

# SAE 301

Mettre en œuvre un système de transmission

PAQUELET - POUT



Monsieur Millet

Du 13/10/2023 au 20/10/2023



# Présentation

Une SAE, ou Situation d'Apprentissage et d'Évaluation, baptisée SAE 301 : "Mettre en œuvre un système de transmission", a débuté le vendredi 13 octobre 2023 et s'est déroulée jusqu'au vendredi 20 octobre 2023 au sein du département Réseaux et Télécommunications du BUT de Montbéliard.

Au cours de ce projet, nous avons déployé plusieurs services et configuré chacun d'entre eux, notamment :

- L'installation d'un serveur web Nginx, ainsi que d'applications pour la transmission vidéo via un serveur vidéo DASH, la transmission sécurisée via des serveurs VPN Wireguard et OpenVPN, et la transmission multimédia via un serveur NAS OpenMediaVault (OMV).

Pour mieux comprendre et exploiter les transmissions réseau, nous avons utilisé divers logiciels. Par exemple, nous avons employé iperf3, un outil informatique permettant de mesurer diverses variables d'une connexion réseau IP, telles que le débit. De plus, Munin, un logiciel open source de surveillance système et réseau, a été utilisé pour superviser les performances du réseau.

Le matériel utilisé au cours de cette SAE comprend :

- Un Raspberry Pi 4, doté de son alimentation, d'une carte micro-SD et d'un câble Ethernet pour la mise en réseau.
- Un PC de l'IUT équipé de VirtualBox, permettant d'exécuter des machines virtuelles.
- L'un de nos propres PC, utilisé en remplacement d'un ordinateur de l'IUT qui ne prenait pas en charge les machines virtuelles faisant l'hôte de l'autre VM nécessaire.

Les objectifs de cette SAE étaient multiples. Ils consistaient à nous initier à la mise en place de services réseau en lien avec les transmissions, à nous sensibiliser à la qualité de ces transmissions, tout en nous encourageant à développer notre autonomie.

Dans ce rapport, nous présenterons en détail le projet réalisé, ainsi que nos interprétations et remarques concernant notre travail.

# Sommaire

Partie 1 : Plateforme d'Apprentissage : Raspberry PI, machine virtuelle.....

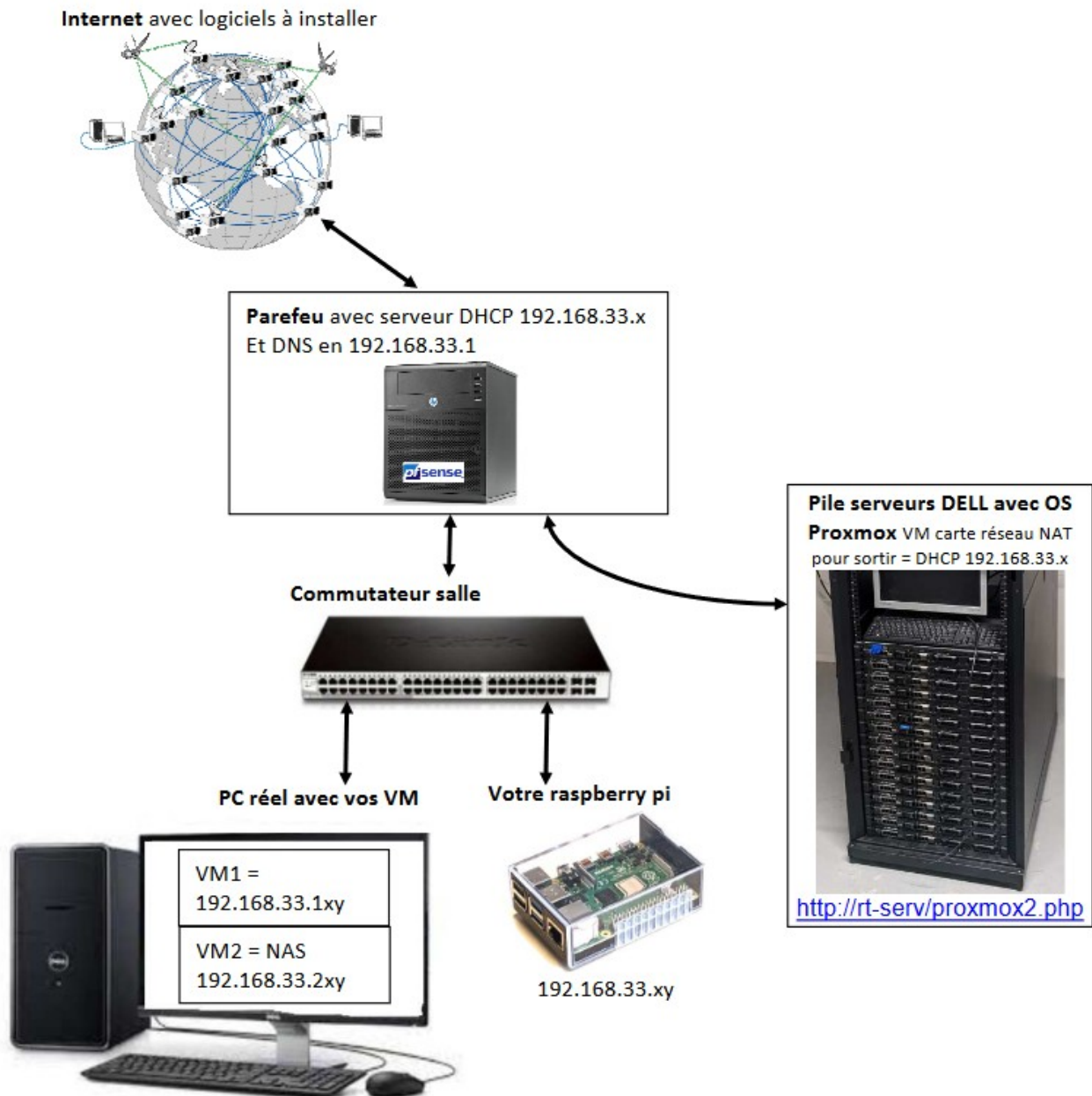
Partie 2 : Débit d'une source d'Information.....

Partie 3 : Système d'analyse de débit : Munin et Iperf3.....

Partie 4 : Mise en place d'Application : serveur VPN, DASH, NAS.....

## Partie 1 : Plateforme d'apprentissage: RPI, machine virtuelle

Dans cette SAE, nous devons obtenir le réseau suivant :



## *Partie 1 : Plateforme d'apprentissage:RPI, machine virtuelle*

En premier lieu, nous avons initié le processus en gravant la carte micro-SD du Raspberry Pi avec le système d'exploitation RaspbianOS. Pour accomplir cette tâche, nous avons utilisé un logiciel disponible sur un ordinateur de la salle 205, nommé "Raspberry Pi imager". Il était essentiel de configurer dans les paramètres l'option SSH avec un utilisateur et un mot de passe. En effet, afin de pouvoir se connecter via VNC, il fallait autoriser les connexions à distance. De plus, n'ayant pas de câble HDMI pour se connecter à l'écran, il a fallu utiliser SSH afin d'installer sur le Raspberry des paquets depuis un ordinateur distant.

La prochaine étape consistait à configurer le réseau en utilisant un serveur DHCP, ce qui nous a permis d'obtenir l'adresse IP du Raspberry Pi. Cette adresse IP a été utilisée pour configurer l'interface réseau du Raspberry Pi via SSH. Nous avons choisi l'adresse IP 192.168.33.15 et avons également spécifié le routeur par défaut ainsi que les serveurs DNS.

Une étape impérative était d'activer VNC via l'outil raspi-config, car cela était essentiel pour accéder au portail captif et pour l'installation de paquets nécessitant une connexion à distance avec une interface graphique.

Nous avons également, par mesure de sécurité, changé le mot de passe de l'utilisateur pi, afin d'éviter de possibles intentions malveillantes.

Nous avons ensuite créé et configuré une Machine Virtuelle telle que celle-ci possède :

- 4 processeurs
- 4 Go de RAM,
- 1 disque dur de 40 Go.

Nous utilisons 4 processeurs pour faciliter l'encodage vidéo lorsque nous aurons besoin du serveur vidéo à débit adaptatif. En effet, si on en mettait pas assez, l'encodage serait très long et nous aurait fait perdre du temps précieux sur ce projet.

Il fallait installer la machine virtuelle sous le système d'exploitation Ubuntu MATE, qui est très intéressant car la version MATE est maintenue en mise à jour et enrichie en fonctionnalités sans perdre son avantage d'avoir une faible consommation de ressource mémoire et CPU.

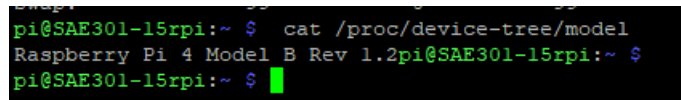
L'adresse IP qui nous était assignée pour notre machine virtuelle était 192.168.33.115. Le nom de cette machine était SAE301-15vm et l'identifiant était groupe15 avec le mot de passe Kirimi15. Il nous a été recommandé d'installer l'utilitaire Netspeed sur notre VM. Netspeed permettant d'analyser le débit au niveau d'une interface réseau, et de le retranscrire sous forme de graphiques.

## Partie 1 : Plateforme d'apprentissage:RPI, machine virtuelle

Nous avons relevé les composants du Raspberry pi et de la VM Dans cette partie, les réponses du jalon 2 sont exposées.

A l'aide de la commande `cat /proc/device-tree/model`, nous pouvons voir quel est le type de raspberry, son modèle :

Type de Raspberry : Raspberry Pi 4 model B.

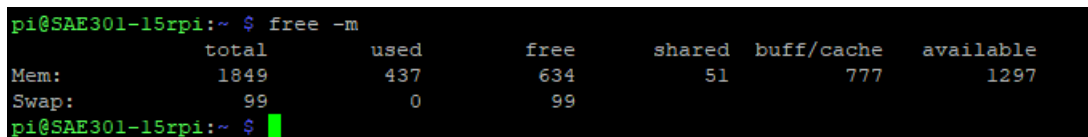


```
pi@SAE301-15rpi:~$ cat /proc/device-tree/model
Raspberry Pi 4 Model B Rev 1.2pi@SAE301-15rpi:~$
```

Figure 2 : Capture d'écran montrant le modèle du Raspberry.

### Caractéristique de la mémoire.

Afin de connaître la capacité de la mémoire du Raspberry Pi, nous utilisons la commande `free -m` qui montre différents des statistiques telles que «nombre d'octet utilisé, total ».



```
pi@SAE301-15rpi:~$ free -m
              total        used        free      shared  buff/cache   available
Mem:           1849          437          634           51           777          1297
Swap:           99             0            99
```

Figure 3 : capture d'écran après la commande `free -m`.

Sur cette capture, nous pouvons voir différentes entrées : total, used, free, shared, buff/cache, available. Et différents type de mémoire : swap, mem.

Pour les types de mémoire, ce Raspberry en possède deux :

- Swap : espace de mémoire virtuelle utilisé par le système lorsque la RAM est pleine. Cela inclut généralement une partition de swap sur le disque dur.
- Mem : mémoire physique (RAM) disponible pour le système.

Voici la description de chaque statistiques pour les type de mémoire :

*total*: La mémoire totale du système en mégaoctets.

*used*: La quantité de mémoire actuellement utilisée par le système en mégaoctets.

*free*: La quantité de mémoire libre en mégaoctets.

*shared*: La quantité de mémoire partagée entre différents processus en mégaoctets.

*buff/cache*: La quantité de mémoire utilisée pour les tampons et les caches en mégaoctets.



*available*: La quantité de mémoire disponible pour les nouveaux processus en mégaoctets.

