pourrons y accéder de n'importe où.

## Matériels

- **Serveur HP Proliant DL360p g8**
  - Processeur : Intel Xeon® CPU E5-2640 v2 (2 GHz, 8 cœurs, 16 threads)
  - Mémoire : 48 Go de RAM DDR3
  - Stockage :
    - 4 HDD SAS de 900 Go (en RAID 5 pour une combinaison de rapidité de traitement et de tolérance aux pannes)
    - 1 SSD de 400 Go.
- **Switch Aruba J9776A**
  - Utilisé pour connecter d'autres serveurs ainsi que les ports de gestion (IDRAC, iLO, UPS, ...)
- **Switch Aruba JL255A**
  - Switch de la baie S1A du professeur
  - Lien fibre de la box
  - Utilisé pour connecter les caméras en POE
- **Caméras ICA-3280 (x2)**
  - Deux caméras pour surveiller les salles S1 et S2
- **Cisco AIR-AP1141N-E**
  - Équipement sans fil Cisco utilisé pour fournir une connectivité réseau sans fil.

# Étapes

1. **Préparation du serveur :**
   - Mise à jour du firmware du serveur.
   - Configuration du RAID.
2. **Installation et Configuration de Proxmox :**
   - Installation du système d'exploitation sur le serveur.
   - Configuration du serveur Proxmox.
3. **Installation d’un routeur virtuel Mikrotik :**
   - Déployer une machine virtuelle Mikrotik pour assurer les fonctions de routage virtuel.
4. **Configuration du réseau avec des VLAN :**
   - Implémenter des VLAN sur les switch et routeur pour une segmentation efficace du réseau.
5. **Installation d’un agent DVR :**
   - Installer l'agent DVR sur une machine virtuelle.
6. **Configuration de l’agent DVR :**
   - Configurer l'agent DVR pour répondre aux besoins de surveillance.
7. **Installation d’un serveur VPN :**
   - Mettre en place une machine virtuelle dédiée pour héberger le service VPN.
8. **Configuration du serveur VPN :**
   - Configurer le VPN pour permettre un accès sécurisé depuis n'importe où.
9. **Configuration de la connexion sans fil :**
   - Configurer le point d'accès sans fil pour permettre une connexion sans fil sécurisée.

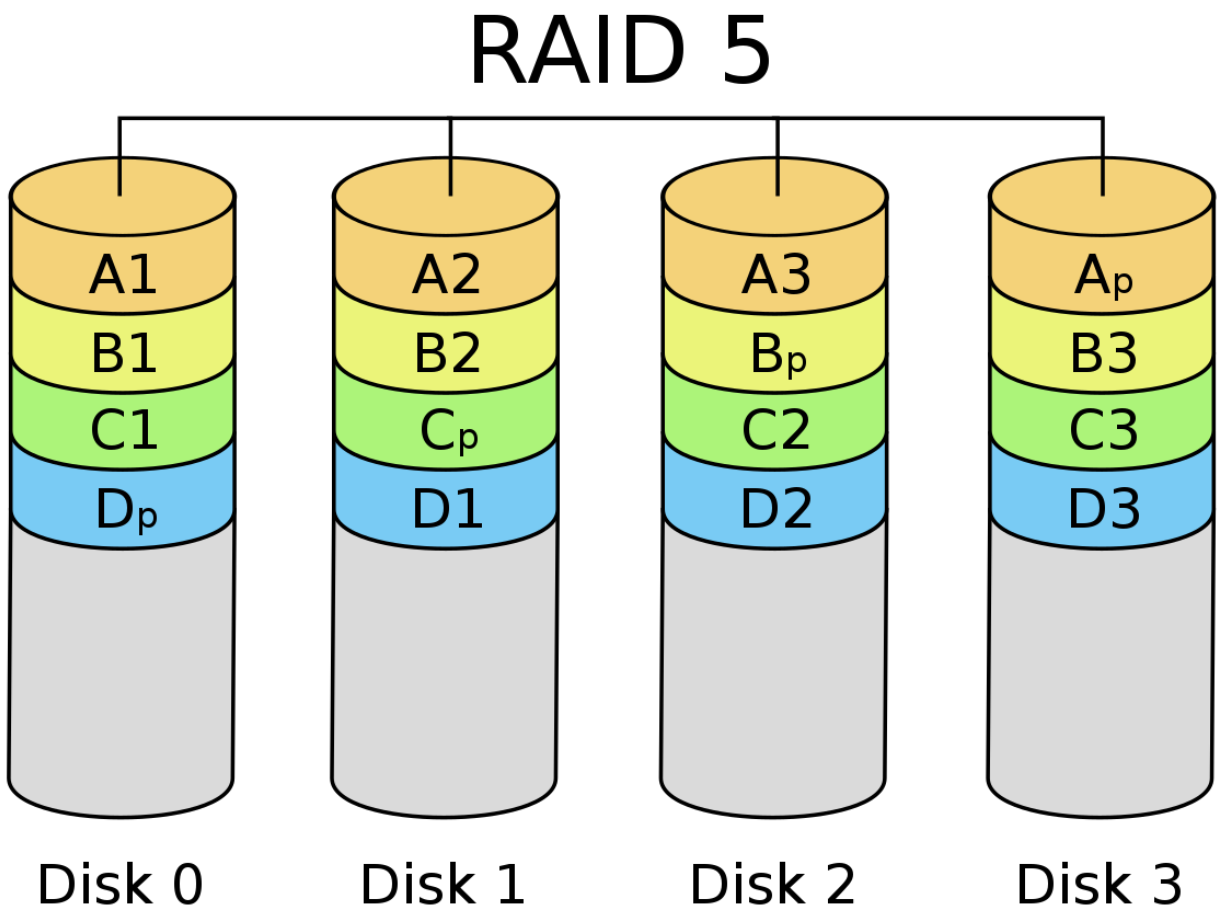
## Préparation du serveur

Avant de commencé la mise en place de l'OS, nous préparons le serveur en lui-même. Pour cela, nous allons mettre à jour le firmware du serveur, puis nous allons configurer le RAID. Une carte fibre **10Gbps** est également installée dans le serveur.
(Malheureusement, nous n'avons pas pu prendre de photos de ces étapes.)

Nous avions déjà **4 disques** durs de **900 Go**, nous avons donc décidé de les mettre en **RAID 5** pour une combinaison de rapidité de traitement et de tolérance aux pannes. Avec cette configuration, nous avons une capacité de stockage de **2,7 To**.

Nous avons également ajouté un **SSD** de **400 Go** pour y installer l'OS.

Le RAID 5 est un système de stockage qui répartit les données sur plusieurs disques durs. Il utilise la technique du striping avec répartition des données et de la parité. La parité est une technique qui permet de reconstruire les données en cas de panne d'un disque dur. Le RAID 5 nécessite au minimum 3 disques durs.



# Installation de Proxmox

> ⓘ **Qu'est-ce que Proxmox ?**
>
> Proxmox VE est une plate-forme de virtualisation open source pour exécuter des machines virtuelles et des conteneurs. Il est basé sur Debian Linux et utilise le noyau Linux KVM (QEMU) pour la virtualisation, la gestion de conteneurs LXC et un système de gestion Web intégré pour une administration facile.

Pour installer Proxmox, nous avons téléchargé l'ISO sur le site officiel de Proxmox. Nous avons ensuite créé une clé USB bootable avec l'ISO à l'aide de Balena Etcher.


Pendant l'installation, nous avons choisi l'option **Install Proxmox VE (Graphical)** afin de simplifier le processus. Ensuite, nous avons accepté les termes de la licence.

