BIGEAU Anthony – GIOVANNENGELI Julien

Bachelor 3 Informatique – 2020/2021

Rapport de projet



Sommaire

[Introduction 2](#_Toc70605615)

[Architecture réseau 3](#_Toc70605616)

[Configuration de la box et du routeur 4](#_Toc70605617)

[Configuration de la BOX 4](#_Toc70605618)

[Configuration du routeur 4](#_Toc70605619)

[Mappage de port 7](#_Toc70605620)

[Configuration et sécurité du Raspberry Pi 8](#_Toc70605621)

[Configuration 8](#_Toc70605622)

[Aspect sécurité 8](#_Toc70605623)

[Sauvegarde Raspberry Pi 9](#_Toc70605624)

[VPN 9](#_Toc70605625)

[Partie serveur et routeur / Box 9](#_Toc70605626)

[Partie client (Windows 10) 10](#_Toc70605627)

[Virtualisation (Proxmox) 10](#_Toc70605628)

[Configuration de base 10](#_Toc70605629)

[Haute disponibilité 12](#_Toc70605630)

[Réplication 12](#_Toc70605631)

[Configuration et sécurisation du NAS 12](#_Toc70605632)

[Configuration 12](#_Toc70605633)

[Sécurité 12](#_Toc70605634)

[Monitoring 14](#_Toc70605635)

[Installation 14](#_Toc70605636)

[Envoi de données sur le promethus 15](#_Toc70605637)

[Conclusion 16](#_Toc70605638)

# Introduction

Ce projet a commencé le 2 mars 2021 et se termine le 27 avril 2021. Le projet est fait en binôme : Anthony BIGEAU et Julien GIOVANNANGELI. L’objectif de celui-ci est de créer une architecture réseau complète dans une maison ou dans un local de A à Z avec un aspect sécurité et des services pouvant être utilisés par des particuliers comme des entreprises. Il s’intitule LANControl.

Ce rapport sera constitué dans un premier temps d’une présentation globale de l’infrastructure, puis, dans un second temps, il y aura une partie documentation avec l’ensemble détaillé de la création à l’essai final en passant par la configuration de chaque service présent sur le réseau.

Différents outils sont utilisés dans le cadre de ce projet :

* Github pour y mettre la documentation : <https://github.com/Celestarien/LANControl>
* Trello pour faire de la planification et avoir une organisation entre collaborateurs : <https://trello.com/b/rbtZIhwU/lancontrol>
* Discord pour la communication, partager des documents et pour avoir un flux d’alertes.

# Architecture réseau

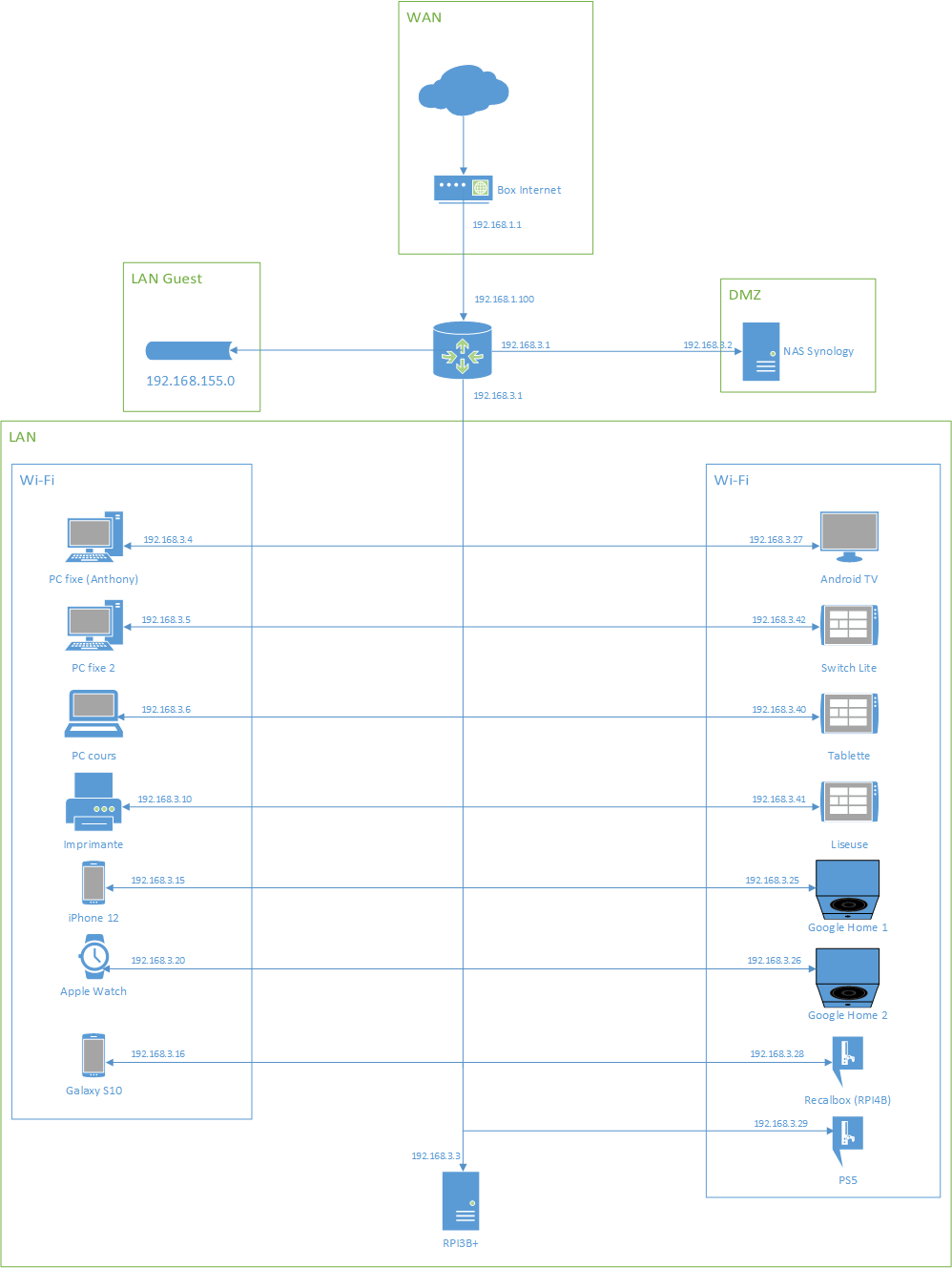


Schéma 1 : Cartographie du réseau LANControl

Notons toutefois qu’il faut savoir que « PC fixe (Anthony) » a lui une carte réseau intégrée à la carte mère (avec un port Ethernet) on lui a attribué l’IP 192.168.3.7.