### Processeur

Nous avons ensuite effectué la commande `cat /proc/cpuinfo` afin d'obtenir des informations supplémentaires sur notre CPU de notre RPI tel que le nom du modèle, qui ici, est : Raspberry Pi 4 Model B Rev 1.2

```
pi@SAE301-15rpi:~ $ cat /proc/cpuinfo

processor       : 3
BogoMIPS      : 108.00
Features       : fp asimd evtstrm crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant    : 0x0
CPU part       : 0xd08
CPU revision   : 3

Hardware       : BCM2835
Revision       : b03112
Serial         : 10000000e9284baf
Model          : Raspberry Pi 4 Model B Rev 1.2
pi@SAE301-15rpi:/dev $
```

Figure 4 : Capture d'écran après avoir fait la commande `cat /proc/cpuinfo`

### Caractéristique de la carte réseau

Nous avons ensuite cherché les caractéristiques de la carte réseau avec la commande `lshw -C network` et nous avons obtenus les informations suivantes :

```
pi@SAE301-15rpi:~ $ sudo lshw -C network
sudo: impossible de résoudre l'hôte SAE301-15rpi: Nom ou service inconnu

*-network:0
  description: Ethernet interface
  identifiant matériel: 1
  nom logique: eth0
  numéro de série: dc:a6:32:ad:56:0f
  taille: 100Mbit/s
  capacité: 1Gbit/s
  fonctionnalités: ethernet physical tp mii 10bt 10bt-fd 100bt 100bt-fd 1000bt 1000bt-fd autonegotiation
  configuration: autonegotiation=on broadcast=yes driver=bcmgenet driverversion=5.15.61-v8+ duplex=full ip=192.168.33.15 link=yes multicast=yes port=twisted pair speed=100Mbit/s
*-network:1 DÉSACTIVÉ
  description: Interface réseau sans fil
  identifiant matériel: 2
  nom logique: wlan0
  numéro de série: dc:a6:32:ad:56:10
  fonctionnalités: ethernet physical wireless
  configuration: broadcast=yes driver=brcmfmac driverversion=7.45.241 firmware=01-703fd60 multicast=yes wireless=IEEE 802.11
pi@SAE301-15rpi:~ $
```

*Figure 5 : caractéristique de la carte réseau*

Nous pouvons voir que l'interface eth0 est activée et a les caractéristiques suivantes :

- numéro de série : l'adresse MAC de cette interface,
- Sa taille : 100Mbit/s
- Sa Capacité : 1Gbit/s
- Sa configuration : auto negotiation, broadcast, version des drivers, duplex, taille.

La différence entre sa taille et sa capacité est que sa taille correspond à sa vitesse actuelle et la capacité à la capacité du réseau.

### **Analyse USB**

```
pi@SAE301-15rpi:~ $ sudo lsusb -t
sudo: impossible de résoudre l'hoste SAE301-15rpi: Nom ou service inconnu
/: Bus 02.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/4p, 5000M
/: Bus 01.Port 1: Dev 1, Class=root_hub, Driver=xhci_hcd/1p, 480M
   |__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
pi@SAE301-15rpi:~ $
```

*Figure 7 : Caractéristiques des bus USB avec la commande lsusb -t.*

Nous pouvons observer sur la figure 7 qu'il y a 2 bus USB : bus 02, et le Bus 01. Le bus 02 est du type USB3. Effectivement, son débit maximum est de 5000Mbit/s. Le bus 01 est du type USB2. Il possède un débit maximum de 480Mbit/s.

La carte réseau Ethernet détermine si elle fonctionne en Ethernet, Fast Ethernet ou Gigabit Ethernet par le biais du processus de négociation de vitesse avec le commutateur ou le routeur auquel elle est connectée. Le processus de négociation de vitesse est le suivant :

1. **Connexion initiale** : Lorsqu'une carte réseau Ethernet est connectée à un commutateur ou à un routeur, la connexion démarre généralement à la vitesse la plus basse prise en charge, c'est-à-dire Ethernet (10 Mbps). C'est la vitesse par défaut à laquelle la négociation commence.
2. **Annonce des capacités** : Les deux périphériques (la carte réseau et le commutateur/routeur) échangent des signaux pour indiquer leurs capacités. Cela comprend la vitesse (10 Mbps, 100 Mbps, 1 Gbps), le mode duplex (half-duplex ou full-duplex), et d'autres caractéristiques.
3. **Comparaison des capacités** : Chaque périphérique compare les informations d'annonce de l'autre pour déterminer la vitesse la plus élevée à laquelle ils peuvent tous deux communiquer. Par exemple, si les deux périphériques sont compatibles avec Gigabit Ethernet (1 Gbps), c'est la vitesse qui sera retenue.



4. **Négociation de la vitesse** : Les périphériques négocient la vitesse optimale. Si l'un des périphériques ne prend pas en charge la vitesse maximale annoncée par l'autre, il peut y avoir une rétrogradation automatique à une vitesse inférieure à laquelle les deux périphériques peuvent communiquer. Par exemple, si l'un prend en charge 1 Gbps et l'autre seulement 100 Mbps, la négociation aboutira à 100 Mbps.
5. **Réglage de la vitesse** : Une fois la négociation terminée, les périphériques ajustent leur vitesse de fonctionnement en conséquence. La vitesse retenue est généralement indiquée dans les informations de configuration de la carte réseau.

Le processus de négociation de vitesse garantit que les périphériques Ethernet communiquent à la vitesse maximale à laquelle ils sont mutuellement compatibles, tout en prenant en compte leurs capacités respectives. Cela permet une communication réseau optimale en fonction de l'infrastructure matérielles.

```
pi@SAE301-15rpi:~ $ sudo ethtool eth0
sudo: impossible de résoudre l'host SAE301-15rpi: Nom ou service inconnu
Settings for eth0:
    Supported ports: [ TP      MII ]
    Supported link modes:   10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Half 1000baseT/Full
    Supported pause frame use: Symmetric Receive-only
    Supports auto-negotiation: Yes
    Supported FEC modes: Not reported
    Advertised link modes:  10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Half 1000baseT/Full
    Advertised pause frame use: Symmetric Receive-only
    Advertised auto-negotiation: Yes
    Advertised FEC modes: Not reported
    Link partner advertised link modes:  10baseT/Half 10baseT/Full
                                         100baseT/Half 100baseT/Full
    Link partner advertised pause frame use: No
    Link partner advertised auto-negotiation: Yes
    Link partner advertised FEC modes: Not reported
    Speed: 100Mb/s
    Duplex: Full
    Auto-negotiation: on
    master-slave cfg: preferred slave
    master-slave status: slave
    Port: Twisted Pair
    PHYAD: 1
    Transceiver: external
    MDI-X: Unknown
    Supports Wake-on: d
    Wake-on: d
    Current message level: 0x00000007 (7)
                               drv probe link
    Link detected: yes
pi@SAE301-15rpi:~ $
```

Figure 8 : Caractéristique de l'interface eth0

Le débit dans les 3 cas est le suivant : pour une carte Ethernet simple : 10Mbps. Pour une carte FastEthernet : 100 Mbps. Pour une carte GigaEthernet : 1Gbps.

La vitesse actuelle du port ethernet, d'après la figure 8, est de 100Mbps, soit du FastEthernet.

### **Carte SD.**

Afin de tester notre disque dur, nous avons réalisé différents tests de lecture et d'écritures. Afin de réaliser le test de lecture, nous avons utilisé la commande `sudo hdparm -Tt /dev/mmcblk0p1` et nous obtenons, avec 3 tests, 3 vitesses de lectures différentes, avec en moyenne une différence de 5 Mbps entre ces test. Notre vitesse de lecture est donc approximativement de 930 Mbit/s.

```
pi@SAE301-15rpi:~ $ sudo hdparm -Tt /dev/mmcblk0p1

/dev/mmcblk0p1:
Timing cached reads:   1864 MB in  2.00 seconds = 933.36 MB/sec
Timing buffered disk reads: 132 MB in  3.03 seconds =  43.56 MB/sec
pi@SAE301-15rpi:~ $
```

*Figure 9 : test de vitesse de lectures*

Nous avons ensuite effectué un test d'écriture de notre carte SD avec la commande `dd if=/dev/zero of=/tmp/sample.bin` et nous obtenons le résultat suivant : notre carte SD a une vitesse d'écriture de 21,9Mbit/s

```
pi@SAE301-15rpi:~ $ dd if=/dev/zero of=/tmp/sample.bin

^C2159468+0=enregistrement lus
2159467+0=enregistrement =crits
1105647104=octets (1,1 GB, 1,0 GiB) copiés, 50,7784 s, 21,8 MB/s
```

*Figure 10 : test de vitesse d'écriture*

Nous avons ensuite comparer ces valeurs aux valeurs d'autres disque dur dont nous représentons les résultats dans le tableau suivant :

	Vitesse d'écriture	Vitesse de Lecture (en Mb/s)
Notre carte SD	21,8 Mb/s	930 Mb/s
disque dur HDD 7200 tours/minute (rpm )	43,6 Mo/s	43,4 Mo/s
disque dur SSD sata	600 Mo/s	600 Mo/s
disque dur M2 sata	5000 Mo/s	5000 Mo/s
disque dur pcie = nvme	8500 Mo/s	10500 Mo/s
clé USB3 sur laquelle on boote.	5 Gbits/s	5 Gbits/s

Nous avons ensuite effectué des tests de connexions entre le RPI et la VM, dans différents mode. Voici ce qui se passe une capture d'écran entre la connexion effectué par le client sur le serveur :

```
groupe15@SAE301-15vm:~$ iperf3 -c 192.168.33.15
Connecting to host 192.168.33.15, port 5201
[ 5] local 192.168.33.115 port 54570 connected to 192.168.33.15 port 5201
[ ID] Interval      Transfer    Bitrate      Retr  Cwnd
[ 5]  0.00-1.00  sec   14.6 MBytes  122 Mbits/sec    0   604 KBytes
[ 5]  1.00-2.00  sec   11.2 MBytes  94.3 Mbits/sec    0   1.15 MBytes
[ 5]  2.00-3.00  sec   11.2 MBytes  94.4 Mbits/sec    0   1.71 MBytes
[ 5]  3.00-4.00  sec   11.2 MBytes  94.4 Mbits/sec    0   2.27 MBytes
[ 5]  4.00-5.00  sec   11.2 MBytes  94.4 Mbits/sec    0   2.84 MBytes
[ 5]  5.00-6.00  sec   11.2 MBytes  94.4 Mbits/sec    0   3.13 MBytes
[ 5]  6.00-7.00  sec   11.2 MBytes  94.3 Mbits/sec    0   3.13 MBytes
[ 5]  7.00-8.00  sec   11.2 MBytes  94.4 Mbits/sec    0   3.13 MBytes
[ 5]  8.00-9.00  sec   11.2 MBytes  94.3 Mbits/sec    0   3.13 MBytes
[ 5]  9.00-10.00 sec   11.2 MBytes  94.4 Mbits/sec    0   3.13 MBytes
- - - - -
[ ID] Interval      Transfer    Bitrate      Retr
[ 5]  0.00-10.00  sec   116 MBytes  97.1 Mbits/sec    0             sender
[ 5]  0.00-10.33  sec   115 MBytes  93.6 Mbits/sec             receiver

iperf Done.
```

Figure 11:connexion au serveur iperf3 depuis le client (VM). Une fois la connexion réussie, il envoi des paquets afin de tester les performances réseaux (débits, latence).

```

Server listening on 5201
-----
Accepted connection from 192.168.33.115, port 54556
[ 5] local 192.168.33.15 port 5201 connected to 192.168.33.115 port 54570
[ ID] Interval      Transfer    Bitrate
[ 5]  0.00-1.00    sec  10.6 MBytes  89.1 Mbits/sec
[ 5]  1.00-2.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  2.00-3.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  3.00-4.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  4.00-5.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  5.00-6.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  6.00-7.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  7.00-8.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  8.00-9.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  9.00-10.00   sec  11.2 MBytes  94.1 Mbits/sec
[ 5] 10.00-10.33   sec   3.67 MBytes  94.1 Mbits/sec
-----
[ ID] Interval      Transfer    Bitrate
[ 5]  0.00-10.33   sec  115 MBytes  93.6 Mbits/sec
-----
receiver

```

Figure 12 : Serveur acceptant la connexion du client (VM) en recevant les paquets envoyé par le client.