Lors du choix du disque, nous avons sélectionné le SSD de 400 Go pour l'installation du système d'exploitation. Le RAID sera dédiée au stockage des machines virtuelles.

Pour le moment nous configurons le l'adresse IP sur `172.20.15.10` située dans le réseau de la salle S2. Par la suite, le réseau sera configuré via un routeur virtuel Mikrotik, établissant une connexion en fibre optique avec la Freebox via le VLAN 6.

# Configuration du serveur Proxmox

Une fois l'installation achevée, nous avons procédé à la configuration du serveur Proxmox. Pour ce faire, nous avons ouvert un navigateur web et accédé à l'adresse `https://172.20.15.10:8006`. Suite à cela, nous avons accepté le certificat de sécurité puis nous nous sommes connectés en utilisant le nom d'utilisateur `root` ainsi que le mot de passe que nous avions défini lors de l'installation.

La configuration du serveur s'effectuera en plusieurs étapes :

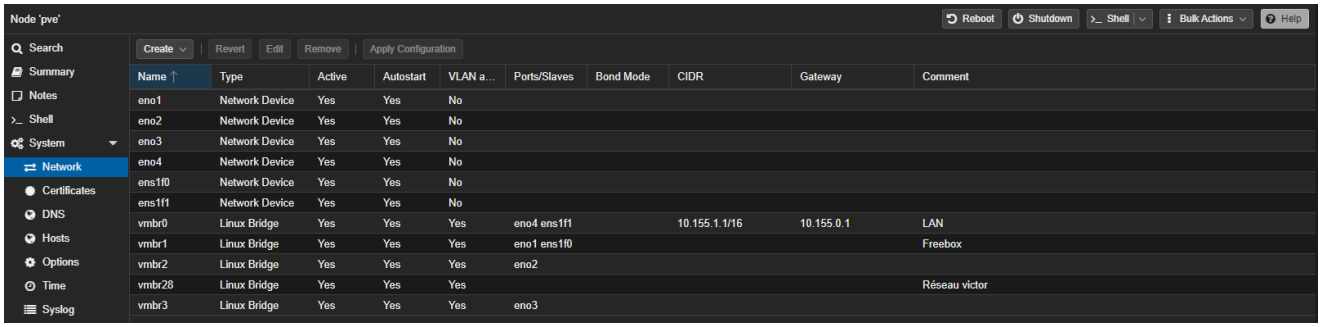
- Configuration du réseau (création des Bridges)
- Configuration du stockage (Préparation du disque pour accueillir les VM)
- Création d'une template (pour installer les VM plus rapidement)
- Création des Pools (pour différencier les VMs des utilisateurs)
- Création des Utilisateurs (ainsi que leurs droits)

## Configuration du réseau

Déjà, en accédant à **pve** > **System** > **Network**, nous constatons que la carte réseau intégrée au serveur est correctement détectée, tout comme la carte fibre à **10 Gbps** (dotée de deux ports).

Afin que les VMs puissent communiquer avec le réseau, il faut créer un bridge. Pour cela, toujours dans le même menu, on clique sur **Create** > **Linux Bridge**. le serveur aura plusieurs bridges :

- **vmbr0** : Bridge par défaut, associé aux cartes **eno4** et **ens1f1**. Il sera utilisé par les VMs et pour relier le switch **Aruba J9776A** de notre baie.
- **vmbr1** : Bridge lié aux cartes **eno1** et **ens1f0**. Il sera utilisé pour relier le switch **Aruba JL255A** de la baie du professeur.
- **vmbr2** et **vmbr3** : Bridges liés aux interfaces **eno2** et **eno3** respectivement. Bien qu'ils ne soient pas actuellement utilisés, ils sont configurés pour d'éventuels besoins futurs.
- **vmbr28** : Bridge utilisé par Victor pour ses VMs. (il dispose de son propre routeur virtuel).



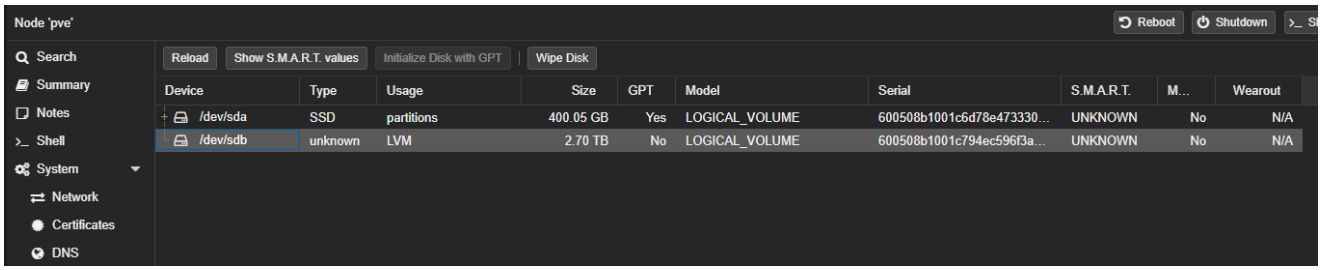

Sur le **vmbr0**, deux interfaces sont présentes, mais seule l'interface **ens1f1** est actuellement utilisée. Elle est connectée au switch **Aruba J9776A** de notre baie. L'interface **eno4** est configurée en tant que redondance, prête à être activée en cas de problème avec la fibre.

La configuration sur le **vmbr1** est similaire, à la différence que l'interface **ens1f0** est utilisée et reliée au switch **Aruba JL255A** de la baie du professeur. De la même manière, une interface de secours est présente, prête à prendre le relais en cas de besoin.

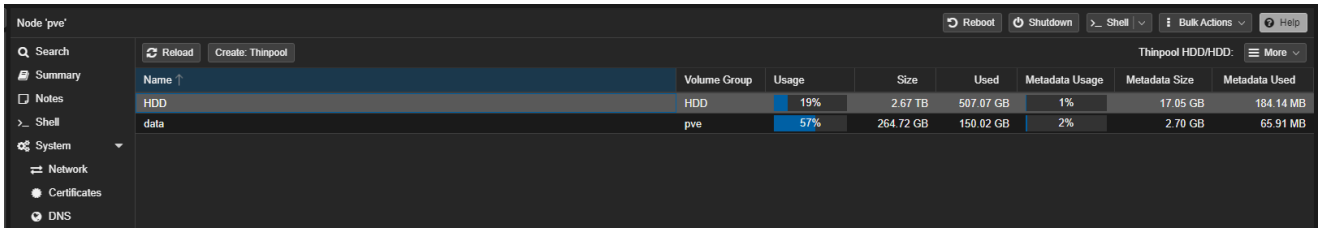
Sur chacun des bridges, nous avons activé **VLAN Aware** afin de pouvoir utiliser des VLANs.

## Configuration du stockage

Avant de commencer la configuration du stockage, nous vérifions que le système détecte correctement les disques. Pour cela, nous accédons à **pve** > **Disks**. Nous constatons que les disques SSD et HDD sont correctement détectés.




Les disques sont présents et détectés. Nous passons donc à la création du stockage. Pour cela, nous nous rendons dans **pve** > **Disk** > **LVM-Thin** > **Create**. Nous choisissons le disque HDD (représentant le RAID 5) et lui attribuons un nom. En cochant la case Add Storage, nous ajoutons le stockage au serveur. La création est validée en cliquant sur Create.




Le stockage est maintenant créé, ce qui nous permet de procéder à la création d'un modèle pour installer les machines virtuelles de manière plus rapide.