# Configuration de la box et du routeur

## Configuration de la BOX

Le modèle en notre possession est une 4GBox de chez Bouygues Télécom de marque HUAWEI :



Sur celle-ci, nous avons uniquement désactivé le Wi-Fi et le pare-feu puisque c’est le routeur connecté à celle-ci qui fera la connectivité et la sécurité, a Box étant uniquement là pour apporter du débit Internet (et servir de passerelle). Pour désactiver le Wi-Fi, il suffit de se connecter à l’interface et accéder aux paramètres Wi-Fi (« Paramètres de base Wi-Fi ») puis enfin tout désactiver.

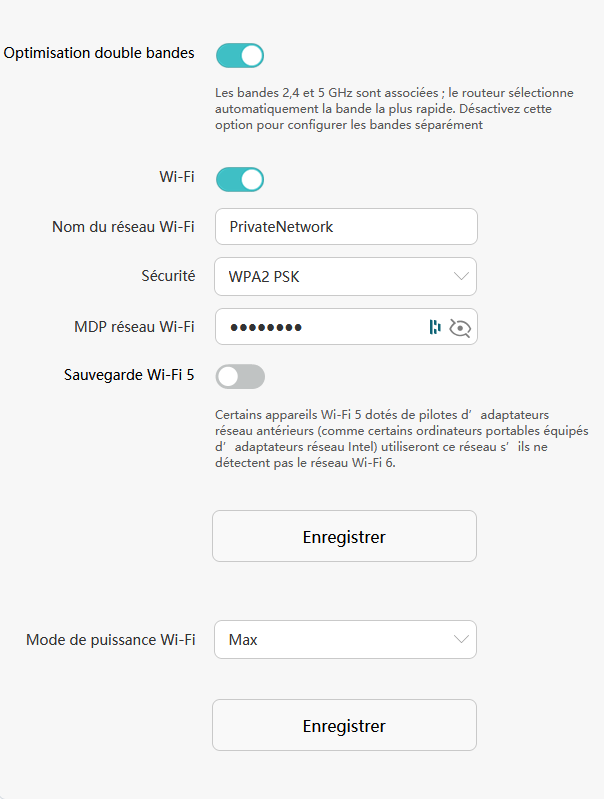
La Box va redémarrer et vous n’aurez normalement plus la LED Wi-Fi présente à l’avant de la 4GBox allumé.

Pour le pare-feu rien de plus simple, il faut aller dans Avancé 🡪 Sécurité 🡪 Commutateur pare-feu et désactiver le pare-feu / enregistrer.

## Configuration du routeur

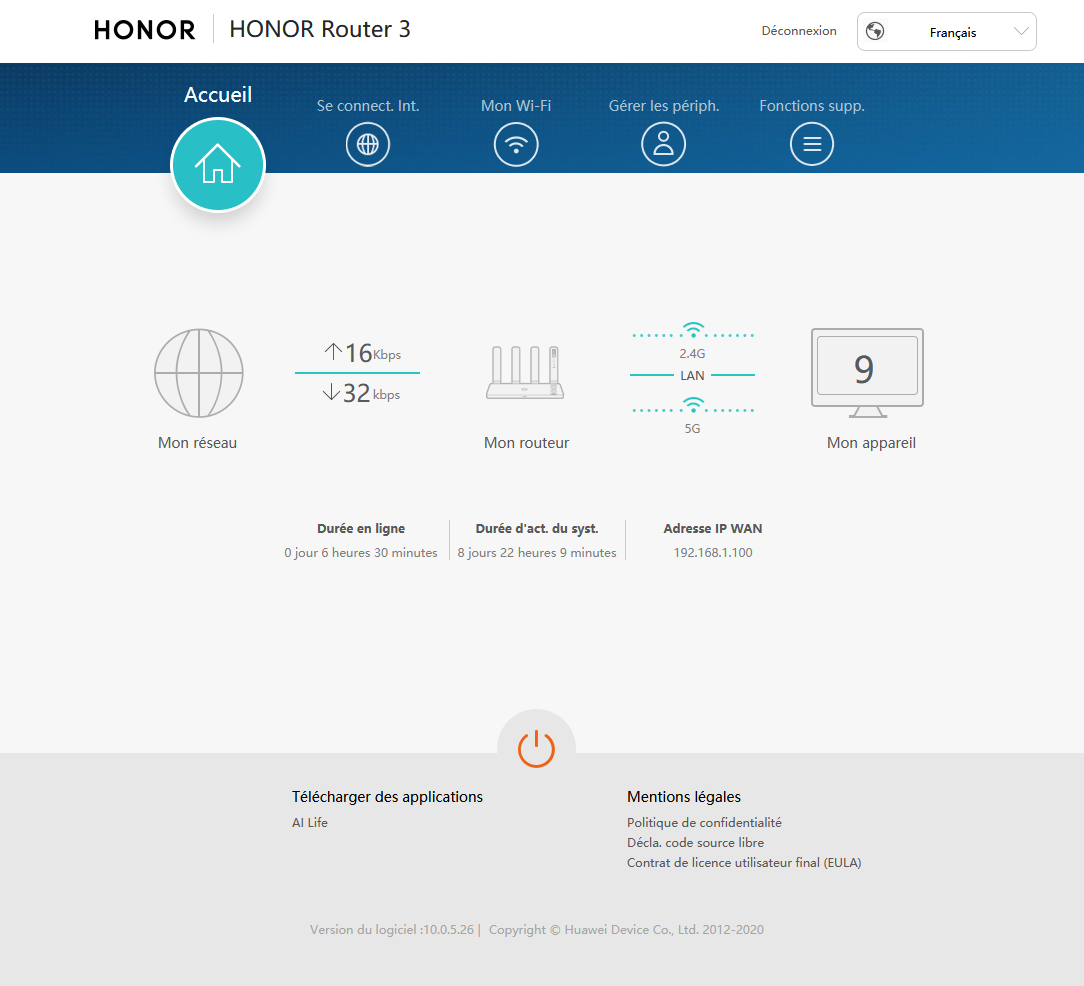
Pour le routeur, un modèle standard et trouvable facilement dans le commerce a été choisi : le routeur HONOR 3 (comportant la dernière technologie Wi-Fi 6). HONOR étant une filiale de HUAWEI, l’interface est quasi identique à celle de la 4GBox Bouygues Télécom.