Nous avons ensuite refait ces tests, mais en mode reverse :

```

groupe15@SAE301-15vm:~$ iperf3 -c 192.168.33.15 -R
Connecting to host 192.168.33.15, port 5201
Reverse mode, remote host 192.168.33.15 is sending
[ 5] local 192.168.33.115 port 39464 connected to 192.168.33.15 port 5201
[ ID] Interval      Transfer    Bitrate
[ 5]  0.00-1.00    sec  11.3 MBytes  94.6 Mbits/sec
[ 5]  1.00-2.00    sec  11.2 MBytes  94.1 Mbits/sec
[ 5]  2.00-3.00    sec  11.2 MBytes  94.2 Mbits/sec
[ 5]  3.00-4.00    sec  11.2 MBytes  94.2 Mbits/sec
[ 5]  4.00-5.00    sec  11.2 MBytes  94.2 Mbits/sec
[ 5]  5.00-6.00    sec  11.2 MBytes  94.2 Mbits/sec
[ 5]  6.00-7.00    sec  11.2 MBytes  94.2 Mbits/sec
[ 5]  7.00-8.00    sec  11.2 MBytes  94.2 Mbits/sec
[ 5]  8.00-9.00    sec  11.2 MBytes  94.2 Mbits/sec
[ 5]  9.00-10.00   sec  11.2 MBytes  94.2 Mbits/sec
-----
[ ID] Interval      Transfer    Bitrate    Retr
[ 5]  0.00-10.07   sec  114 MBytes  94.7 Mbits/sec    0
[ 5]  0.00-10.00   sec  112 MBytes  94.2 Mbits/sec
-----
iperf Done.

```

Figure 13 : Client en mode bidirectionnel

```

Server listening on 5201
-----
Accepted connection from 192.168.33.115, port 39460
[ 5] local 192.168.33.15 port 5201 connected to 192.168.33.115 port 39464
[ ID] Interval      Transfer    Bitrate    Retr  Cwnd
[ 5]  0.00-1.00    sec  11.1 MBytes  93.4 Mbits/sec    0   106 KBytes
[ 5]  1.00-2.00    sec  11.2 MBytes  93.8 Mbits/sec    0   106 KBytes
[ 5]  2.00-3.00    sec  11.2 MBytes  93.8 Mbits/sec    0   112 KBytes
[ 5]  3.00-4.00    sec  11.2 MBytes  93.8 Mbits/sec    0   112 KBytes
[ 5]  4.00-5.00    sec  11.2 MBytes  93.8 Mbits/sec    0   112 KBytes
[ 5]  5.00-6.00    sec  11.2 MBytes  93.8 Mbits/sec    0   112 KBytes
[ 5]  6.00-7.00    sec  11.4 MBytes  95.9 Mbits/sec    0   112 KBytes
[ 5]  7.00-8.00    sec  12.0 MBytes  101 Mbits/sec    0   256 KBytes
[ 5]  8.00-9.00    sec  11.3 MBytes  94.9 Mbits/sec    0   304 KBytes
[ 5]  9.00-10.00   sec  11.2 MBytes  93.8 Mbits/sec    0   354 KBytes
[ 5] 10.00-10.07   sec   764 KBytes  84.1 Mbits/sec    0   354 KBytes
-----
[ ID] Interval      Transfer    Bitrate    Retr
[ 5]  0.00-10.07   sec  114 MBytes  94.7 Mbits/sec    0
-----
sender
Server listening on 5201

```

Figure 14 : Serveur qui envoie des paquets au client

Nous remarquons que le client peut également fonctionner dans l'autre sens, c'est à dire, que c'est le client qui reçoit et envoie des paquets au serveur pour évaluer les performances réseau dans les deux sens.

Nous avons ensuite effectué ce test avec un serveur externe. Nous avons pris le serveur externe : ping.online.net sur le port 5200. Nous avons également réaliser la commande tracer pour voir les étapes de ce transit de paquets.

```
pi@S301-15rpi:~$ iperf3 -s ping.online.net -p 5200
Connecting to host ping.online.net, port 5200
[ 5] local 192.168.33.15 port 35260 connected to 62.210.18.40 port 5200
[ ID] Interval      Transfer    Bitrate    Retr    Cwnd
[ 5] 0.00-1.00 sec  12.7 MBytes 107 Mbits/sec  0      356 KBytes
[ 5] 1.00-2.00 sec  11.1 MBytes 93.3 Mbits/sec  0      356 KBytes
[ 5] 2.00-3.00 sec  11.2 MBytes 93.8 Mbits/sec  0      356 KBytes
[ 5] 3.00-4.00 sec  11.2 MBytes 93.8 Mbits/sec  0      356 KBytes
[ 5] 4.00-5.00 sec  11.2 MBytes 93.8 Mbits/sec  0      375 KBytes
[ 5] 5.00-6.00 sec  11.2 MBytes 93.8 Mbits/sec  0      375 KBytes
[ 5] 6.00-7.00 sec  11.2 MBytes 93.8 Mbits/sec  0      375 KBytes
[ 5] 7.00-8.00 sec  11.7 MBytes 98.3 Mbits/sec  0      423 KBytes
[ 5] 8.00-9.00 sec  10.6 MBytes 89.3 Mbits/sec  0      465 KBytes
[ 5] 9.00-10.00 sec 10.6 MBytes 89.3 Mbits/sec  0      465 KBytes
- - - - -
[ ID] Interval      Transfer    Bitrate    Retr    sender
[ 5] 0.00-10.00 sec 113 MBytes 95.1 Mbits/sec  0
[ 5] 0.00-10.00 sec 112 MBytes 93.8 Mbits/sec  0      receiver

iperf Done.
pi@S301-15rpi:~$ tracer 62.210.18.40
-bash: tracer : commande introuvable
pi@S301-15rpi:~$
```

```
C:\Users\admin>tracert 62.210.18.40
Détermination de l'itinéraire vers ping.online.net [62.210.18.40]
avec un maximum de 30 sauts :

 1  <1 ms    <1 ms    <1 ms    192.168.33.1
 2  1 ms     1 ms     1 ms     gw55src.pu-pm.univ-fcomte.fr [192.168.100.253]
 3  1 ms     <1 ms    1 ms     cerbere1.pu-pm.univ-fcomte.fr [194.57.85.2]
 4  *        *        *        Délai d'attente de la demande dépassé.
 5  4 ms     2 ms     2 ms     172.22.0.42
 6  6 ms     3 ms     3 ms     172.22.0.22
 7  *        *        *        Délai d'attente de la demande dépassé.
 8  3 ms     2 ms     2 ms     172.20.248.206
 9  *        *        *        Délai d'attente de la demande dépassé.
10  4 ms     9 ms     4 ms     194.57.79.198
11  4 ms     4 ms     3 ms     rr-sequane-ren-nr-besancon-rtr-091.noc.renater.fr [193.55.202.81]
12  4 ms     3 ms     2 ms     vlan1034-be2-ren-nr-besancon-rtr-091.noc.renater.fr [193.55.202.80]
13  16 ms    16 ms    16 ms     te0-0-0-9-ren-nr-dijon-rtr-091.noc.renater.fr [193.51.177.183]
14  13 ms    13 ms    12 ms     xe-1-0-3-ren-nr-lyon1-rtr-131.noc.renater.fr [193.55.204.13]
15  13 ms    13 ms    12 ms     et-3-1-7-ren-nr-paris1-rtr-131.noc.renater.fr [193.51.180.166]
16  13 ms    15 ms    12 ms     et-2-0-0-ren-nr-paris2-rtr-131.noc.renater.fr [193.55.204.193]
17  13 ms    13 ms    13 ms     193.51.189.243
18  17 ms    17 ms    17 ms     62.210.0.158
19  13 ms    13 ms    13 ms     51.158.1.35
20  16 ms    17 ms    *        45x-s44-2-a9k1.dc3.poneytelecom.eu [195.154.1.105]
21  13 ms    13 ms    13 ms     ping.online.net [62.210.18.40]

Itinéraire déterminé.
```

Figure 15 : connexion sur un serveur externe en iperf3 ainsi que les étapes effectués par ces paquets grâce à la commande tracer

Nous avons ensuite effectué des tests de vitesses à l'aide du site speedtest.net sur le RPI et sur la VM. Voici les résultats :



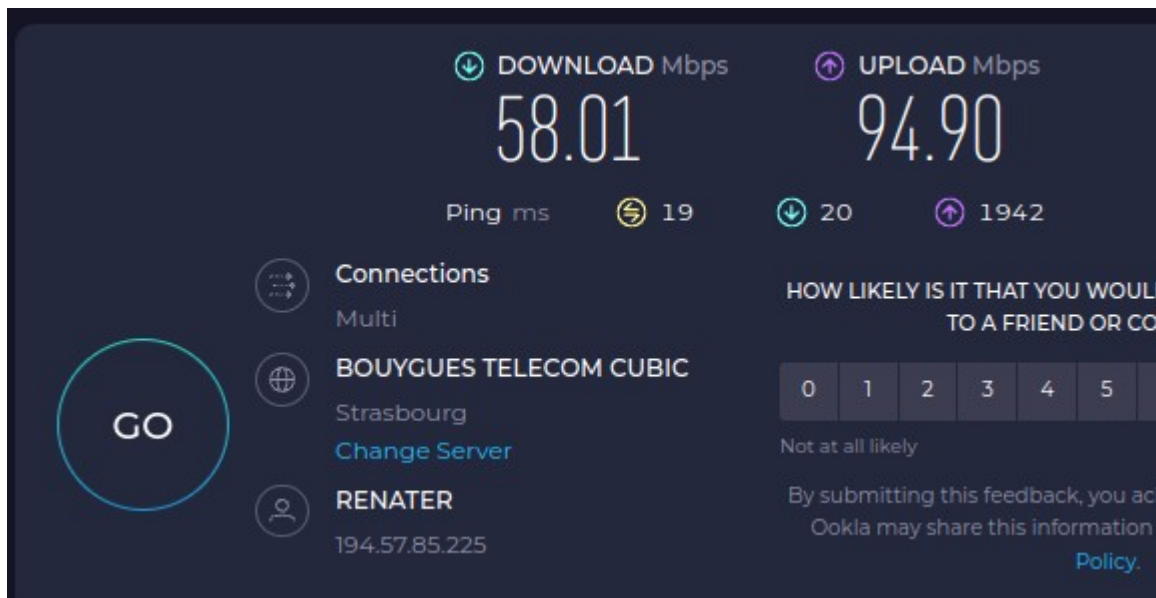


Figure 16 : capture d'écran de la vitesse sur le RPI.

On remarque que sur le RPI, la vitesse de téléchargement de paquets est plus lente que l'envoi de paquets, alors que c'est censé être plus ou moins équivalent. Cela peut être dû au fait que le réseau était surchargé et il acceptait les paquets sortants des machines, mais peinait à envoyer au destinataire les paquets qui lui étaient adressés qui transitaient par le réseau.