## Création d'une Template

> **Qu'est-ce qu'une Template et à quoi sert-elle ?**
>
> Une template est une image d'une VM. Elle permet de créer des VMs plus rapidement. Elle contient l'OS et des programmes de base. On peut ensuite créer des clones de cette template pour installer des VMs.

Pour créer un modèle, la procédure habituelle consiste à mettre en place une machine virtuelle (VM), y installer le système d'exploitation (OS) ainsi que les programmes de base, puis la convertir en modèle. Toutefois, pour simplifier notre démarche, les équipes de Debian et Ubuntu mettent à notre disposition des modèles préconfigurés prêts à l'emploi. Nous allons ainsi opter pour l'utilisation d'un modèle Debian 12.

Pour ce faire, nous nous rendons sur la page de téléchargement des modèles Debian : Debian Cloud Images. On cherche la distribution **Debian Bookworm** et on télécharge la template **debian-12-generic-amd64.qcow2**.

> **qcow2 est un format de fichier de stockage de machine virtuelle utilisé par QEMU.**

Une fois le téléchargement terminé, nous envoyons le fichier sur le serveur Proxmox et créons une machine virtuelle (VM) avec ce fichier. Pour simplifier la procédure, nous utiliserons l'outil en ligne de commande `qm`.

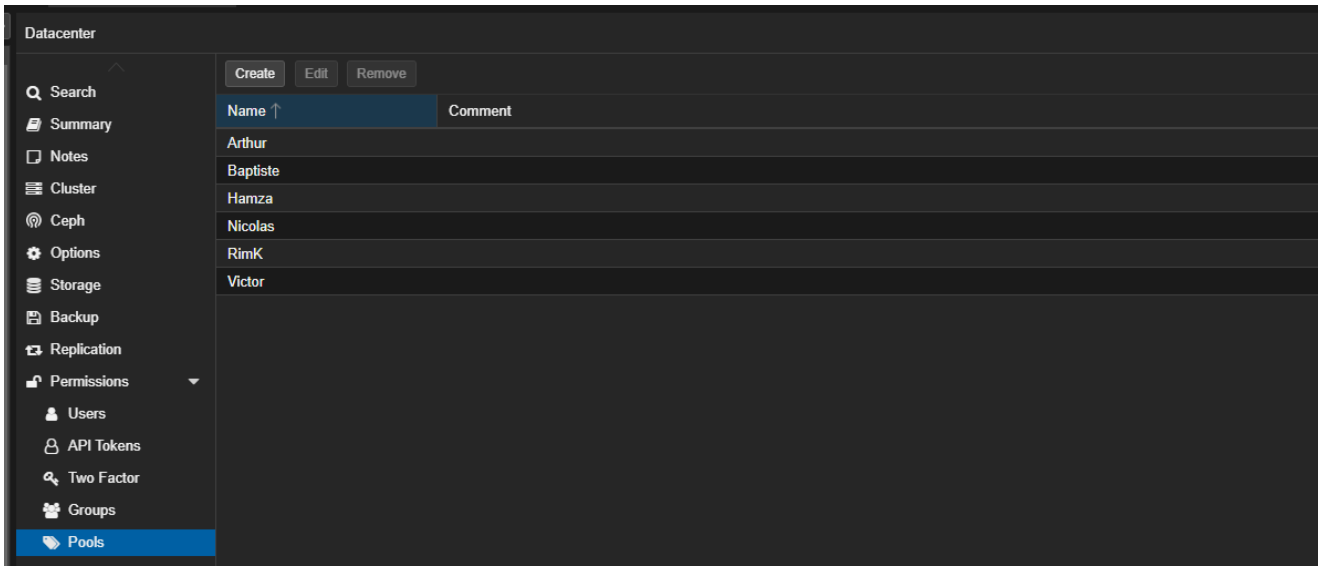
```
qm create 9000 --name debian12 --memory 2048 --net0 virtio,bridge=vmbr0
qm importdisk 9000 debian-12-generic-amd64.qcow2 HDD
qm set 9000 --scsihw virtio-scsi-pci --scsi0 HDD:vm-9000-disk-0
qm set 9000 --boot c --bootdisk scsi0
qm set 9000 --serial0 socket --vga serial0
qm set 9000 --ide2 HDD:cloudinit
qm template 9000
```

La première commande crée une machine virtuelle (VM) avec l'identifiant 9000, 2 Go de RAM et une interface réseau connectée au pont (bridge) **vmbr0**. Ensuite, nous importons le disque **debian-12-generic-amd64.qcow2** dans la VM. Nous configurons le disque en utilisant l'interface SCSI et le définissons comme disque de démarrage.
Ensuite, nous configurons le port série pour permettre l'accès à la console de la VM via le terminal. Enfin, nous mettons en place la configuration du disque **cloudinit** afin que la VM puisse récupérer les informations de configuration depuis le serveur Proxmox.

## Création des Pools

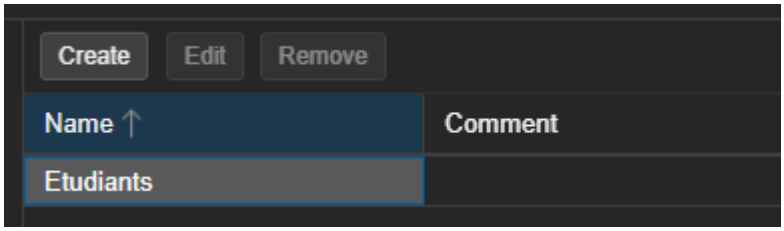
> **Qu'est-ce qu'un Pool et à quoi sert-il ?**
>
> Les pools peuvent être utilisés pour regrouper un ensemble de machines virtuelles et de banques de données. Vous pouvez ensuite simplement définir des autorisations sur les pools, qui sont héritées par tous les membres du pool. C'est un excellent moyen de simplifier le contrôle d'accès.

Pour créer un pool, nous nous rendons dans **Datacenter** > **Permissions** > **Pools** > **Create**. Nous lui attribuons un nom pour chacun des utilisateurs.

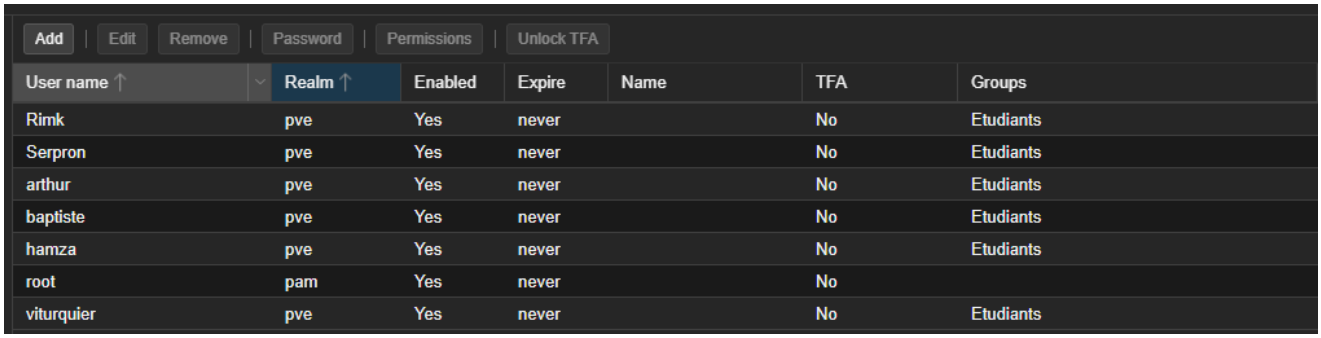


## Création des Utilisateurs

Avant de créer les utilisateurs, nous créons un groupe Etudiants dans **Datacenter** > **Permissions** > **Groups** > **Create**. Ce groupe sera utilisé pour attribuer des droits aux utilisateurs.