Dans un premier temps, dans un premier temps il faut accéder à l’interface du routeur et suivre l’installation de base. Pour ce qui est de la partie réseau, voici une capture écran de la configuration :

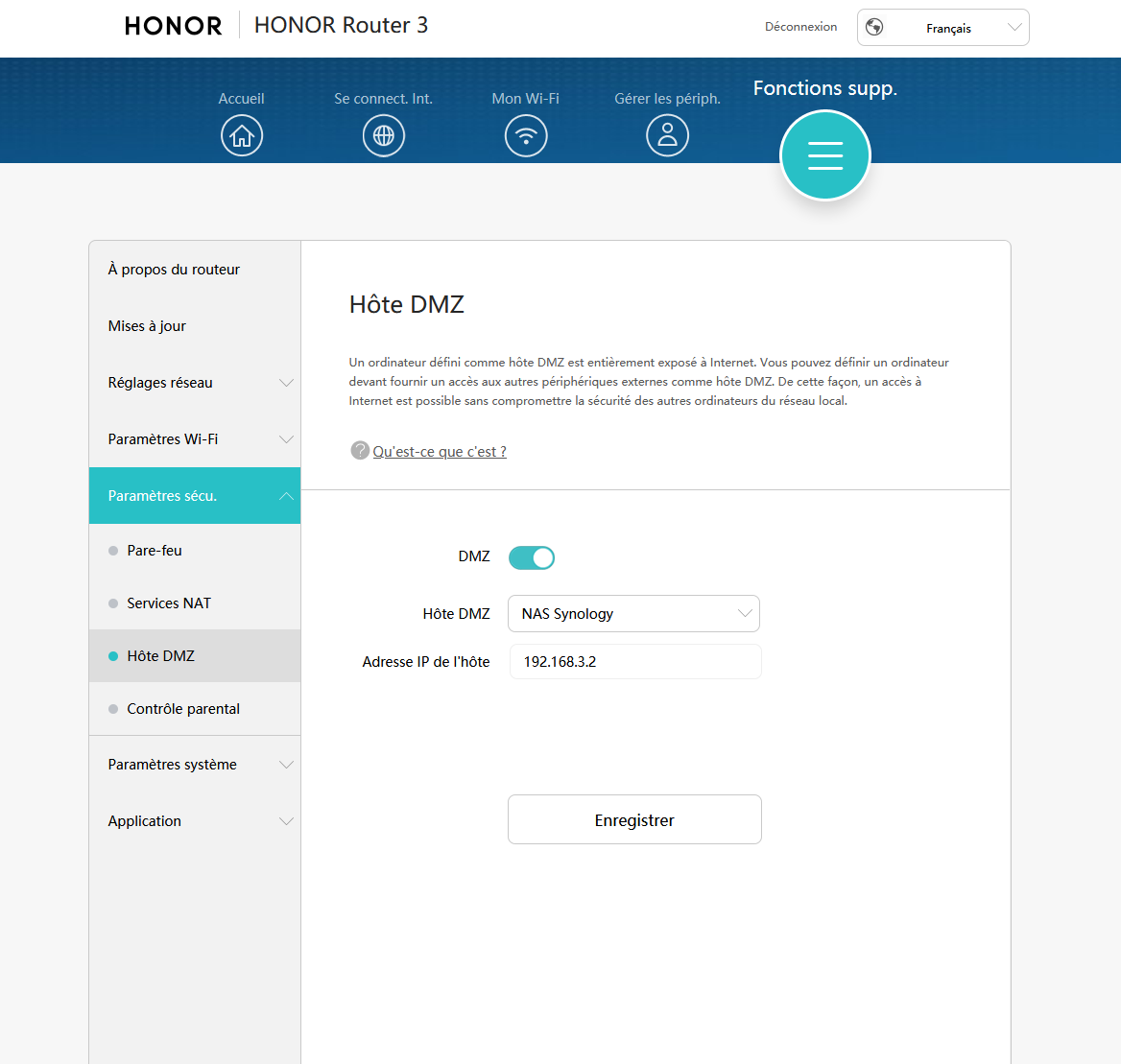


Nous avons décidé de désactiver la sauvegarde Wi-Fi 5, car l’ensemble des appareils détecte dans notre cas, le réseau « de base ».

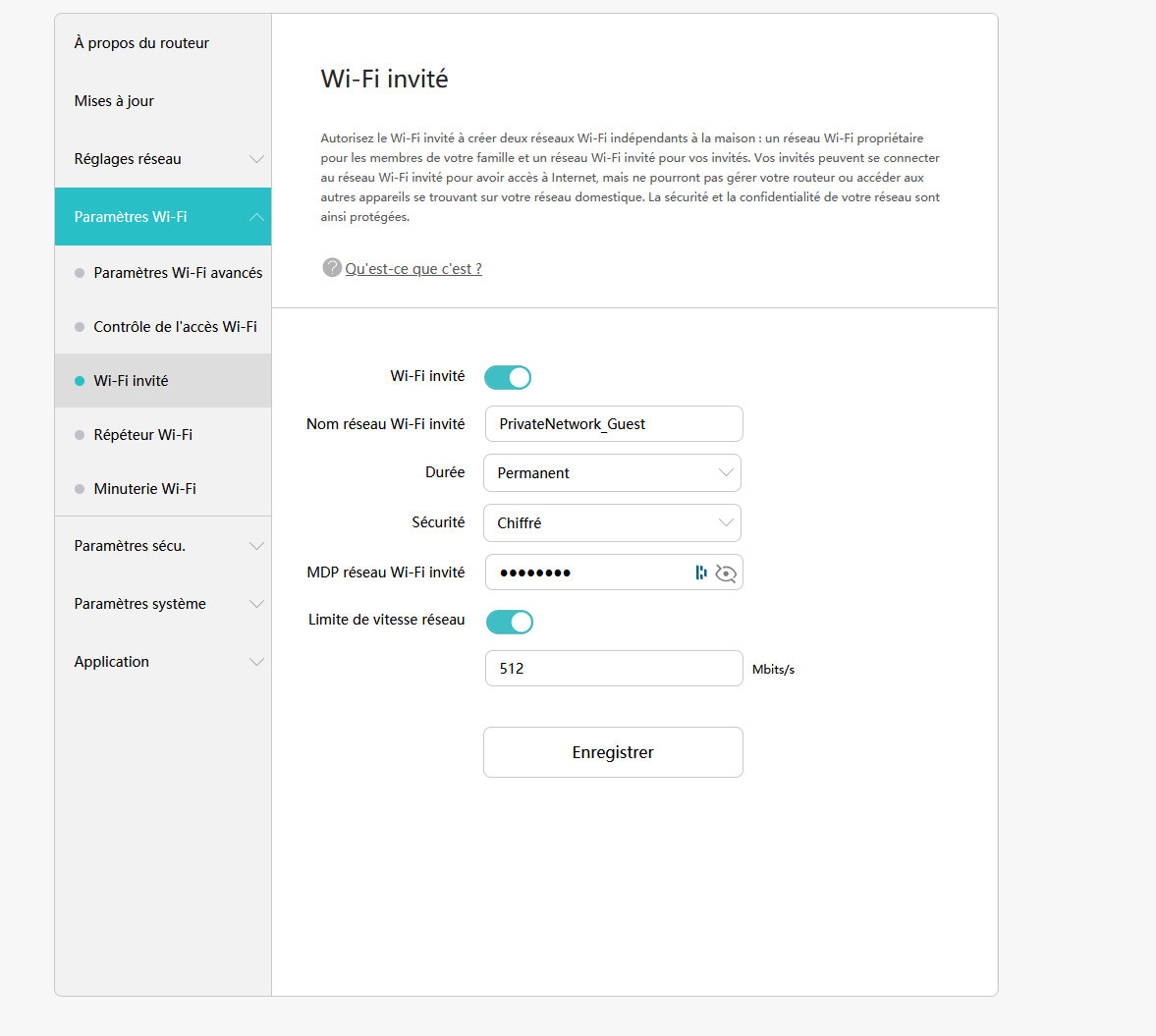
Au départ, nous arrivons sur cette interface qui nous donne certains éléments essentiels :