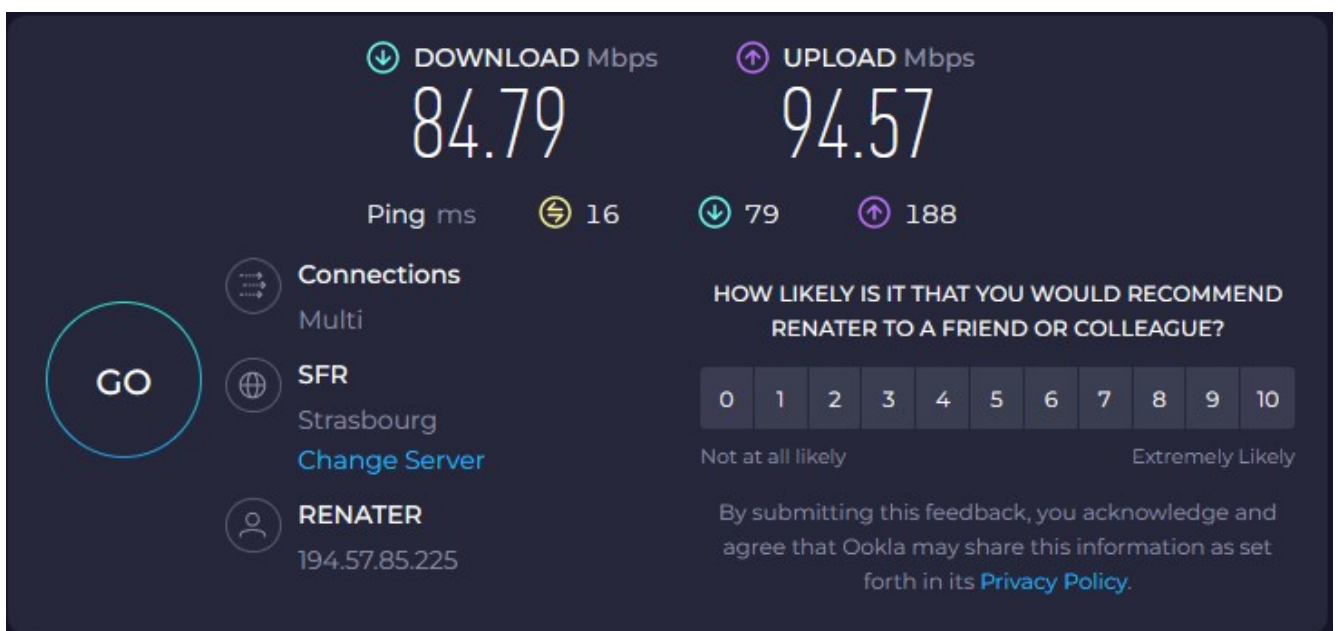


Figure 17 : capture d'écran du test de vitesse effectué sur la VM

Sur le test effectué auprès de la VM, on remarque que la vitesse de téléchargements est à peu près équivalente à la vitesse d'envoi des paquets. Si on compare avec la figure 15, il y



a plusieurs explication possible, mais nous pensons que le fait que nous avons effectué ce test à un autre moment de la journée où il y avait moins de monde connecté pouvait être la cause de cette différence.

## Partie 2 : Débit d'une source d'information

Nous avons recherché et trouvés les informations suivantes concernant le débit. Nous commencerons par les généralités, puis nous interpréterons des mesures :

Qu'est-ce que le débit ?

Il y a plusieurs type de débit d'une information :

- Voix : Téléphone (100kbit/s, pas de latence, accepte les erreurs)
- Données : Internet (débit au mieux, ne tolère pas d'erreur, latence accepté)
- Image : Télévision (10 Mbit/s, pas de latence, erreurs autorisée)

Source : Cours R305

Plusieurs causes peuvent interférer avec le débit et le ralentir :

L'Horloge, le processeur et les threads sont des causes qui sont liées aux traitements de l'information et qui peuvent potentiellement ralentir le débit.

**Vitesse d'Horloge** : correspond à la fréquence à laquelle l'oscillateur du processeur émet des impulsions. Par exemple, un processeur cadencé à 2 GHz effectue 2 milliards de cycles d'horloge par seconde, où chaque cycle représente une impulsion, équivalant à 1 Hz.

**Coeur** : Le nombre de cœurs dans un CPU détermine sa capacité à traiter simultanément plusieurs tâches. Plus un CPU dispose de cœurs, plus il peut exécuter un grand nombre de tâches en parallèle.

**Threads** : Les threads sont des canaux qui permettent aux cœurs du processeur de traiter simultanément plusieurs tâches. Dans un scénario où il y a 4 cœurs et 8 threads, chaque cœur peut gérer deux tâches en parallèle. Cette approche réduit les temps d'attente lors du traitement des tâches, ce qui améliore l'efficacité globale du CPU.

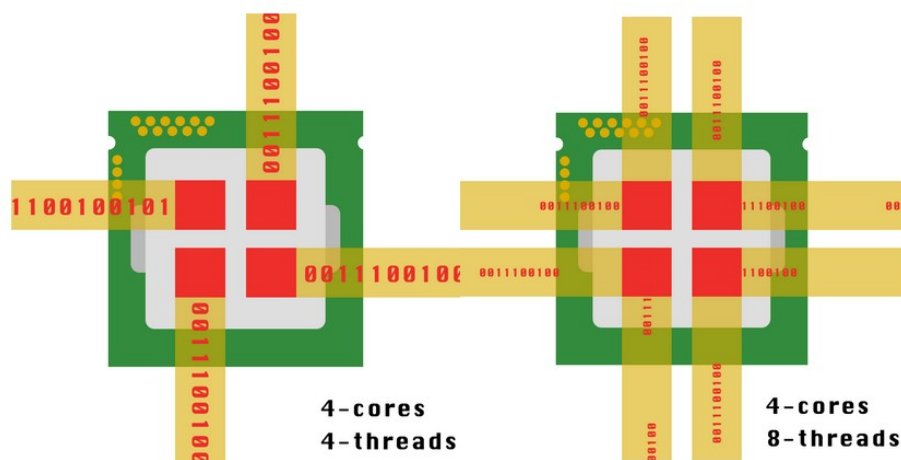


Figure 18 : Explication des threads au niveau du CPU

Ce qui peut ralentir le débit peut également être causé par les transmissions via canal, c'est à dire, par les différents type de câbles utilisés dans le réseaux.

**Électricité paire torsadée** : Une paire torsadée est une ligne symétrique composée de deux fils conducteurs enroulés en hélice l'un autour de l'autre. Cette conception en torsion permet d'éviter les phénomènes d'interférences, notamment la diaphonie. En agissant ainsi, ce câble se comporte comme un filtre passe-bas pour les hautes fréquences, réduisant la transmission des signaux indésirables. Par conséquent, la bande passante de ce type de câble est limitée aux fréquences plus basses, ce qui peut avoir un impact sur la transmission de données à haute vitesse.

**Électricité coaxe** : câble coaxial : pour les hautes fréquences le débit est élevé. On l'utilise dans le cœur du réseau quand l'utilisation de fibre n'est pas possible tel que les réseaux de télévisions par câble, la télévision par satellite.

**Fibre monomode** : Pour de plus longues distances et/ou de plus hauts débits, on préfère utiliser des fibres monomodes (dites SMF, pour *Single Mode Fiber*), qui sont technologiquement plus avancées car plus fines. Leur cœur très fin n'admet ainsi qu'un mode de propagation, le plus direct possible c'est-à-dire dans l'axe de la fibre. Les pertes sont donc minimales (moins de réflexion sur l'interface cœur/gaine, perte de puissance due à la flexion, diffusion, absorption) que cela soit pour de très haut débits et de très longues distances. Les fibres monomodes sont de ce fait adaptées pour les lignes intercontinentales (câbles sous-marins).

**Fibre multimode** : Les fibres multimodes se caractérisent par la possibilité de supporter plusieurs modes de propagation lumineuse. Ces différents mode de propagations lumineuse se caractérise par le fait que plusieurs rayons se propagent avec des distances différentes. Vu que la fibre est de la lumière, il n'y a donc pas d'influence par de champ magnétique et ainsi d'interférences magnétiques. Il y a des pertes de puissance due à l'absorption, à la diffusion et à la flexion. Son débit de données élevé.

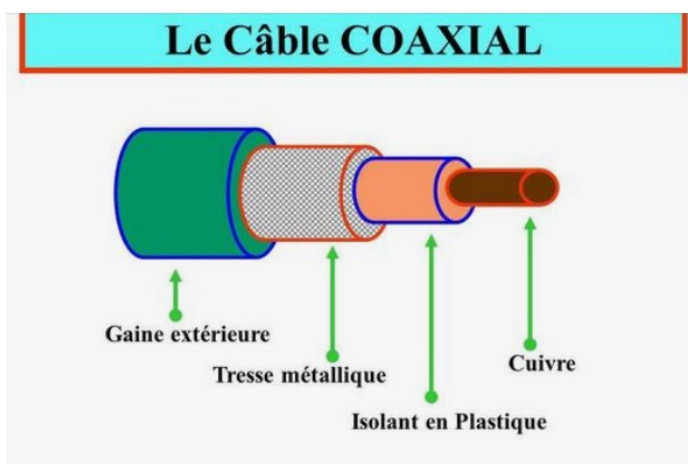


Figure 19 : Câble coaxial

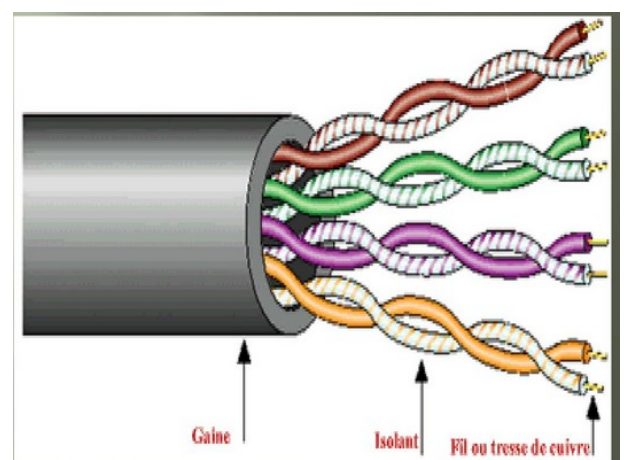


Figure 20 : Câble paire torsadée

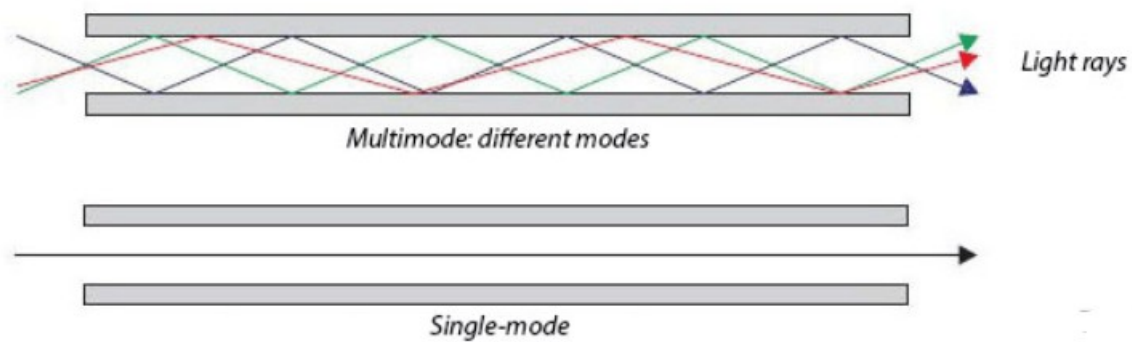


Figure 21 : schéma de la fibre multimode et la fibre monomode

Nous avons ensuite écouté de la radio sur la VM afin de remplir l'écran netspeed

Nous avons commencé par faire 3 analyses de radio, grâce au site <http://fluxradios.blogspot.fr> en écoutant la radio de franceinfo, nous avons d'abord fait une analyse du Midfi 128kbps.