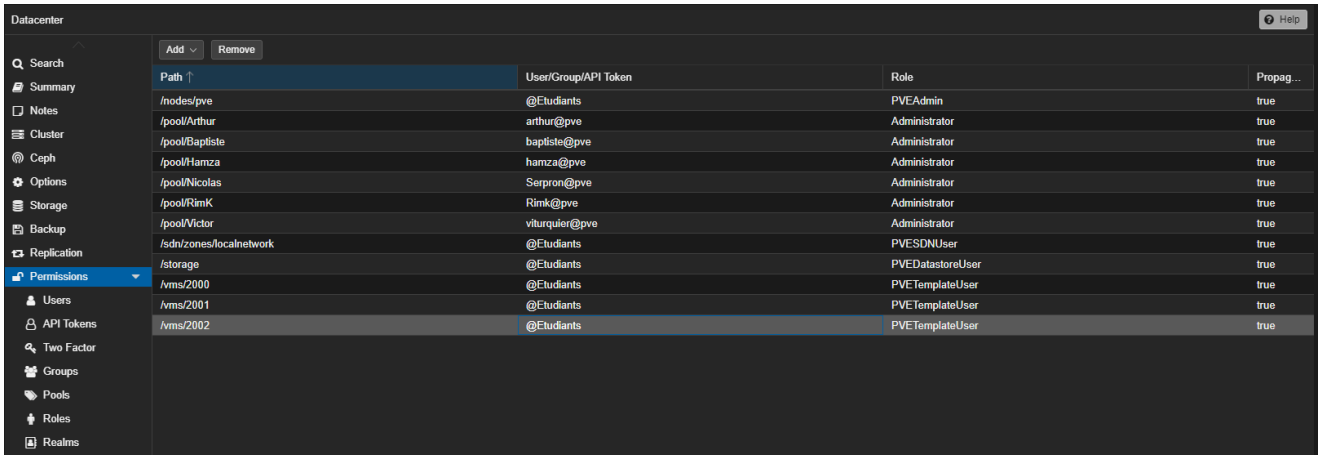



Pour créer un utilisateur, nous nous rendons dans **Datacenter** > **Permissions** > **Users** > **Add**. Nous spécifions un nom d'utilisateur, définissons un mot de passe en précisant le Realm (**Proxmox VE Authentication Server**) et attribuons l'utilisateur au groupe **Etudiants**.




| User name | Realm | Enabled | Expire | Name | TFA | Groups |
|---|---|---|---|---|---|---|
| Rimk | pve | Yes | never | | No | Etudiants |
| Serpron | pve | Yes | never | | No | Etudiants |
| arthur | pve | Yes | never | | No | Etudiants |
| baptiste | pve | Yes | never | | No | Etudiants |
| hamza | pve | Yes | never | | No | Etudiants |
| root | pam | Yes | never | | No | |
| viturquier | pve | Yes | never | | No | Etudiants |

Maintenant que les utilisateurs sont créés, il est nécessaire de leur attribuer des droits. Pour ce faire, nous accédons à **Datacenter** > **Permissions**.




Le groupe **Etudiants** est doté du rôle **PVETemplateUser**, qui autorise la création de machines virtuelles à partir de modèles. De plus, le rôle **PVEDatastoreUser** est attribué pour permettre la création de machines virtuelles sur le stockage. Enfin, le rôle **PVESDNUser** est assigné pour l'utilisation des ponts réseau (bridges).

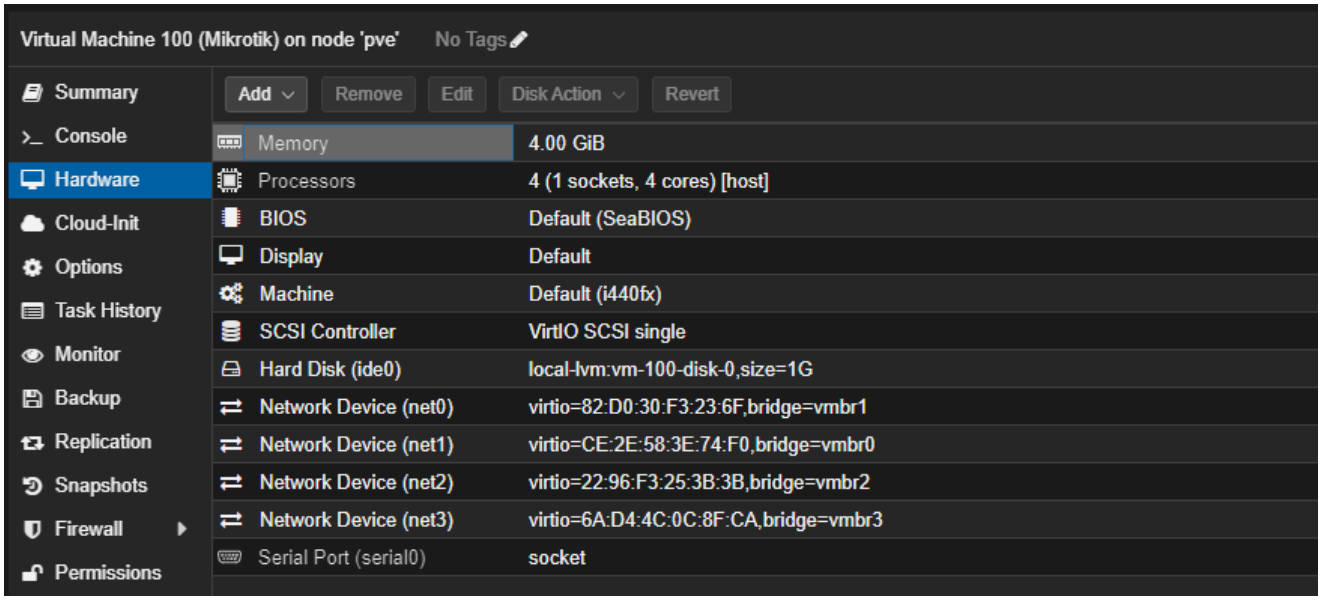
Enfin, chaque utilisateur dispose d'un accès en tant qu'administrateur à son propre pool respectif.

# Installation d’un routeur virtuel Mikrotik

> **Qu'est-ce que Mikrotik ?**
>
> MikroTik est une société lettonne qui développe des routeurs et des logiciels réseau. Le système d'exploitation du routeur est appelé RouterOS.

Pour installer le routeur virtuel Mikrotik, nous avons téléchargé l'ISO sur le site officiel de Mikrotik. Et l'avons installé sur une VM.

Le Mikrotik, étant le routeur principal du réseau, il contient tous les bridges.

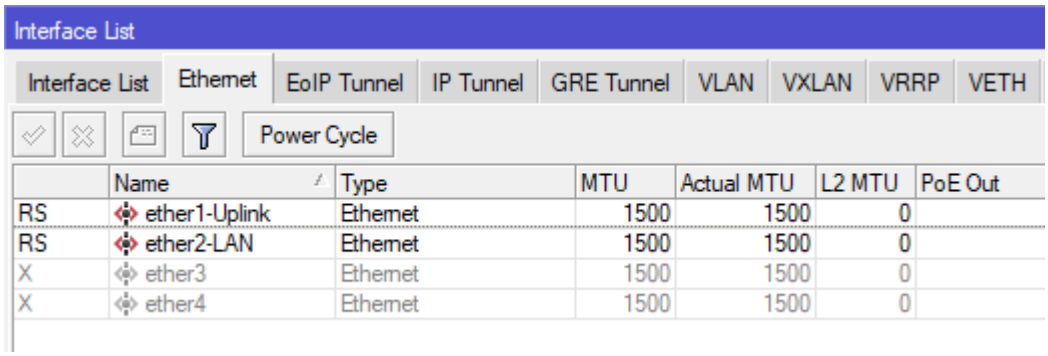



On y accède avec le logicel Winbox en utilisant directement l'adresse MAC de la VM en attendant que le routeur soit configuré.