Ensuite concernant la DMZ rien de difficile, il suffit d’accéder aux fonctions supplémentaires et paramétrer comme ci-dessous :

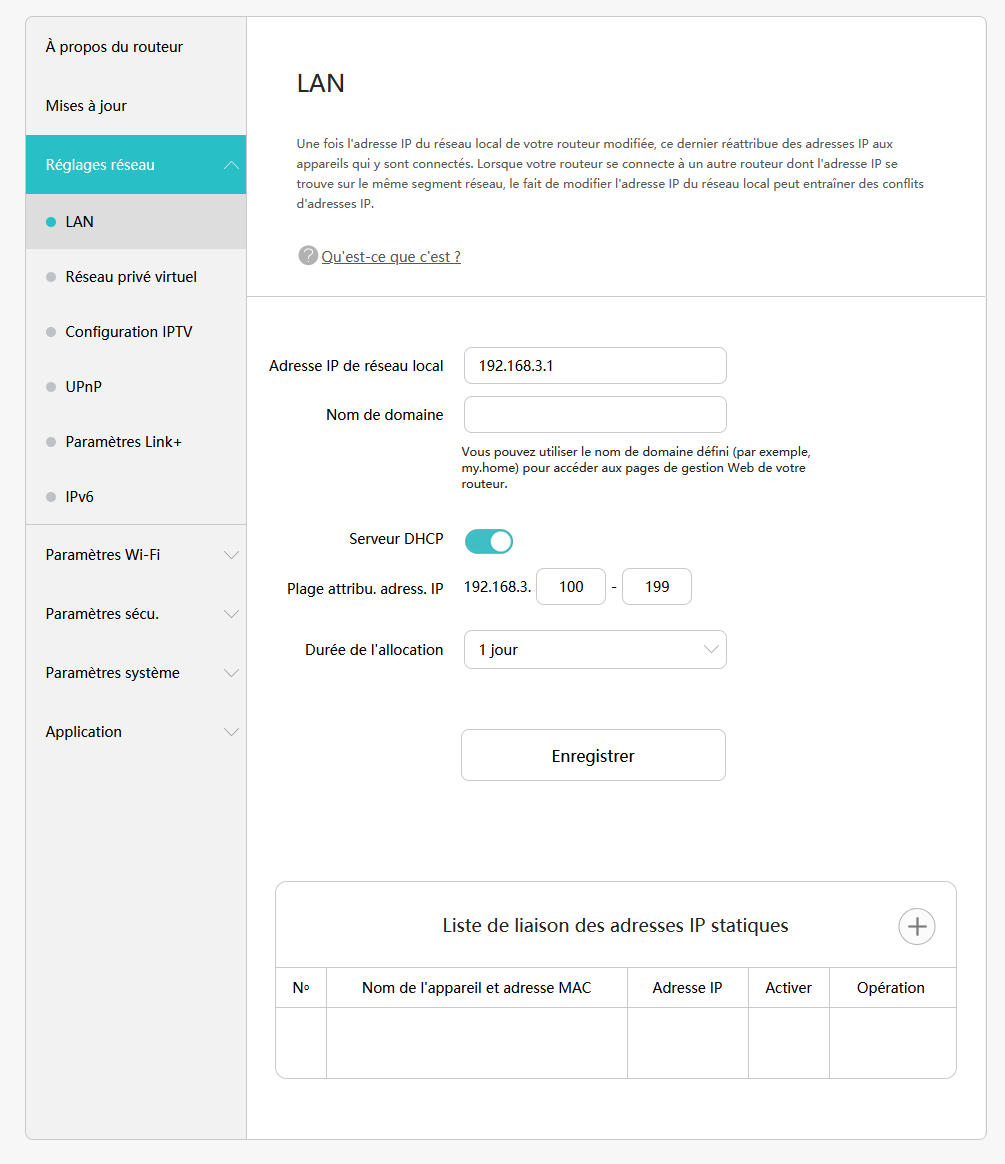


Pour la partie Wi-Fi Invité rien de difficile également, il suffit de paramétrer comme ci-dessous :

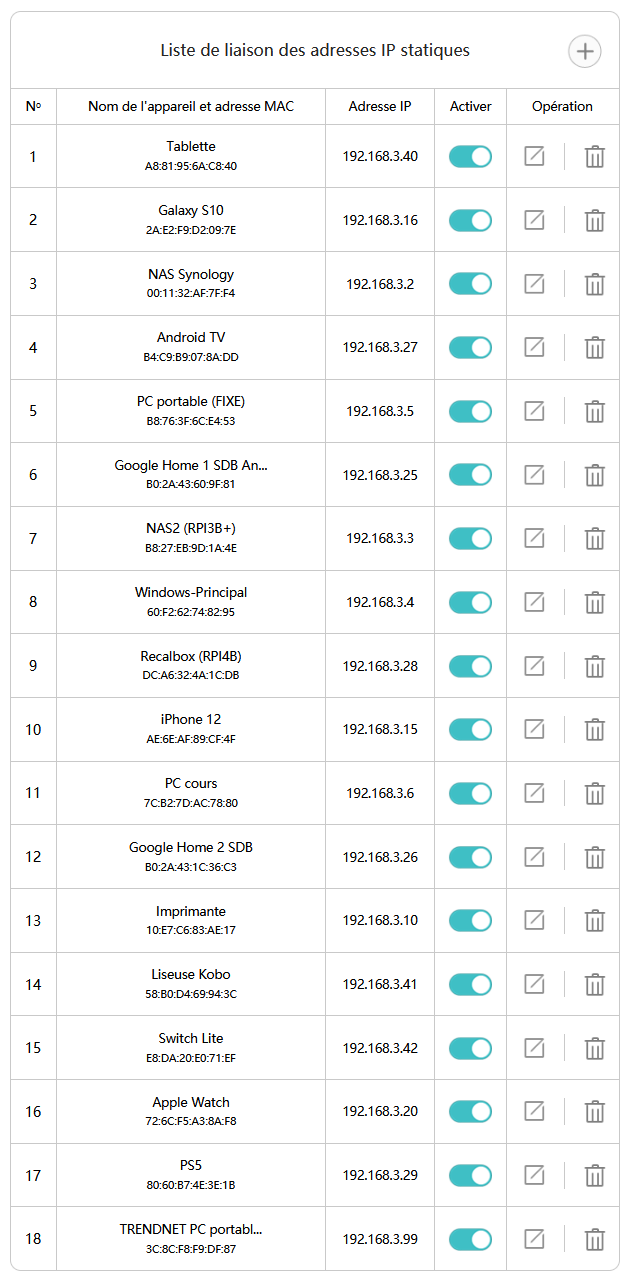


Nous avons laissé la limite de vitesse réseau par défaut, mais celle-ci peut être modifiée si nécessaire.