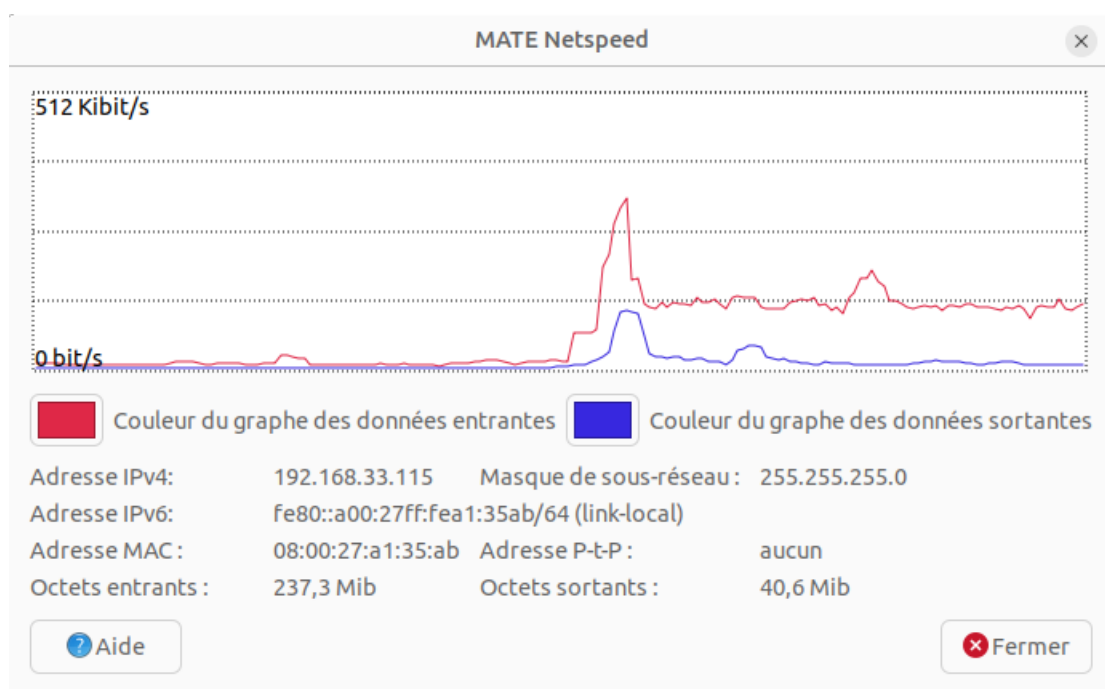


Figure 22: capture d'écran du débit de la radio franceinfo en mid-fi.

Grâce à netspeed, on constate que chaque mesure commence avec un pic de données entrantes, mais après un temps de stabilisation, on arrive à voir une diffusion beaucoup plus régulière, le graphe ayant une échelle de montrant un max de 512 Kbit/s, on arrive à avoir une diffusion de 128 Kbit/s

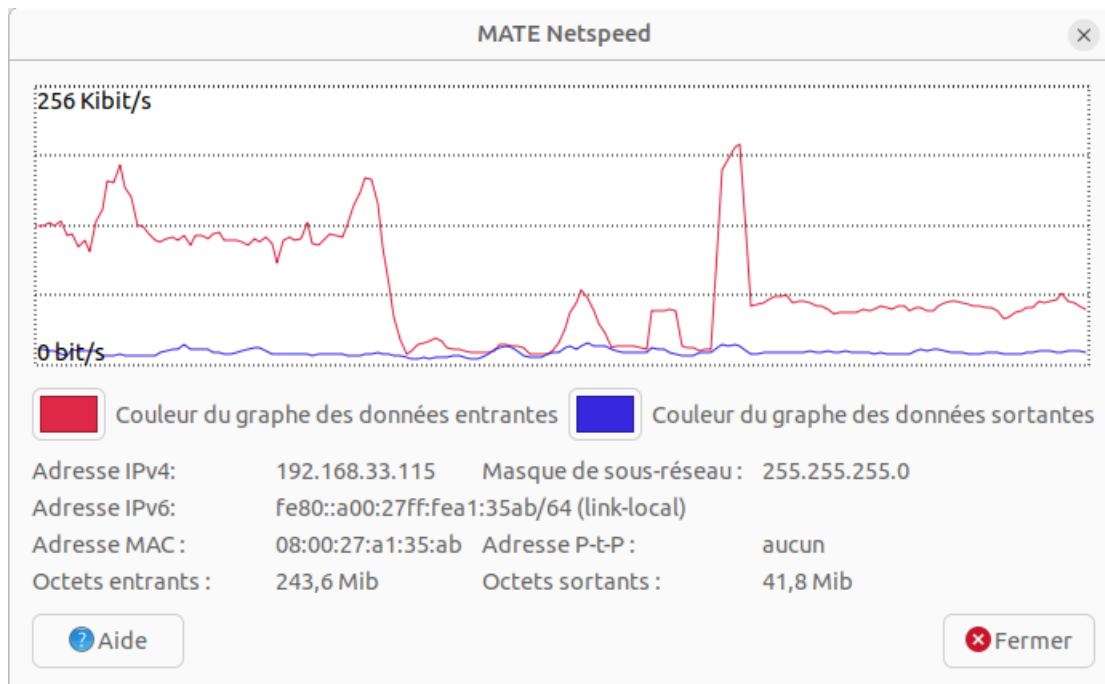


Figure 23 : capture d'écran de cette même radio en lo-fi.

Avec l'étude du lofi (32 kbps), on arrive à voir une nouvelle mesure qui est, logiquement, bien plus petite que celle d'avant, on peut même voir une échelle égal à 256kbit/s, cela signifie que notre signal est dans les alentours des 40 Kbit/s

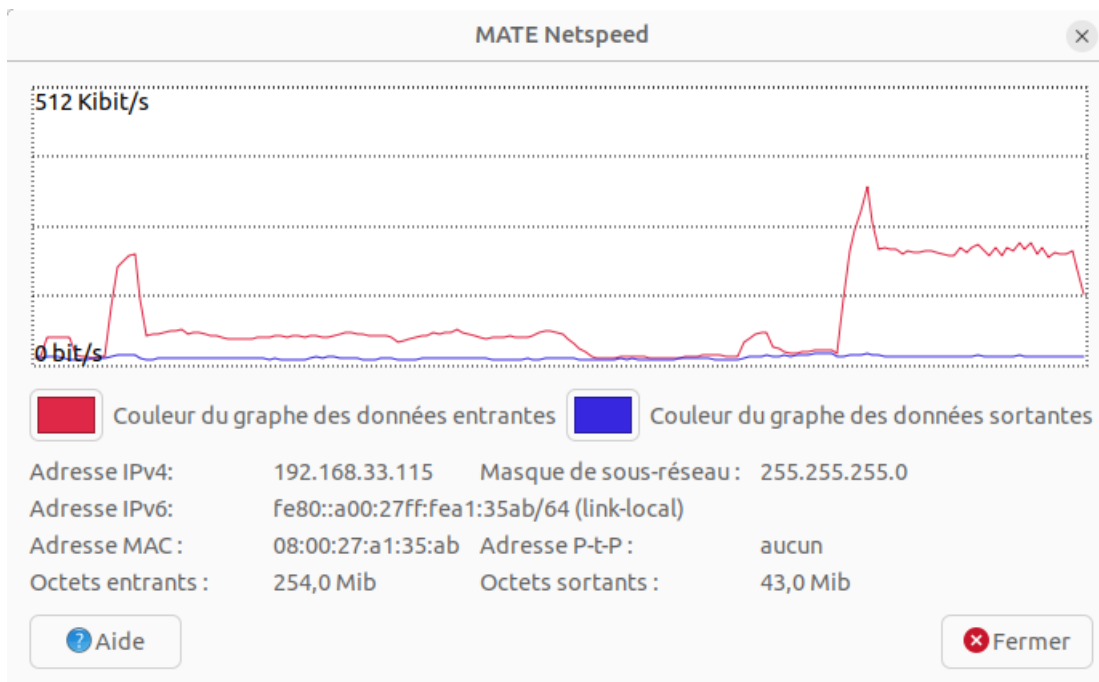


Figure 24 : Capture d'écran du débit de cette même radio en hi-fi

Enfin, avec l'étude du hi-fi (192 kbps), on peut voir une mesure qui est beaucoup plus accentuée, qui est mesurée vers dans les alentours des 192 kbps.

Nous pouvons conclure que dépendant de la qualité de l'audio, le débit peut varier. En effet, lorsqu'on est sur la même radio, mais avec des qualités différentes (lo-fi, mid-fi, hi-fi), nous observons différents débits. Cette perte de débit peut également être due au fait que selon le type de transmission, câble coaxial, paire torsadée, ces câbles peuvent affecter le débit en fonction de la qualité de l'audio, c'est à dire, au niveau de ses fréquences. Exemple : un audio de basse qualité (144p, lo-fi) aura une basse fréquence et ne pourra avoir un débit correct puisque le câble serait dans de trop haute fréquence pour cet audio.

Pour la vidéo, on a décidé de prendre une vidéo quelconque sur YouTube et on a analysé 3 résolutions différentes, le 144p, le 360p et le 1080p (aussi dit, HD)

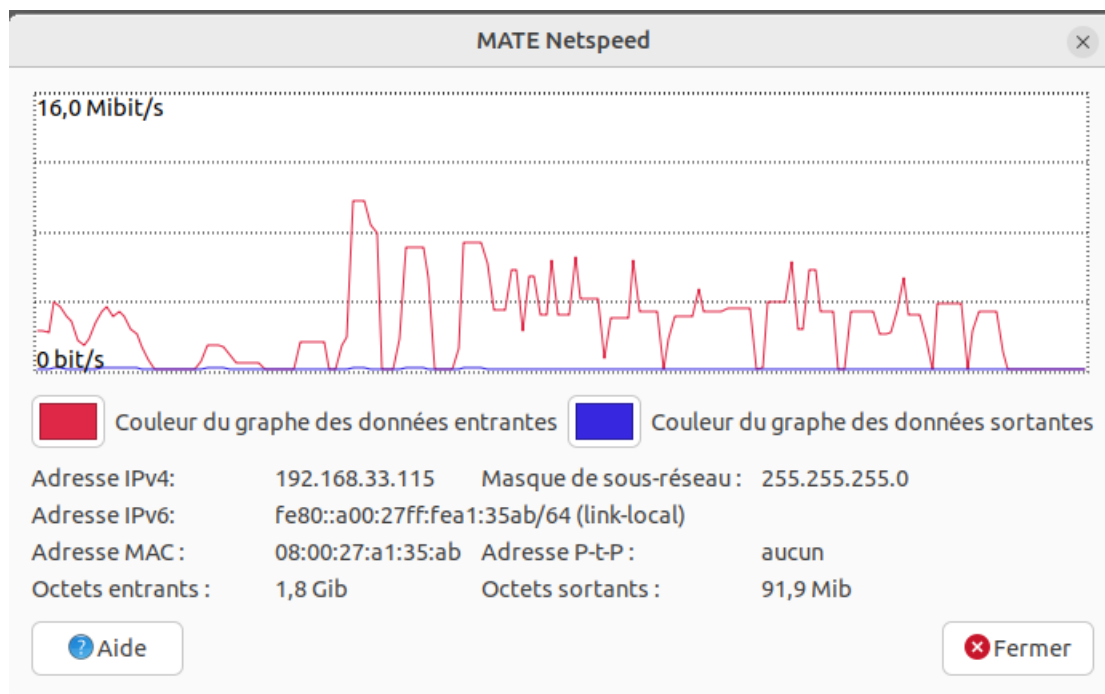


Figure 25 : capture d'écran du débit d'une vidéo youtube en qualité 1080p.