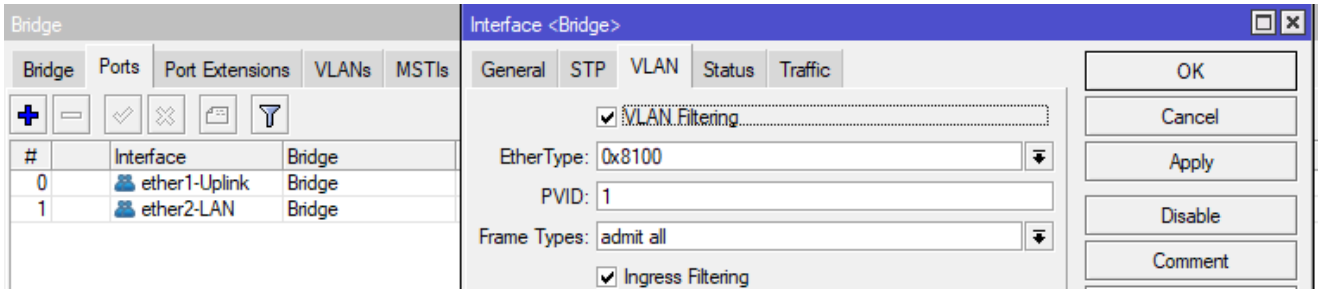
# Configuration du réseau avec des VLAN

Afin de simplifier la gestion, les différentes interfaces du routeur on été renommées :

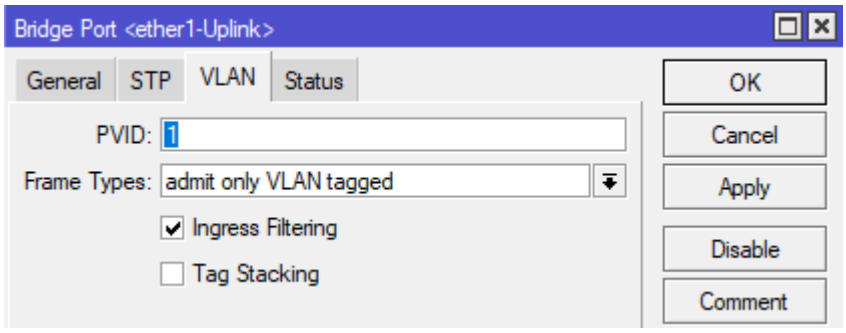
- **ether1-Uplink** : Interface connectée au switch **Aruba JL255A** de la baie du professeur.
- **ether2-LAN** : Interface connectée au switch **Aruba J9776A** de notre baie.
- **ether3** et **ether4** : Interfaces non utilisées et désactivées.




Sur le routeur, nous créons un bridge et y ajoutons les deux interfaces, **ether1-Uplink** et **ether2-LAN**. Ce bridge sera utilisé pour la gestion des VLANs sur le routeur, il est donc crucial d'activer l'option **VLAN filtering**.




Afin d'éviter que notre réseau ne se propage sur l'interface **ether1-Uplink** en raison du bridge, nous spécifions que seules les trames étiquetées (VLAN tagged) sont autorisées. Sans cette option, le routeur diffuserait notre DHCP, etc.



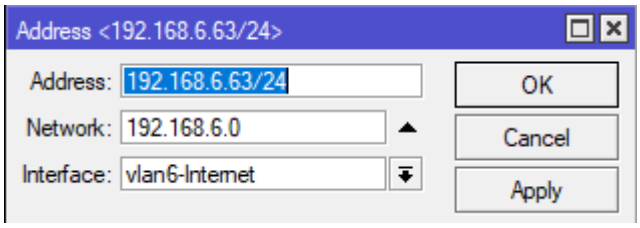

Le protocol **STP** (Spanning Tree Protocol) est également activé sur le bridge afin d'éviter les boucles de réseau.

Pour avoir accès a la Freebox (et donc à internet), nous avons **taggé le VLAN 6** sur l'interface **ether1-Uplink** et également sur **l'interface 27** du switch **Aruba JL255A** de la baie du professeur (interface auquel nous sommes connectés). Ce VLAN est utilisé par le professeur pour connecter son réseau à la Freebox.

```
vlan 6
   name "internet freebox"
   untagged 19
   tagged 12,18,24,27,Trk1
   no ip address
   exit
```

| | | |
|---|---|---|
| RS | ether1-Uplink | Ethernet |
| R | vlan6-Internet | VLAN |

Il suffit ensuite de mettre une adresse IP sur l'interface **Vlan6-Internet** et le routeur aura accès à internet.



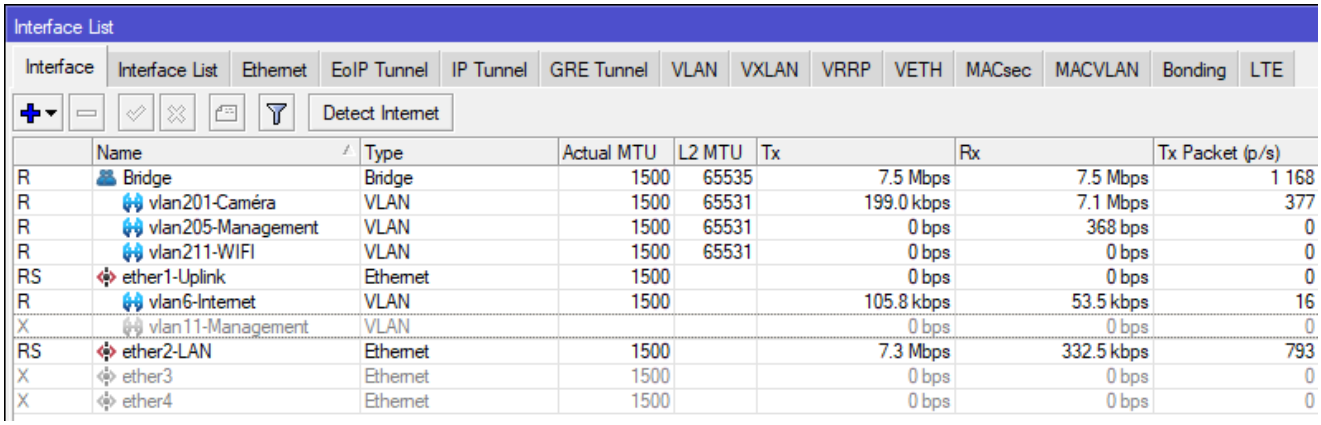

```
[admin@MikroTik] > ping 1.1.1.1
  SEQ HOST                                     SIZE TTL TIME       STATUS
    0 1.1.1.1                                    56  57 6ms615us
    1 1.1.1.1                                    56  57 6ms66us
    sent=2 received=2 packet-loss=0% min-rtt=6ms66us avg-rtt=6ms340us max-rtt=6ms615us
```

Nous en profitons également pour attribuer une adresse IP à l'interface **Bridge** afin de pouvoir accéder au routeur depuis le réseau local. Nous avons choisi l'adresse `10.155.0.1/16` pour le routeur.