Pour finir, le plus important, il faut faire l’adressage réseau, notons que l’on peut d’ailleurs gérer la vitesse réseau de chaque appareil dans l’onglet « Gérer les périphériques ».



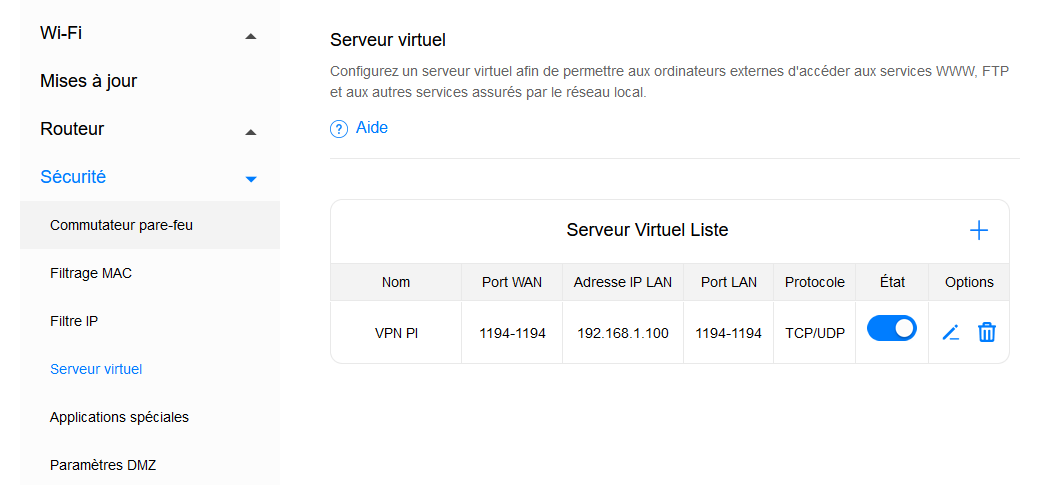
Nous avons laissé une part de DHCP notamment en cas de connexion d’appareils connectés (lampe, alarme, caméra IP…). Ensuite, il nous a suffi d’ajouter l’ensemble des IP. En voici un extrait :



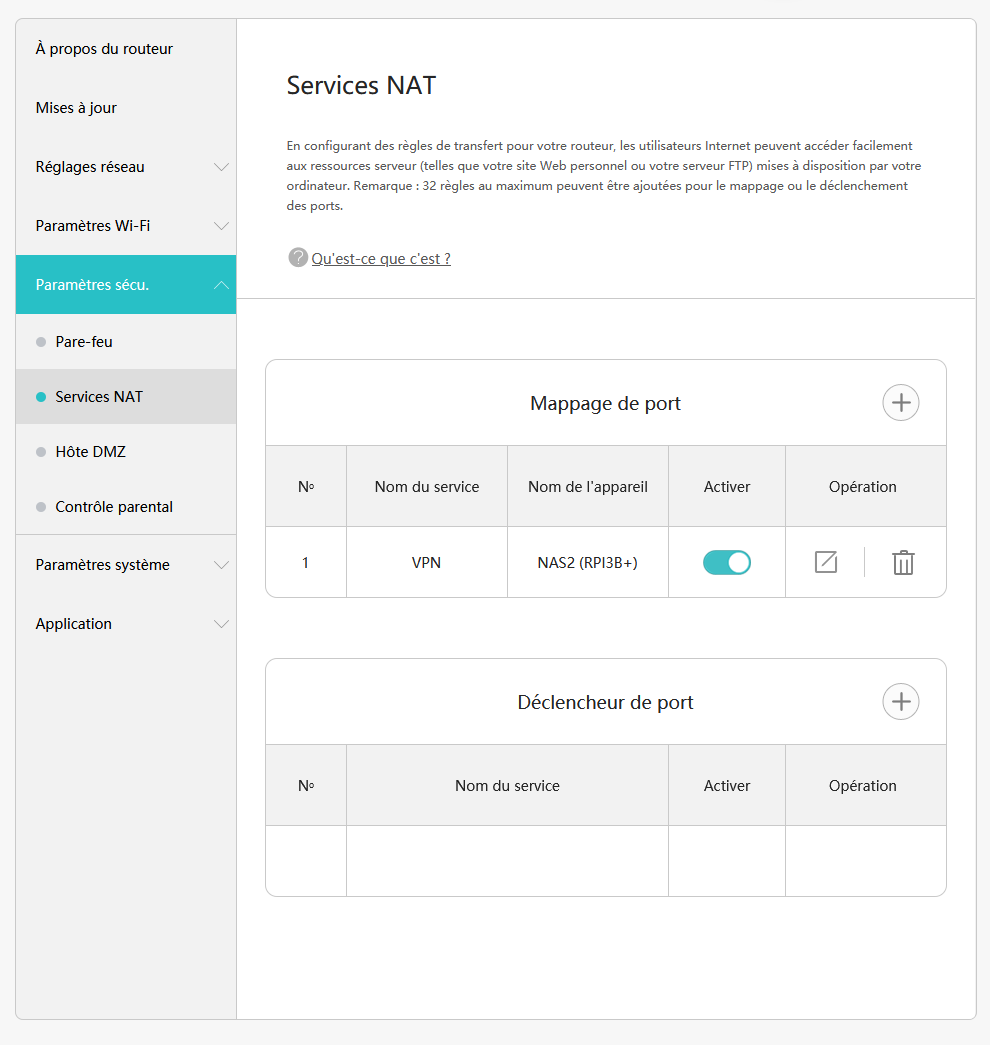
## Mappage de port

Lorsqu’un service a besoin qu’on ouvre des ports spécifiques, il faut faire deux manipulations : l’une sur la box, l’autre sur le routeur. Comme exemple, nous prenons le VPN qui a besoin du port 1194 pour fonctionner.

Côté 4GBox, il faut aller dans « serveur virtuel » comme montré ci-dessous :



Pour le routeur, cela se passe via « Services NAT » 🡪 Mappage de port :

\*

# Configuration et sécurité du Raspberry Pi

## Configuration

Nous utilisons le Raspberry Pi 3B+ pour faire certaines manipulations. Par défaut, celui-ci a comme login pi et le mot de passe raspberry.