Ayant commencé avec une vidéo de 1080p, on peut voir des valeurs qui ont une variation qui est très facile à voir, avec presque 2 Gb en octets entrants et seulement 91,9 Mb en entrant sortant, on peut facilement voir ce qui est requis pour afficher une vidéo en Haute Définition



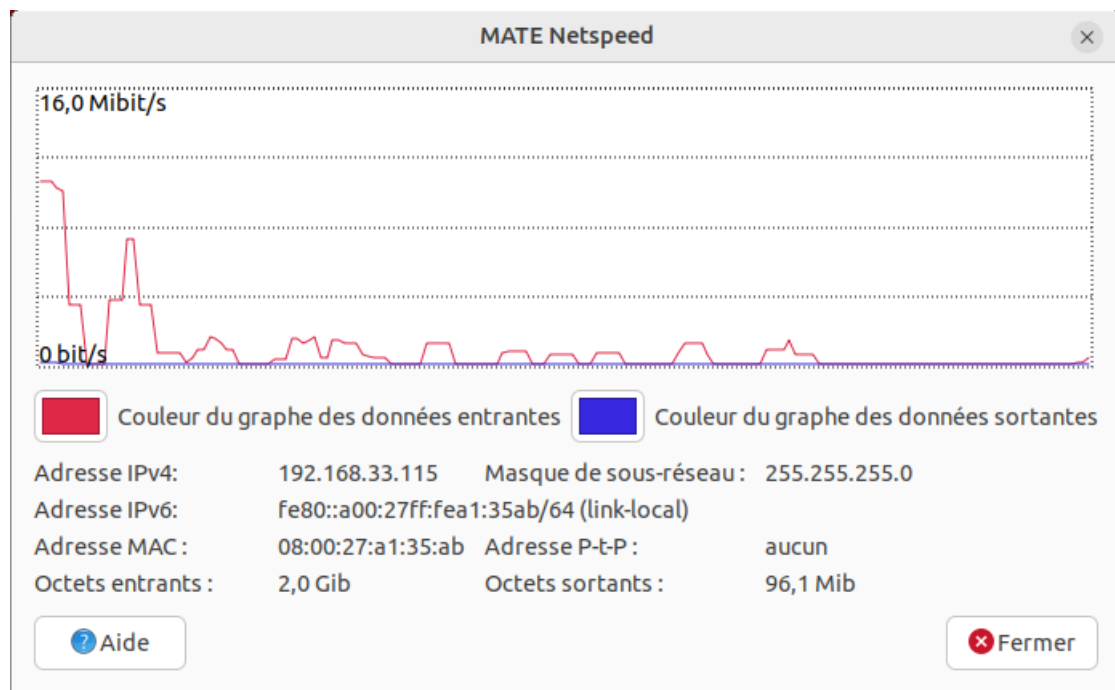
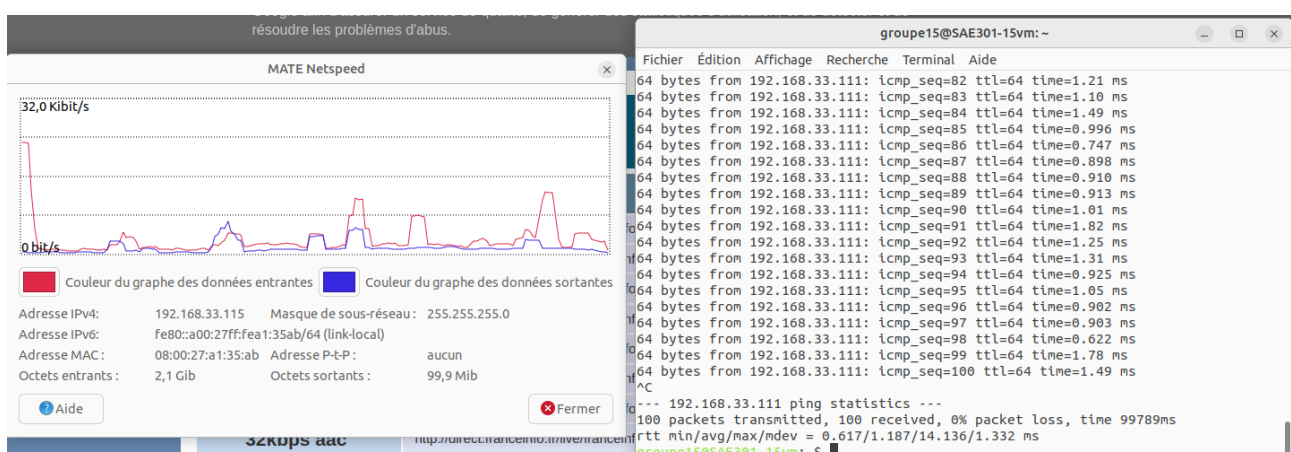


Figure 26 : capture d'écran du débit de cette même vidéo en qualité 144p.

Comparaison de cela avec la même vidéo en 144p, on peut voir une différence de nuit et jour, n'arrivant même pas au quart l'échelle dédiée à la vidéo HD on peut de nouveau voir une forme de fluctuation, on peut déduire que la valeur maximale est proche des 1 mb/s

Pour l'analyse de donnée, nous avons fait 2 mesures, une avec un ping, et une avec un téléchargement. En ping, on a eu des résultats minimes



On peut voir une échelle qui est mesurée dans les Kbit/s, avec une valeur moyenne qui n'atteint même pas les 8kbit/s

L'ordre de grandeurs des débits serait aux alentours des Mbit/s. Cependant, ce n'est qu'un ordre de grandeur, ainsi, nous pouvons très bien obtenir de faible débit en fonction de la requête effectuée.

Le débit d'une vidéo HD peut varier en fonction de nombreux facteurs, y compris la qualité de la vidéo elle-même. En général, le débit d'une vidéo HD peut varier de quelques Mbit/s (pour une qualité standard) à plus de 10 Mbit/s pour une haute qualité. Le type de débit audio serait variable. En effet, sur les différentes capture de débit avec des radios, (figure ), nous voyons que le débit n'est pas constant sur munin. Il est donc variable.

Nous avons également mesurer le débit lors d'un téléchargement. Nous avons déjà mis notre VM à jour, nous avons donc télécharger un gros paquet comme l'émulation windows wine.

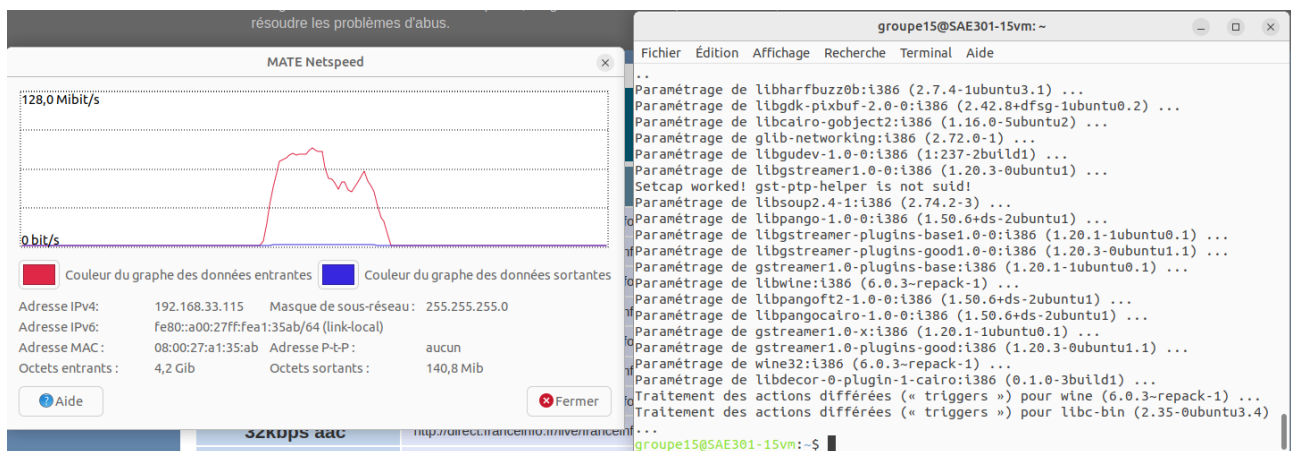


Figure 26 : écran netspeed montrant le débit de la carte réseau lors d'un téléchargement

Dans cette figure, nous pouvons observer un pic de débit atteignant environ 80 Mbit/s. Nous pouvons remarquer que cette valeur est atteinte lors du téléchargement du logiciel d'émulation Windows Wine.

## Partie 3 : Système d'analyse de débit : Munin

### Qu'est-ce que Munin ?

Munin est un système complet de monitoring (ou métrologie) de machine. Il permet de générer une série de graphes à partir des informations envoyées par les autres machines : utilisation de la mémoire vive, « load average », utilisation du processeur, réseau,...

Munin fonctionne sur le modèle client-serveur. Un petit démon tourne sur chacune des machines devant être surveillée : *munin-node*. Ce démon fournit les informations brutes concernant la machine dès que le grapheur *munin* les lui demande.

Afin d'installer Munin, il fallait installer au préalable un serveur web, tel que Apache. Nous avons pris le serveur Nginx car celui-ci a été développé pour dépasser les limites imposées par Apache (notamment dans la charge serveur).

Nous avons configuré notre réseau ainsi : La VM était le « Master » de Munin et le RPI un appareil quelconques. Après quelques configurations, nous sommes parvenus à obtenir des informations concernant la VM, tel que le débit passant par la carte réseau, son trafic, l'usage du CPU.



Figure 27 : Image de bienvenue au service Nginx sur le port 80

Nous avons constaté que les graphiques de Munin ne se mettaient à jour que toutes les 5 minutes, ce qui ne nous permettait pas d'obtenir des mises à jour en temps réel. Pour résoudre ce problème, nous avons cherché un moyen d'optimiser Munin afin que les mises à jour se fassent instantanément lorsque nécessaire. Nous avons opté pour l'utilisation de FCGI (Fast Common Gateway Interface) au lieu de CRON, qui mettait à jour les données toutes les 5 minutes.

Après avoir installé et configuré les fichiers système nécessaires pour mettre en place CGI, nous avons implémenté des plugins Munin et Nginx, en nous assurant qu'ils étaient actifs. Cela nous a permis d'obtenir des mises à jour en temps réel pour toutes les mesures, y compris celles entre le nœud Munin et le système d'exploitation du Raspberry Pi.

Ensuite, nous avons simplement redémarré les serveurs, et nous avons remarqué que lors de l'ajout de trafic avec iperf3, les graphiques se sont automatiquement mis à jour pour refléter les nouvelles données. Cette optimisation nous a permis d'obtenir des informations actualisées en temps réel, améliorant ainsi la surveillance des performances.

Dans la capture ci-dessous, nous pouvons observer le fait que après avoir réaliser un test de performances iperf, le trafic de la carte réseau enp0s3 avait des pics d'utilisations.

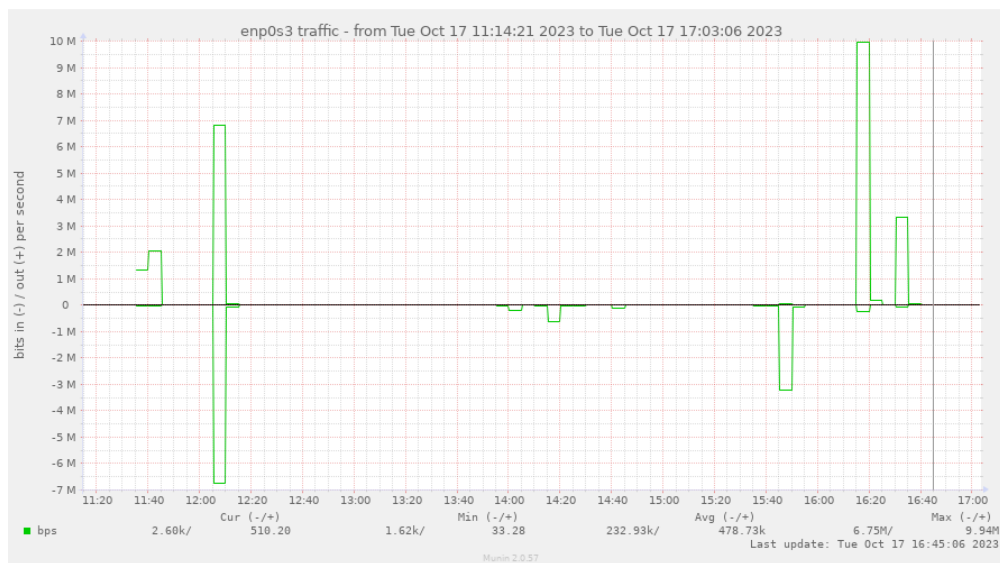
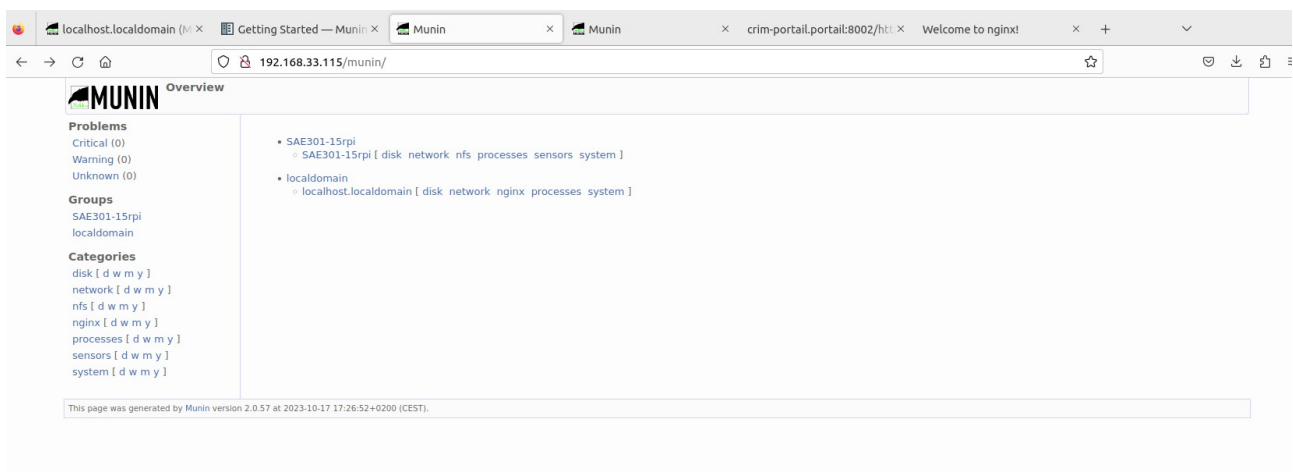


Figure 28 : Trafic passant par la carte réseau enp0s3.