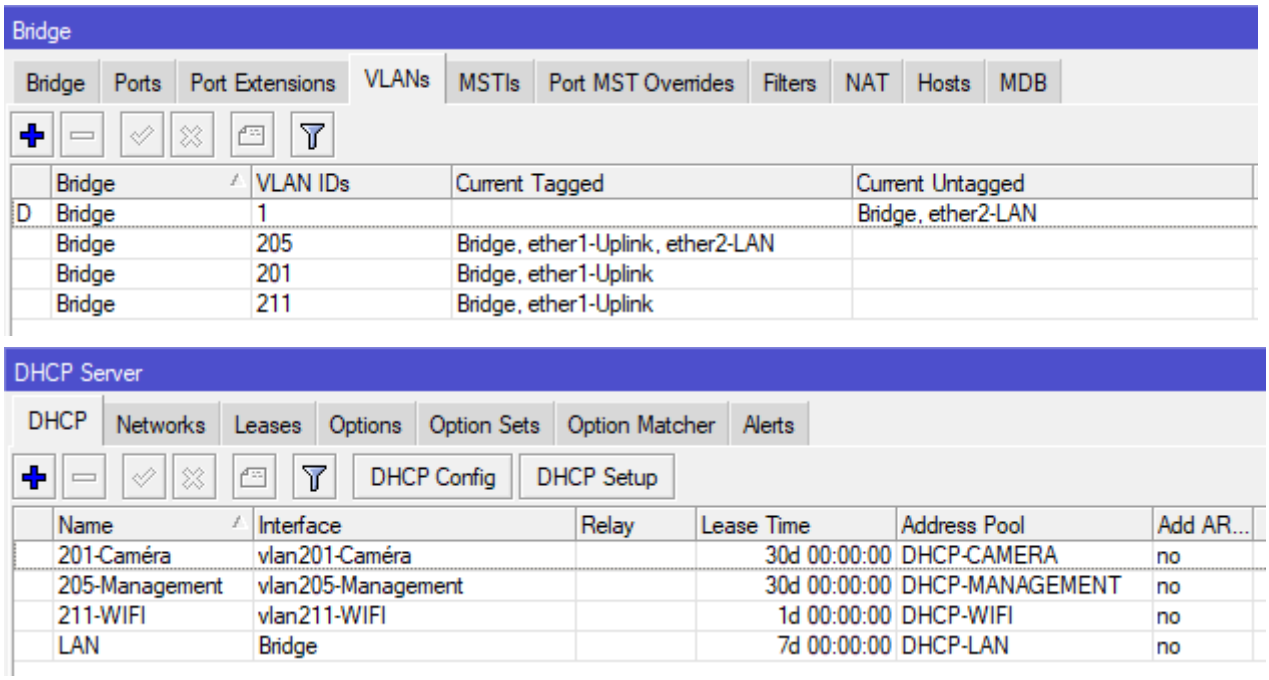
Divers VLAN sont créés pour les différents services :

- **Vlan201-Caméras** ( `192.168.201.1/24` ) : VLAN dédié aux caméras.
- **Vlan205-Management** ( `192.168.205.1/14` ) : VLAN pour la gestion des équipements (switch, routeur, UPS...)
- **Vlan211-Wifi** : ( `192.168.211.1/14` ) : VLAN pour le Wifi.

Ils sont liés au bridge et sont configurés pour être utilisés en tant que VLANs tagged. De plus, nous avons activé le serveur DHCP sur chacun des VLANs.




| | Name | Type | Actual MTU | L2 MTU | Tx | Rx | Tx Packet (p/s) |
|---|---|---|---|---|---|---|---|
| R | Bridge | Bridge | 1500 | 65535 | 7.5 Mbps | 7.5 Mbps | 1 168 |
| R | vlan201-Caméra | VLAN | 1500 | 65531 | 199.0 kbps | 7.1 Mbps | 377 |
| R | vlan205-Management | VLAN | 1500 | 65531 | 0 bps | 368 bps | 0 |
| R | vlan211-WIFI | VLAN | 1500 | 65531 | 0 bps | 0 bps | 0 |
| RS | ether1-Uplink | Ethernet | 1500 | | 0 bps | 0 bps | 0 |
| R | vlan6-Internet | VLAN | 1500 | | 105.8 kbps | 53.5 kbps | 16 |
| X | vlan11-Management | VLAN | | | 0 bps | 0 bps | 0 |
| RS | ether2-LAN | Ethernet | 1500 | | 7.3 Mbps | 332.5 kbps | 793 |
| X | ether3 | Ethernet | 1500 | | 0 bps | 0 bps | 0 |
| X | ether4 | Ethernet | 1500 | | 0 bps | 0 bps | 0 |




Ces VLANs taggés sont ensuite configurés sur les interfaces du switch du professeur (taggés sur **l'interface 27**, qui est le lien fibre en direction de notre routeur Mikrotik).

```
vlan 201
   name "sio-camera"
   untagged 13-14
   tagged 27
   no ip address
   exit
vlan 205
   name "sio-management"
   tagged 27
   ip address 192.168.205.101 255.255.255.0
   exit
vlan 211
   name "sio-WIFI"
   untagged 15
   tagged 27
   no ip address
   exit
```

Le VLAN 201, dédié aux caméras, est **untaggé** sur les **interfaces 13 et 14** du switch du professeur, permettant ainsi aux caméras de communiquer avec le routeur. (Les caméras sont connectées à ce switch pour être alimentées en POE.)

Le VLAN 211, destiné au Wifi, est également **untaggé** sur l'interface **15** du switch du professeur, autorisant ainsi les clients à se connecter au réseau Wifi. (La configuration du Wifi n'a pas encore été réalisée, et nous envisageons une solution avec un **portail captif** qui sera implémentée ultérieurement dans le projet.)

Enfin, le VLAN 205, réservé à la gestion, est simplement utilisé sur ce switch pour permettre une connexion en SSH.

# Installation d’un agent DVR

> **Qu'est-ce qu'un agent DVR ?**
>
> Un agent DVR est un logiciel qui permet de gérer des caméras de surveillance. Il permet de visualiser les caméras, de les enregistrer, de les configurer, etc.

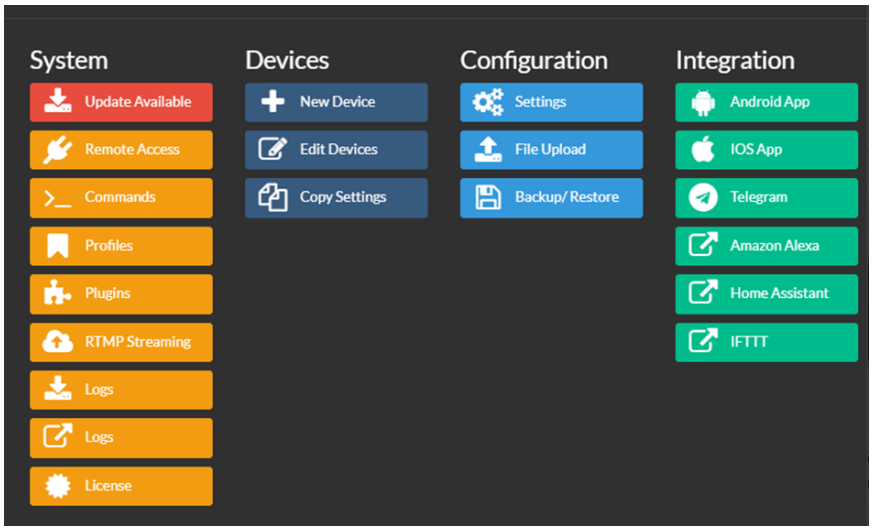
Pour installer l'agent DVR, nous avons utiliser la template Debian 12 créée précédemment. Nous avons ensuite installé l'agent DVR en exécutant les commandes précisées dans la documentation officielle : Agent DVR Documentation.

```
bash <(curl -s "https://raw.githubusercontent.com/ispysoftware/agent-install-scripts/main/v2/install.sh")
```

L'agent DVR est maintenant installé sur notre VM. Nous pouvons y accéder en utilisant l'adresse IP de la VM sur le port **8090**.