Voici la procédure que nous avons fait pour configurer et sécuriser le PI correctement :

sudo apt-get update

sudo apt-get upgrade

sudo raspi-config

sudo apt-get install mc -y (gestionnaire de fichiers)

### Suivre les indications : ###

1 System Options 🡪 S3 Password (celui-ci change le mot de passe de l’utilisateur « pi » qui est par défaut « raspberry » pour quelque chose de bien plus sécurisé (16 caractères, lettres minuscules / majuscules, chiffres, caractères spéciaux))

1 System Options 🡪 S4 Hostname (changement en RaspberryLAN)

5 Localisation Options 🡪 L1 Locale (décocher « en\_GB.UTF-8 UTF-8 » et cocher à la place les 3 « fr\_FR », le principal étant « fr\_FR.UTF-8 UTF-8 », il faut sélectionner celle-ci dans l’étape d’après)

5 Localisation Options 🡪 L2 Timezone (sélectionner Europe puis Paris)

5 Localisation Options 🡪 L4 WLAN Country 🡪 FR France

6 Advanced Options 🡪 A1 Expand Filesystem

Enfin, lorsque vous quittez le menu de raspi-config il vous propose de reboot (ce qu’il faut accepter).

## Aspect sécurité

* Il faut vérifier régulièrement avec la commande « sudo service –status-all » s’il n’y a aucun service inutile présent afin d’éviter des failles potentielles.
* Pour le sudo, il est conseillé de le sécuriser avec un mot de passe ainsi que de supprimer l’utilisateur par défaut « pi » et créer d’autres utilisateurs, dans notre cas, pour des raisons de simplicité, nous ne l’avons pas fait.
* Nous avons également fait le choix de passer par l’authentification par mot de passe plutôt que par des clés pour le SSH toujours pour des raisons de simplicité. Cependant le port SSH n’est plus celui par défaut (22) mais 6720, pour le changer il faut aller dans la configuration SSH « sudo nano /etc/ssh/sshd\_config » et modifier la ligne « Port 22 » par « Port XXXX » (XXXX étant le nouveau port souhaité (attention à bien vérifier que celui-ci n’est pas utilisé par un autre service)). Il ne faut bien sûr pas oublier de redémarrer le service avec la commande « sudo service ssh restart ». Changer le port SSH est recommandé pour éviter les bots.
* Dans notre projet, nous avons fait le choix de ne pas installer de pare-feu en cas d’ajout de dernière minute d’un service quelconque, celui-ci sera bien sûr ajouté pour maximiser la sécurité, nous avons en attendant le pare-feu du routeur qui protège le Raspberry Pi. Mais nous avons en revanche fait le choix d’ajouter une sécurité à l’authentification en utilisant l’outil fail2ban :

Commande d’installation : sudo apt-get install fail2ban -y

Pour accéder à la configuration : sudo nano /etc/fail2ban/jail.conf

Pour appliquer la configuration : sudo service fail2ban restart

Pour accéder au log : sudo nano /var/log/fail2ban.log

La configuration par défaut est présente dans notre cas, le service bloquera une IP 10 minutes après 5 essais ratés (cela limite donc à 720 essais par jour).

# Sauvegarde Raspberry Pi

Un système de sauvegarde à chaud a été intégré au Pi en cas de problèmes avec la carte micro-SD, nous avons pour cela utilisé un script complet, nous utilisons comme matériel une clé USB de la même capacité que la micro-SD :

Il faut installer git pour récupérer le script : sudo apt install git -y

rsync est nécessaire pour le bon fonctionnement du script : sudo apt install rsync -y (si ce n’est pas déjà installé)

git clone https://github.com/billw2/rpi-clone.git

cd rpi-clone

sudo cp rpi-clone rpi-clone-setup /usr/local/sbin

Cette commande vous permet de voir le nom de la clé USB connecté : sudo fdisk -l

Faire une copie (la première sauvegarde mettra un peu de temps : sudo rpi-clone sda -v (il va demander une confirmation juste après)

Source : <https://raspberrytips.fr/sauvegarde-raspberry-pi/> ; <https://github.com/billw2/rpi-clone>

# VPN

Le problème qu’on avait dans un premier temps c’est que dans notre binôme, seulement Anthony a accès à l’infrastructure en physique, de plus, en prévision d’une éventuelle reprise des cours en présentiel, il nous fallait en priorité un VPN pour accéder au réseau local.

## Partie serveur et routeur / Box

Nous utilisons donc OpenVPN sur le raspberry pi. Par souci de simplicité, nous avons utilisé un script : <https://github.com/angristan/openvpn-install>. Celui-ci automatise l’ensemble des tâches.

Nous avons dû dans un premier temps récupérer le script et lui donner les bonnes permissions :

* curl -O <https://raw.githubusercontent.com/angristan/openvpn-install/master/openvpn-install.sh>
* chmod +x openvpn-install.sh

Après avoir fait la commande « sudo ./openvpn-install.sh », nous avons suivi les différentes indications :

IP address : XX.XX.XX.XX (adresse IP publique mise normalement automatiquement)

Do you want to enable IPv6 support (NAT) ? : n

What port do you want OpenVPN to listen to ? Port choice : 1 (qui correspond au port 1194)

What protocol do you want OpenVPN to use ? Protocol : 1 (qui correspond au protocole UDP)

What DNS resolvers do you want to use with the VPN ? DNS : 8 (qui correspond à OpenDNS)

Enable compression ? : n

Customize encryption settings ? n

Après quelques minutes nous pouvons créer le premier client :

Client name : XXXXXXXX (nom du client)

Do you want to protect the configuration file with a password ? Select an option : 1 (qui correspond à un client sans mot de passe)

Vu qu’il nous faut deux clients, nous avons du relancer le script et créer un nouveau client :

What do you want to do ? Select an option : 1 (qui correspond à l’ajout d’un nouvel utilisateur)

On applique ensuite la même chose que précédemment.

Nous avons donc maintenant nos deux clients (dans notre cas « anthony-hp » et « julien ».

Il faut ensuite redémarrer le PI et si on fait la commande « ifconfig » nous devrions, voir apparaitre « tun0 ».

Enfin, il ne faut pas oublier d’ouvrir le port 1194 sur la box et le routeur (donné en exemple dans la partie configuration BOX et routeur).

N.B. : Nous n’avons pas mis de DDNS en place donc lorsqu’il y aura un changement d’IP publique par le FAI, il faudra modifier l’IP dans la configuration côté client. (Lien pour voir son IP publique : <https://www.whatismyip.com/>)

## Partie client (Windows 10)

En premier, je récupère le fichier .ovpn via WinSCP. Après avoir installé le client sur ma machine, j’importe la configuration. Il suffit ensuite de se connecter et la connexion sera opérationnelle. Sur l’ordinateur nous avons une IP en 10.8.0.X (tun0 sur le PI est en 10.8.0.1) et nous pouvons accéder à l’ensemble du réseau.