Notre raspberry détecté comme un appareil du réseau. Nous pouvons donc accéder à ses statistiques.

Nous avons ensuite connecté le RPI au nœud Munin, et nous pouvons observer le trafic sur les cartes réseau des deux appareils, RPI, et VM en tant que serveur Munin dans les figures suivantes :

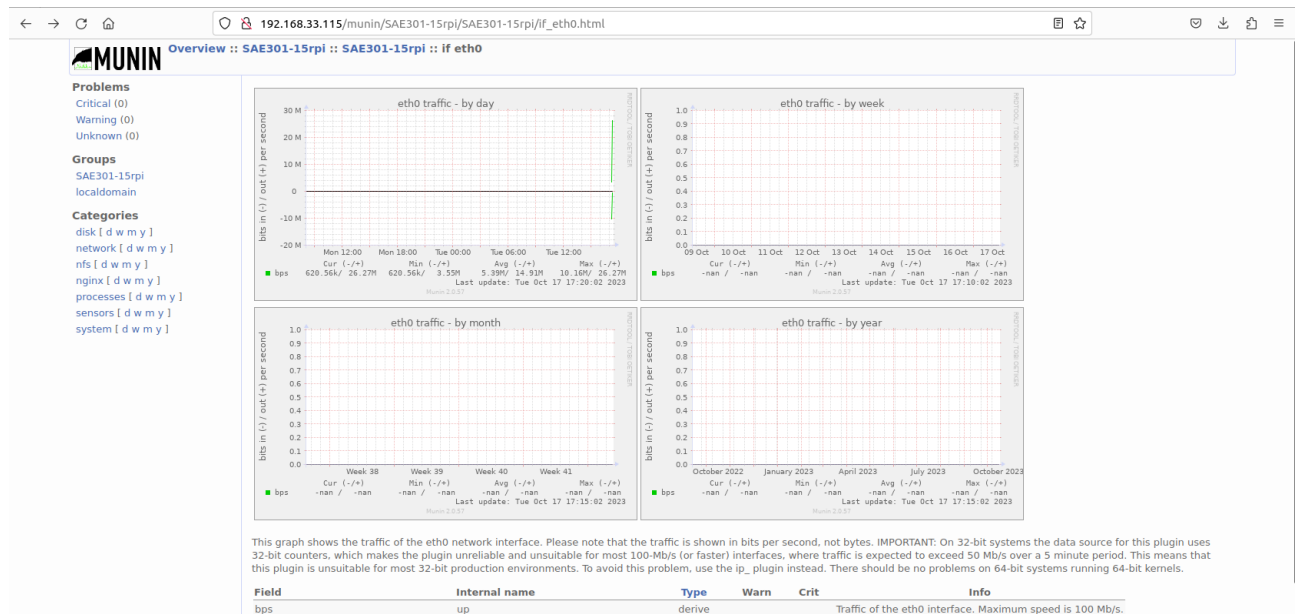


Figure 29 : trafic passant par la carte réseau Eth0 du RPI après un iperf.

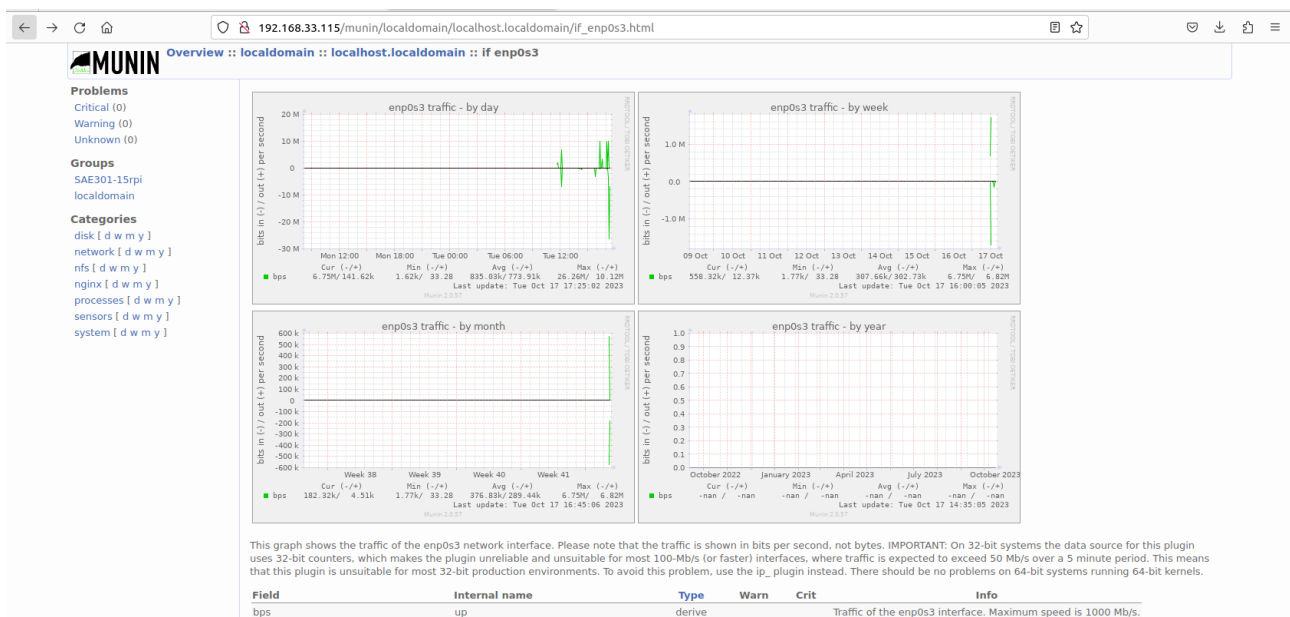


Figure 30 : trafic passant par la carte réseau enp0s3 après avoir effectué un iperf sur le serveur.

Nous observons que les pic de trafic sur ces cartes réseaux (figure 29 et 30) interviennent lorsqu'il y a eu une augmentation de trafic, par l'utilisation de iperf.

**FIN JALON 2**

## PARTIE 4 : Application a une transmission multimédia : Serveur DASH

Dans cette partie, l'objectif était de produire un résultat similaire à YouTube à propos du changement de résolution : lorsqu'on clique sur YouTube sur la résolution à 720p, la qualité vidéo change et la fluidité de celle-ci également. Nous nous sommes basé sur notre VM déjà installée avec 4 processeurs.

Afin de réaliser ceci, nous avons téléchargé 2 vidéos Youtube en 1080p. Puis, nous avons utilisé l'utilitaire ffmpeg afin de segmenter ces vidéos en différents segment de différentes qualités chacun afin de pouvoir la changer plus tard. Nous avons utilisé la commande suivantes et nous obtenions un fichier en **.mpd**.

Nous devons suivre des démarches telles que celle-ci pour que la résolution puisse changer en fonction de la bande passante :

On a au choix 2 démarches pour arriver au résultat:

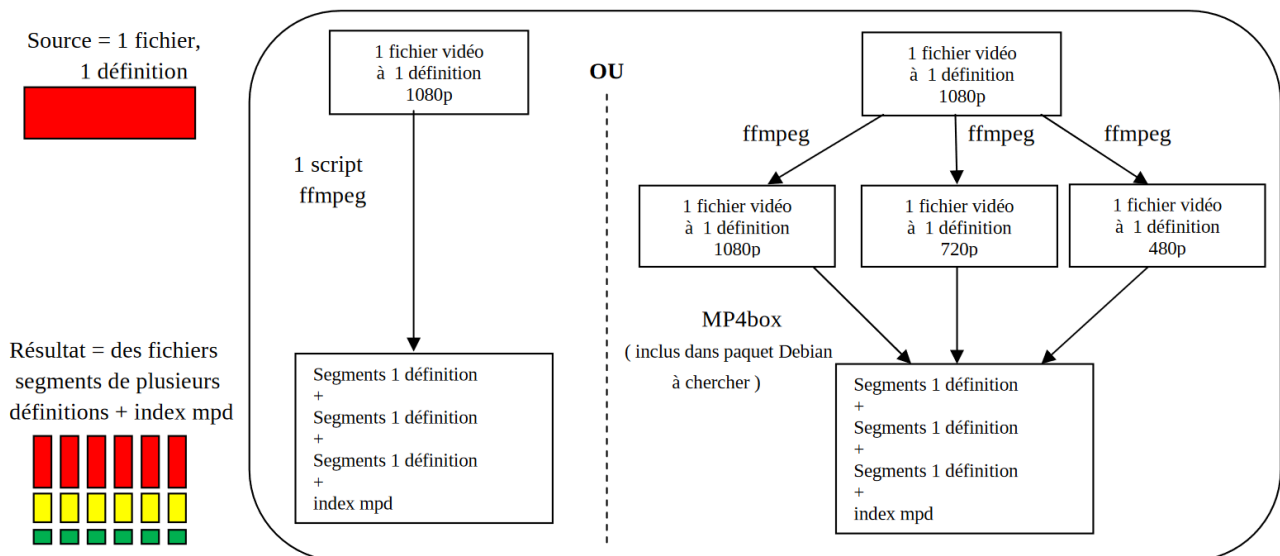


Figure 31 : schéma afin d'arriver à un serveur de débit adaptatif

Afin de segmenter la vidéo, nous avons tout d'abord crée 2 copies de ces vidéos en plus faibles qualités : une en 720p, et une autre en 480p. Nous avons utilisé pour cela la commande suivantes :

```
ffmpeg -i Metallica.mp4 -b 2M Metallica720.mp4  
ffmpeg -i Metallica.mp4 -b 500k Metallica480.mp4
```



## PARTIE 4 : Application a une transmission multimédia : Serveur DASH

Enfin, nous avons crée des segments de ces 3 vidéos grâce à la commande MP4Box et nous obtenions des segments et un fichier **.mpd**.

```
286 sudo MP4Box -dash 4000 -rap -frag-rap -profile live -out /var/www/html/dash/video1_dash.mpd  
Metallica480.mp4#video:id=480p Metallica720.mp4#video:id=720p Metallica.mp4#video:id=1080p
```

Figure 32 : commande MP4Box permettant la création d'un fichier mpd.

**-dash 4000** : créer des segments DASH de 4 secondes

**-rap**: Active le mode Random Access Points. Cela signifie que chaque fragment DASH commence à un point d'accès aléatoire dans le fichier source.

**-frag-rap**: Active le mode Random Access Points pour les fragments. Cela signifie que chaque fragment DASH commence à un point d'accès aléatoire.

**-out /var/www/html/dash2/video2\_dash.mpd**: Spécifie le chemin de sortie du fichier de description du média MPEG-DASH (MPD).