# Configuration de l’agent DVR

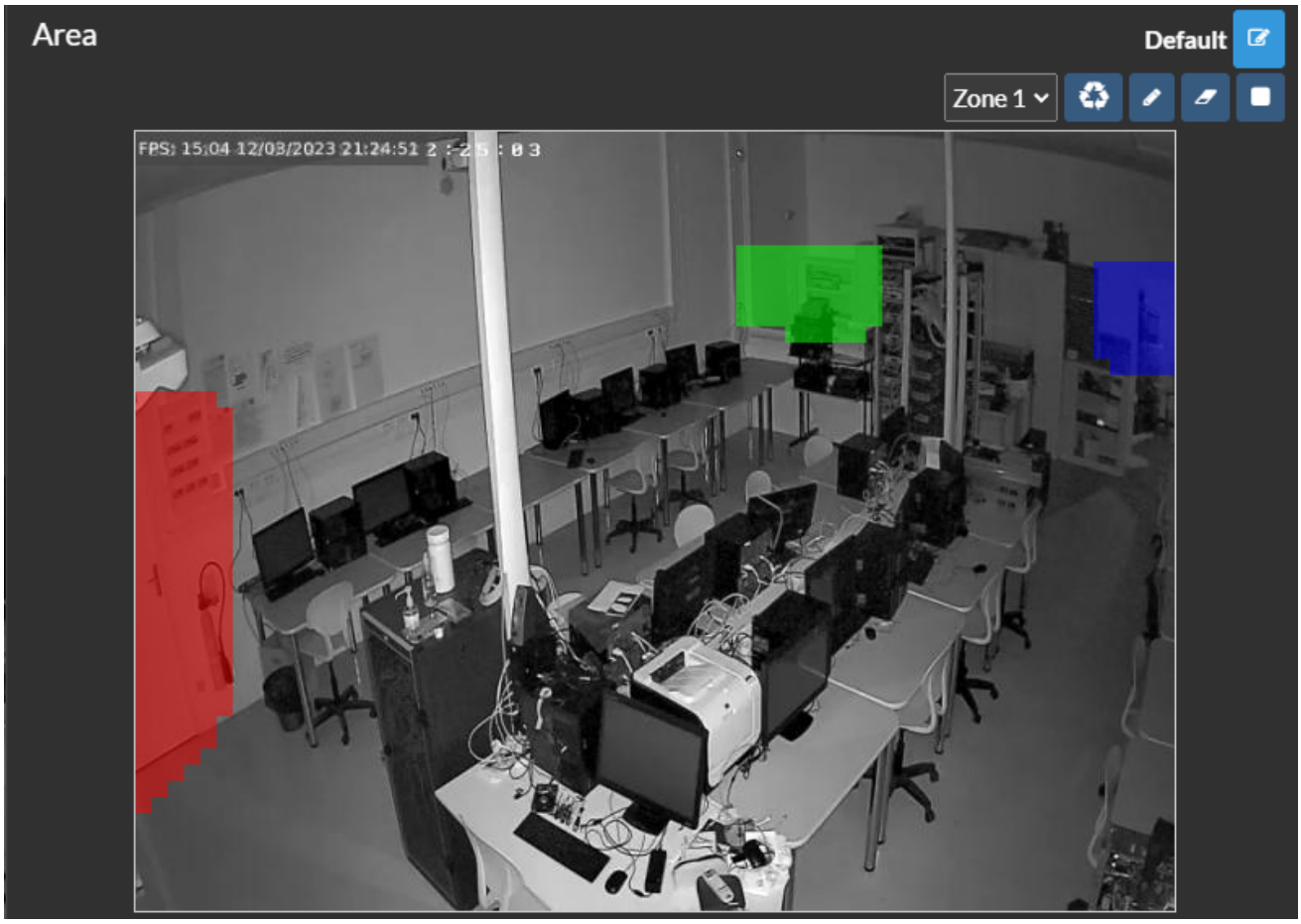
Sur l'interface web, nous nous dirigeons vers les paramètres pour configurer les caméras. Pour ajouter une caméra, nous cliquons sur New device et précisons qu'il s'agit d'une caméra IP ONVIF.




On nomme la caméra et sur **Source Type**, on sélectionne **ONVIF**.

Nous spécifions l'adresse IP de la caméra ainsi que les identifiants de connexion. Étant donné qu'il existe plusieurs Live URL pour les différents flux de la caméra, nous choisissons celui en 720x480 pour le streaming, car il sera moins gourmand en ressources. En outre, nous sélectionnons le flux en 2304x1296 pour l'enregistrement afin d'obtenir une meilleure qualité.


Ensuite, dans l'onglet **Detector**, nous activons la détection de mouvement et spécifions la zone de détection. Ainsi, la caméra enregistrera uniquement lorsqu'il y aura du mouvement dans la zone définie, ce qui permet d'éviter de saturer le disque dur avec des enregistrements inutiles.



# Installation d’un serveur VPN

> **Quelle solution VPN avons-nous choisi ?**
>
> Nous avons opté pour **WireGuard** comme solution VPN. WireGuard est un logiciel libre et open source. Pour simplifier la configuration, nous avons intégré un panel d'administration appelé **Wireguard-UI**.

De manière similaire à l'agent DVR, nous avons utilisé la template Debian 12 pour installer le serveur VPN, en lui attribuant une adresse IP via la configuration **cloudinit**.

L'installation de **Wireguard** s'est effectuée avec la commande `apt install wireguard`. Ensuite, nous avons téléchargé la dernière version de **Wireguard-UI** depuis le GitHub officiel, décompressé l'archive, et déplacé le fichier **wireguard-ui** dans le répertoire `/usr/local/bin`.

Afin de faciliter le démarrage du serveur, nous avons créé trois fichiers de service systemd. Le premier est le fichier `/etc/systemd/system/wireguard-ui.service` avec le contenu suivant :

```
[Unit]
Description=Wireguard-ui
After=network.target

[Service]
WorkingDirectory=/etc/wireguard
Environment="WGUI_DEFAULT_CLIENT_ALLOWED_IPS=10.252.1.0/24,10.155.0.0/16,10.156.0.0/16"
Environment="WGUI_DEFAULT_CLIENT_USE_SERVER_DNS=false"
ExecStart=/usr/local/bin/wireguard-ui
Restart=on-failure

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

Le deuxième est le fichier `/etc/systemd/system/wg.path` avec le contenu suivant :

```
[Unit]
Description=Watch /etc/wireguard/wg0.conf for changes

[Path]
PathModified=/etc/wireguard/wg0.conf

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

Et le dernier est le fichier `/etc/systemd/system/wg.service` avec le contenu suivant :

```
[Unit]
Description=Restart WireGuard
After=network.target

[Service]
Type=oneshot
ExecStart=/usr/bin/systemctl restart wg-quick@wg0.service

[Install]
RequiredBy=wg.path
```

Ensuite, nous avons activé les services avec les commandes suivantes :