# Virtualisation (Proxmox)

Nous avons fait le choix comme service d’utiliser proxmox principalement pour nous permettre de monter en compétence sur cette notion, il faut bien sûr préciser que celui-ci ne servira pas au particulier mais exclusivement à des entreprises.

## Configuration de base

Avant toute chose, il faut savoir que Proxmox ne sera pas installé sur une ou plusieurs machines physiques car ce service demande du matériel bien spécifique (serveur), un ordinateur est rarement compatible. Il y aura donc 3 nœuds (3 machines virtualisées) sur le « PC fixe (Anthony) » avec l’outil VirtualBox.

Nous devons obligatoirement être branchés sur le routeur en physique pour éviter d’avoir la sécurité du Wi-Fi qui n’est pas désactivable sur le routeur.  
Configuration de chaque host : Debian (64-bit), 1 processeur, 5 cœurs, 8192 Go de RAM, 200 Go de disque, VT-x/AMD-V et PAE/NX activés, mémoire vidéo 128 MB, son désactivé, réseau : accès par pont et mode promiscuité Allow All.

Ensuite nous démarrons les VM et nous les configurons :

* Dans un premier, on nous demande le disque à utiliser : Options 🡪 Filesystem : zfs (RAID0), le choix du disque ne doit pas être modifié.
* Ensuite, nous choisissons le pays, la langue et le fuseau horaire.
* Après nous rentrons les informations de l’utilisateur root.
* Enfin, nous mettons la configuration réseau :

Proxmox 0 : host0.lancontrol.local / 192.168.3.80/24 / Gateway et DNS : 192.168.3.1

Proxmox 1 : host1.lancontrol.local / 192.168.3.81/24 / Gateway et DNS : 192.168.3.1

Proxmox 2 : host2.lancontrol.local / 192.168.3.82/24 / Gateway et DNS : 192.168.3.1

Après l’installation, nous créons le cluster pour relier les 3 VMs.

Sur Proxmox 0 : pvecm create clusterLC

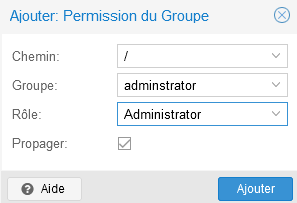
Sur Proxmox 1 et 2 : pvecm add 192.168.3.80 (et mettre le mot de passe de Proxmox 0)

Vérifier si tout est fonctionnel sur Proxmox 0 : pvecm nodes / pvecm status

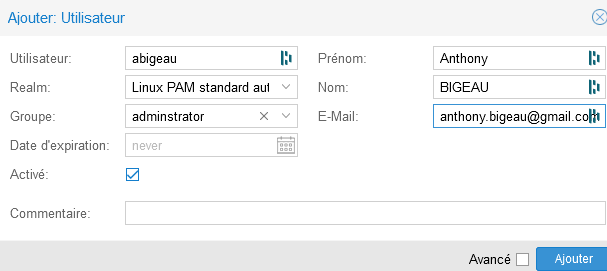
**Création d’un pool Ubuntu pour y mettre les VMs :** Datacenter (clusterLC) 🡪 Pools 🡪 Créer

**Création d’un groupe d’utilisateur, attribuer des permissions et créer des utilisateurs :**

Dans Datacenter (Permissions), il faut créer un groupe :



Et ensuite pour créer un utilisateur :



On peut plus ou moins changer les permissions dans le détail de chaque rôle, dans notre cas, nous n’en avons pas besoin. Cependant, l’utilisateur 2 « jgiovannangeli » a une date d’expiration fixée au 1er mai.

Également le compte root est désactivé (ce qui n’est pas le cas par défaut), on devra obligatoirement passer par les autres utilisateurs, il sera quand même utilisable pour la console Proxmox.

**Créer une VM :**

On aura pour des raisons de limitation technique uniquement deux VMs (1 processeur, 2 cœurs, 2048 Go de RAM, 32 Go de disque), l’une sur le host0, l’autre sur host1.

Dans un premier temps, il faut récupérer l’ISO, l’upload dans le disque local puis ISO Images et Upload du nœud où l’on souhaite avoir la VM.

Général : Attribuer un nœud et un Pool de ressource.

OS : Mettre la bonne ISO.

Réseau : Mettre en modèle Intel E1000.

On peut une fois créer la transformer en Template dans « Plus » pour en créer rapidement sur le même modèle.

Il faut avant de démarrer la VM désactiver la virtualisation matérielle KVM dans les options.

Une fois la VM correctement installée, il ne faut pas oublier de retirer l’ISO dans matériel (Lecteur CD/DVD).

N.B : Les VMs étant très lente dans notre cas, nous avons installé openssh-server pour s’y connecter en SSH directement sur la machine hôte.

## Haute disponibilité

Pour rappel, la HA permet de mettre en place des actions et des paramètres techniques pour que l’infrastructure permette de toujours répondre à la requête d’un utilisateur.

Datacenter 🡪 HA (Groupes) 🡪 Créer, on choisit comme nom dans ID, on sélectionne les nœuds qu’on utilise et on met une priorité si on le souhaite.

Pour ajouter une machine virtuelle : Sélectionner l’une d’elles 🡪 Plus 🡪 Gérer la HA et faire la configuration qu’on souhaite.

Pour tester son bon fonctionnement, il suffit d’éteindre un nœud et voir.

## Réplication

Définition : Cela permet de copier du contenu dans un autre endroit pour le sécuriser en cas de problèmes.

Configuration : Sélectionner une VM et aller dans l’onglet Réplication et Ajouter, configurer celle-ci comme vous le souhaitez (nous l’avons programmé chaque jour sur l’ensemble des nœuds à 17h30 juste avant la fin de la journée)).

# Configuration et sécurisation du NAS

Le modèle du NAS utilisé est le Synology DS218play. Il a la particularité d’avoir un transcodage 4K UHD pour les vidéos. Il a comme OS : DSM (disk station manager) basé sur FreeBSD. Toutes les installations et les configurations peuvent se faire via son interface graphique. Il a deux baies, dans notre cas nous avons à notre disposition deux HDD de 2 To chacun de marque Western Digital Red.

## Configuration

Lors du branchement du NAS, on peut directement se connecter à l’interface web pour faire la configuration de base avec un utilisateur principal (qui est aussi compte administrateur). On a la possibilité de configurer QuickConnect qui est un service nous permettant d’accéder au NAS via l’extérieur sans transmission de port (avec un lien comme <http://quickconnect.to>). Une légère personnalisation est faite partie Thème dans le panneau de configuration.

Directement, on ajoute un utilisateur « Julien » ayant des accès administrateur mais expirant le 1/05/2021. L’ajout peut se faire dans panneau de configuration 🡪 Utilisateur et si on sélectionne l’onglet Avancé, on peut appliquer divers paramètres de sécurité de mot de passe (notamment la double authentification).

Nous avons également mis en place une planification d’alimentation (partie Matériel et alimentation 🡪 onglet Planif. Alim.), en semaine le NAS est allumé de 9h à 21h30 et en week-end de 10h à 21h30.

## Sécurité

Comme dit dans la partie configuration, nous avons fait le choix d’activer la double authentification pour le groupe administrateur et de renforcer les règles pour les mots de passe (majuscule / minuscule, chiffres, caractères spéciaux, longueur minimale…).

Nous avons également modifié le port DSM par défaut dans Réseau 🡪 Paramètres de DSM et activer la redirection HTTPS.  
Dans la partie sécurité, nous avons activé le pare-feu (onglet Pare-feu), activer la protection DoS (onglet Protection), activer le blocage automatique d’IP au bout d’un certains nombres d’essais (onglet Compte) et créer un certificat Let’s Encrypt (onglet Certificat).

Un système d’alerte par mail est mis en place via la partie Notification. Cela est très pratique notamment en cas de coupure d’électricité.

Dans la partie Terminal & SNMP, nous avons changé le port SSH par défaut.

En cas de problèmes physiques avec le NAS et en particulier avec le disque dur, nous avons fait le choix de configurer les deux baies en RAID1 qui utilise la technique du mirroring (miroir). Nous devons prendre en compte que nous devons diviser par deux la capacité de stockage. Mais pour nous 2 To est plus que suffisant.

Pour avoir du RAID il faut aller dans le Gestionnaire de stockage puis dans la partie Groupe de stockage.

Enfin, le plus important c’est l’outil mis à disposition par Synology : Conseiller de sécurité qui permet de faire une analyse complète du NAS au niveau sécurité.

Voici la dernière analyse faite au moment de la rédaction de ce document :



N.B. : Si on sélectionne l’une des tâches, nous pouvons avoir le détail pour améliorer l’état d’une potentielle faille.

# Monitoring

Nous avons fait en sorte d’avoir un point de vue graphique sur tout cela en faisant du monitoring. Prometheus nous permet d’enregistrer et d’analyser des données métriques qui, grâce à Grafana, sont affichées de façon plus esthétique. Ces deux derniers sont installés sur le Raspberry Pi.

## Installation

Nous commençons par installer Prometheus à l’aide de la commande « sudo apt-get install promethus -y ».

Une fois installé, nous éditons le fichier de configuration se trouvant dans : /etc/prometheus/prometheus.yml

Voici notre configuration :

# Sample config for Prometheus.

static\_configs:

- targets: ['localhost:9090']

- job\_name: node

# If prometheus-node-exporter is installed, grab stats about the local

# machine by default.

static\_configs:

- targets: ['192.168.3.3:9100','192.168.3.6:9100']

- job\_name: wmi\_exporter

static\_configs:

- targets: ['192.168.3.4:9182', '192.168.3.6:9182', '192.168.3.5:9182','192.168.3.7']

On peut voir dans cette configuration que nous avons un job\_name pour Node Exporter (« node ») et un autre pour WMI Exporter. Node Exporter va nous servir à relever des métriques sur les ordinateurs Linux et WMI Exporter pour les ordinateurs Windows.

Une fois édité, nous rechargeons la configuration avec la commande « curl -X POST http://localhost:9090/-/reload ».

Puis nous pouvons désormais accéder à « localhost:9090 » sur la Raspberry Pi ou alors si on souhaite y accéder via un autre appareil sur le réseau il faut remplacer « localhost » par l’IP locale du Raspberry Pi.

## Envoi de données sur le promethus

* Linux :

Création d’un utilisateur « node\_exporter » avec un « /bin/false » de manière à éviter qu’il soit utilisé à d’autres fins : sudo useradd -rs /bin/false node\_exporter

wget https://github.com/prometheus/node\_exporter/releases/download/v0.18.1/node\_exporter-0.18.1.linux-amd64.tar.gz

tar -xvzf node\_exporter-0.18.1.linux-amd64.tar.gz

sudo mv node\_exporter-0.18.1.linux-amd64/node\_exporter /usr/local/bin/

sudo chown node\_exporter:node\_exporter /usr/local/bin/node\_exporter

sudo nano /etc/systemd/system/node\_exporter.service

Et on y colle ceci :

|  |
| --- |
| [Unit]  Description=Node Exporter  After=network-online.target  [Service]  User=node\_exporter  Group=node\_exporter  Type=simple  ExecStart=/usr/local/bin/node\_exporter  [Install]  WantedBy=multi-user.target |

sudo systemctl daemon-reload

sudo systemctl enable node\_exporter.service

sudo systemctl start node\_exporter.service

sudo systemctl status node\_exporter.service

* Windows :

Installer le .exe ou le .msi et le lancer (<https://github.com/prometheus-community/windows_exporter/releases>)

À partir de là, nous avons pu observer que Prometheus recevait bien les métriques de chaque appareil. Nous avons ensuite installé Grafana pour mettre toutes ces données en forme avec les commandes suivantes :

sudo apt install apt-transport-https curl

curl https://bintray.com/user/downloadSubjectPublicKey?username=bintray | sudo apt-key add -

echo "deb https://dl.bintray.com/fg2it/deb stretch main" | sudo tee -a /etc/apt/sources.list.d/grafana.list

sudo apt-get update

sudo apt-get install grafana

Démarrage du service : sudo /etc/init.d/grafana start

L’accès à Grafana : localhost:3000 (ou IP du Raspberry Pi depuis un navigateur d’un ordinateur du réseau)

Par la suite, nous avons importé sur Grafana des dashboards déjà existants pour mettre en place l’aspect esthétique. Nous avons par la suite ajouté un système de notification qui est envoyé via webhook Discord en cas d’alerte.

# Conclusion

Pour conclure, on peut dire que grâce à ce projet, nous avons pu monter en compétences sur divers aspects et découvrir de nouvelles technologies.

Nous souhaitons aller plus loin par la suite pour faire évoluer le projet :

* Continuer à mettre le service web sur le NAS (il est créé mais inaccessible à l’extérieur et sans nom de domaine)
* Créer une sauvegarde du NAS vers le Raspberry Pi et ajouter des caméras IP dans la partie domotique (n’a pas pu être fait par manque de budget)
* Mettre en place un DDNS pour le VPN
* Régler le problème de monitoring du NAS
* Mettre en place de la détection d’intrusion sur le réseau (à l’aide par exemple de Suricata sur le Raspberry Pi)