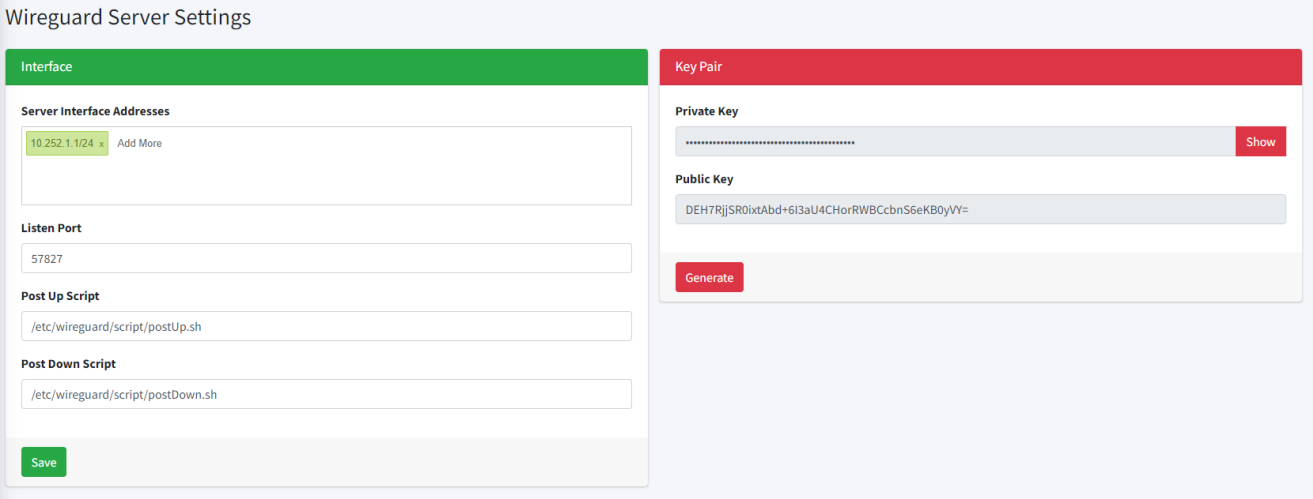
```
systemctl enable wg.service
systemctl enable wg.path
systemctl enable wireguard-ui.service
```

Ces services assurent le redémarrage du serveur **WireGuard** lorsqu'une modification est apportée au fichier de configuration.

# Configuration du serveur VPN

Pour configurer le serveur VPN, nous accédons à l'interface web de **Wireguard-UI** en utilisant les identifiants `admin` et le mot de passe `admin`.

Nous débutons en configurant le serveur dans la section **Wireguard Server**. Nous spécifions les adresses qui seront utilisées par les clients, le port, et la clé privée (qui est générée automatiquement).

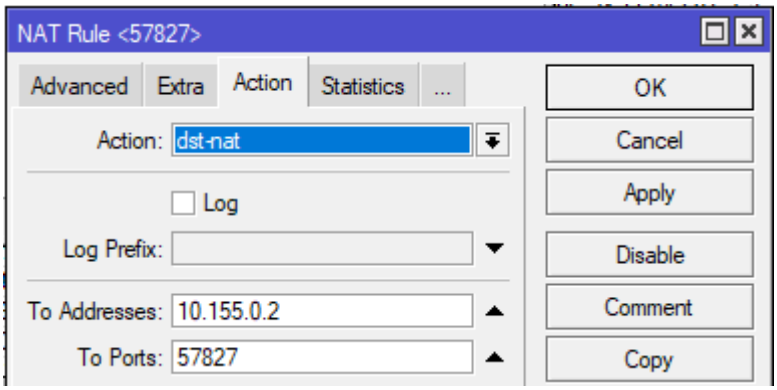



Le script **PostUp** et **PostDown** permettent de configurer le serveur pour qu'il puisse router le trafic des clients. Mais également pour faire quelque **filtrage de paquets**. (Par exemple, nous avons bloqué les paquets en direction d'internet pour éviter que les clients utilisent le VPN naviguer sur internet depuis le serveur. **Ce serveur est dédié à l'accès au réseau local**.)

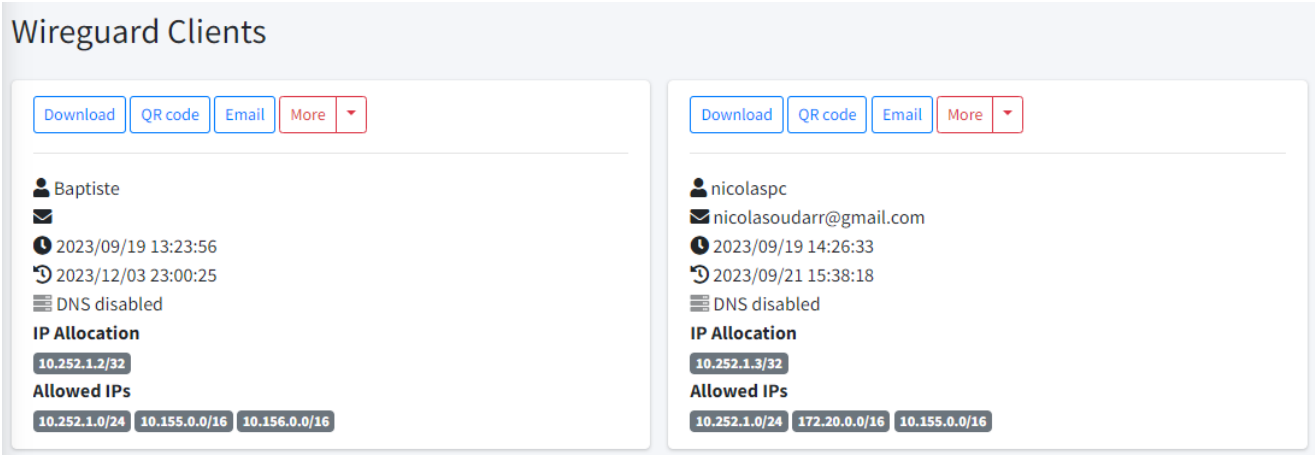
```
#!/bin/bash
iptables -I INPUT -p udp --dport 57827 -j ACCEPT
iptables -I FORWARD -i eth0 -o wg0 -j ACCEPT
iptables -I FORWARD -i wg0 -j ACCEPT
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

# DROP Trafic from VPN to external network
iptables -t mangle -A PREROUTING -s 10.252.1.0/24 -d 192.168.0.0/16 -j ACCEPT
iptables -t mangle -A PREROUTING -s 10.252.1.0/24 -d 172.16.0.0/12 -j ACCEPT
iptables -t mangle -A PREROUTING -s 10.252.1.0/24 -d 10.0.0.0/8 -j ACCEPT
iptables -t mangle -A PREROUTING -s 10.252.1.0/24 -j DROP
```

Ne pas oublier de faire une redirection de port sur le routeur pour que les clients puissent se connecter au serveur VPN. Pour cela, nous avons redirigé le port **57827** vers l'adresse IP du serveur VPN.




Nous pouvons dès a présent ajouter des clients dans la section **Wireguard Clients**. Pour cela, en cliquant sur **New Client**, nous spécifions un nom. Ensuite, le script génère automatiquement une clé privée et une clé publique. Nous pouvons maintenant télécharger le fichier de configuration du client.



# Configuration de la connexion sans fil

**Quel équipement sans fil avons-nous choisi ?**

Nous avons opté pour un équipement sans fil Cisco AIR-AP1141N-E. Cet équipement est un point d'accès sans fil qui permet de fournir une connectivité réseau sans fil.

**Nous ne l'avons pas encore mis en place, mais nous envisageons d'utiliser un portail captif pour sécuriser l'accès au réseau Wifi. Nous mettrons à jour ce document lorsque nous aurons mis en place cette solution.**