**-Metallica480.mp4#video:id=480p**: Utilise le fichier "Metallica480.mp4" en tant que source vidéo avec un identifiant "480p". Cela signifie que ce flux sera utilisé pour la résolution 480p.

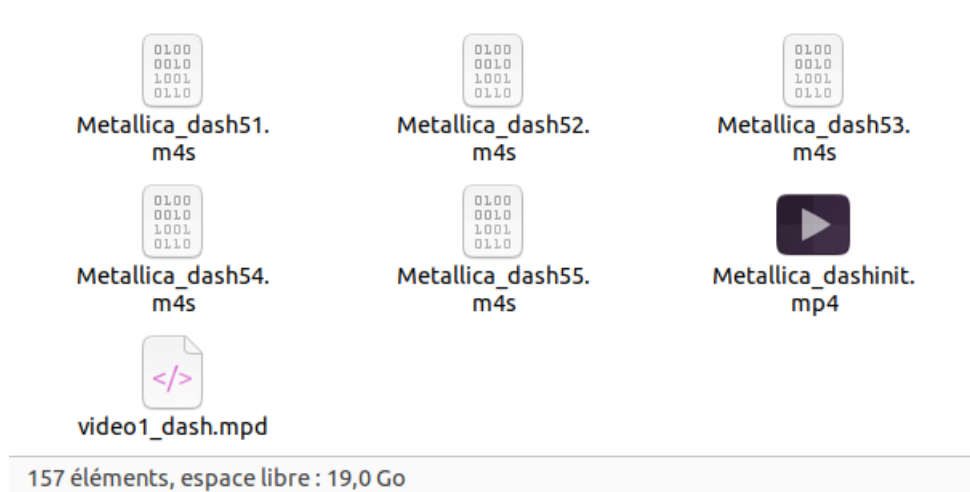


Figure 33 : Fichiers segmenté et le fichier en .mpd, qui va définir la résolution en fonction de la bande passante.

Afin de tester que cela a fonctionné, nous avons configuré le serveur web nginx pour qu'il puisse lire les vidéos. Pour ceci, nous avons créer les fichiers suivants :

- `index.html` :

```
GNU nano 6.2 index.html
<html>
  <head>
    <script src="//cdn.jsdelivr.net/npm/shaka-player@3.3.2/dist/shaka-player.compiled.min.js"></script>
    <script type="module" src="ActiveLecture.js"></script>
  </head>
  <body>
    Voici ma video
    <video id="video" controls width="75%" height="75%"></video>
  </body>
</html>
```

*Figure 34 : fichier index.html pour la première vidéo.*

- `ActiveLecture.js` :

```
GNU nano 6.2 ActiveLecture.js *
shaka.polyfill.installAll();

var video = document.getElementById('video');
var player = new shaka.Player(video);
window.player = player;

//var UrlMpd = 'https://dash.akamaized.net/akamai/bbb_30fps/bbb_30fps.mpd';
var MpdUrl = 'video1_dash.mpd';

player.load(MpdUrl);
```

*Figure 35 : Fichier ActiveLecture.js*

## PARTIE 4 : Application a une transmission multimédia : Serveur DASH

Il fallait également faire attention de bien accordé les bons droits au répertoire des segments et des vidéos. Sans cela, il nous aurait été impossible d'accéder à la page et les vidéos avec www-data.

Nous nous connectons ensuite sur la page web suivante : <http://192.168.33.111> et nous pouvions voir les vidéos et leurs résolutions changer au fur et à mesure que la bande passante se réduisait. Afin de mener à bien ces test, nous avons changer la bande passant des vidéos plusieurs fois en allant dans la console d'un navigateur web et dans l'onglet « Réseaux », nous pouvions changer la bande passante en :

- Aucune limitation de la bande passante
- GPRS
- Regular 2G
- Good 2g
- Regular 3G
- Good 3G
- Regular 4G/Lite
- DSL
- WIFI.

Nous avons fait 3 tests : 1 sans aucune limitation de bande passante, 1 en Regular 2G, et 1 autre en Good 3G.

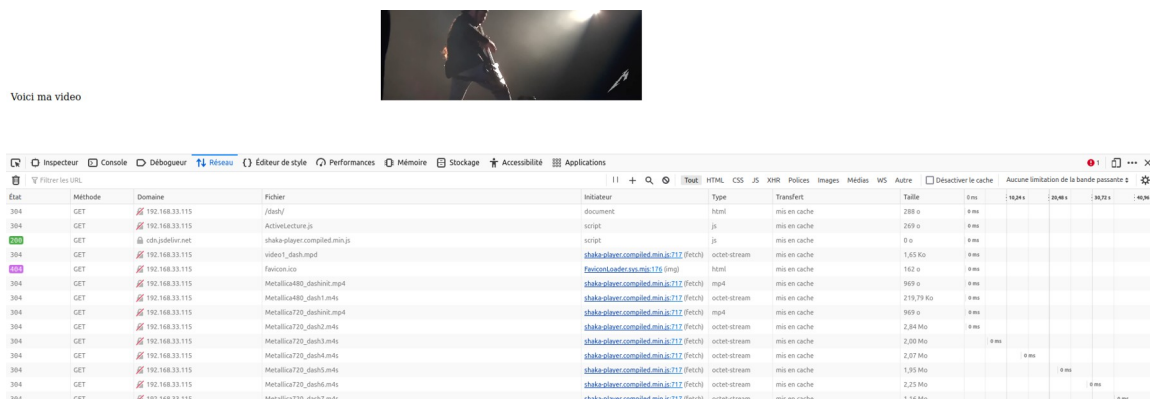
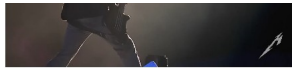


Figure 36 : vidéo sans limitation de bande passante.

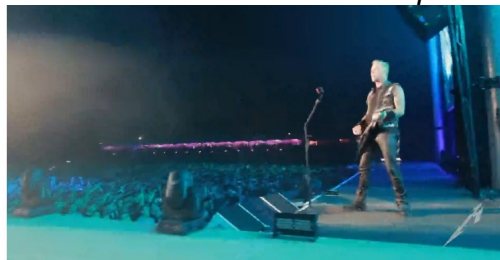
## PARTIE 4 : Application a une transmission multimédia : Serveur DASH

Voici ma video



État	Méthode	Domaine	Fichier	Initiateur	Type	Transfert	Taille	0 ms	40,36 s	1,37 ms
200	GET	192.168.33.115	/dash/	document	html	461 o	288 o	0 ms		
200	GET	192.168.33.115	ActiveLecture.js	ActiveLecture.js	js	529 o	269 o	0 ms		
200	GET	cdn.jsdelivr.net	shaka-player.compiled.min.js	shaka-player.compiled.min.js	script	134,96 Ko	411,46 Ko	129 ms		
200	GET	192.168.33.115	favicon.ico	favicon.ico	html	318 o	162 o	0 ms		
200	GET	192.168.33.115	video1_dash.mpd	shaka-player.compiled.min.js	octet-stream	1,91 Ko	1,65 Ko	0 ms		
200	GET	192.168.33.115	Metallica480_dashinit.mp4	shaka-player.compiled.min.js	mp4	1,22 Ko	969 o	0 ms		
200	GET	192.168.33.115	Metallica720_dash1.m4s	shaka-player.compiled.min.js	octet-stream	220,06 Ko	219,79 Ko	0 ms		
200	GET	192.168.33.115	Metallica720_dashinit.mp4	shaka-player.compiled.min.js	mp4	1,22 Ko	969 o	0 ms		
200	GET	192.168.33.115	Metallica720_dash2.m4s	shaka-player.compiled.min.js	octet-stream	2,84 Mo	2,84 Mo	38 ms		
200	GET	192.168.33.115	Metallica720_dash3.m4s	shaka-player.compiled.min.js	octet-stream	2,00 Mo	2,00 Mo	0 ms		
200	GET	192.168.33.115	Metallica720_dash4.m4s	shaka-player.compiled.min.js	octet-stream	2,07 Mo	2,07 Mo	0 ms		
200	GET	192.168.33.115	Metallica720_dash5.m4s	shaka-player.compiled.min.js	octet-stream	1,95 Mo	1,95 Mo	0 ms		
200	GET	192.168.33.115	Metallica480_dashinit.mp4	shaka-player.compiled.min.js	mp4	969 o	969 o	0 ms		
200	GET	192.168.33.115	Metallica480_dashinit.mp4	shaka-player.compiled.min.js	mp4	969 o	969 o	0 ms		

Figure 37 : vidéo avec limitation de bande passante en mode Regular 2G



Voici ma video

État	Méthode	Domaine	Fichier	Initiateur	Type	Transfert	Taille	0 ms	25,48 s	40,36 s	1,37 ms
200	CET	192.168.33.115	Metallica480_dashinit.mp4	shaka-player.compiled.min.js	mp4	mis en cache	969 o	0 ms	9 ms		
200	CET	192.168.33.115	Metallica480_dash5.m4s	shaka-player.compiled.min.js	octet-stream	477,66 Ko	477,66 Ko	9030 ms			
200	CET	192.168.33.115	Metallica480_dash6.m4s	shaka-player.compiled.min.js	octet-stream	629,76 Ko	629,76 Ko	9077 ms			
200	CET	192.168.33.115	Metallica480_dash7.m4s	shaka-player.compiled.min.js	octet-stream	352,00 Ko	352,00 Ko	1052 ms			
200	CET	192.168.33.115	Metallica480_dash8.m4s	shaka-player.compiled.min.js	octet-stream	348,24 Ko	348,24 Ko	1191 ms			
200	CET	192.168.33.115	Metallica480_dash9.m4s	shaka-player.compiled.min.js	octet-stream	448,06 Ko	448,06 Ko	1209 ms			

Figure 38: Vidéo avec limitation de bande passante a Good 3G.

Nous pouvons remarquer que dépendant la limitation de bande passante, la résolution des vidéos changeait ainsi que le débit et la latence. En effet, dans la figure 36, on observe que le débit est correct et une très bonne latence, et la résolution est maximale, alors que dans les autres figures avec limitation de bande passante, la résolution est passée à 480p dans les deux cas, et des débits qui baissent et une augmentation de la latence. Plus on change de sans limite de bande passante a Regular 2G, plus le débit vidéo diminue et la latence augmente.

## Partie 5: Application à une transmission multimédia : Serveur NAS

Le NAS, ou Network Attached Storage, est un appareil de stockage autonome qui peut se connecter à votre réseau privé ou professionnel via Internet. Il permet de sauvegarder, partager, sécuriser mais aussi de faciliter l'accès à vos fichiers depuis plusieurs appareils. Ce serveur était une machine virtuelle sur laquelle nous avons installé l'ISO Openmediavault (OMV). Celle-ci était hébergée sur un des deux PC physiques mis à notre disposition durant cette Saé. Afin de pouvoir configurer cette VM, nous avons deux possibilités : se connecter via SSH et ainsi configurer en ligne de commande, ou se connecter sur la VM, toujours en ligne de commande.

```
openmediavault 6.5.0-3 (Shaitan) sae301g15ovm tty1
Copyright (C) 2009-2023 by Volker Theile. All rights reserved.

To manage the system visit the openmediavault workbench:

enp0s3: 192.168.33.215

By default the workbench administrator account has the
username 'admin' and password 'openmediavault'.
It is recommended that you change the password for this account
within the workbench or using the 'omv-firstaid' CLI command.

For more information regarding this appliance, please visit the
web site: https://www.openmediavault.org

sae301g15ovm login: root
Password:
Linux sae301g15ovm 6.1.0-0.deb11.7-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.20-2~bpo11+1 (2023-04-23)
x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Oct 19 10:46:31 CEST 2023 on tty1
root@sae301g15ovm:~# _
```

Figure 40:écran de connexion à la machine virtuelle

Nous pouvons également accéder à son interface de configuration en ligne pour réaliser les différentes étapes demandées dans le sujet en rentrant son IP dans la barre de recherche.

## Partie 5: Application à une transmission multimédia : Serveur NAS

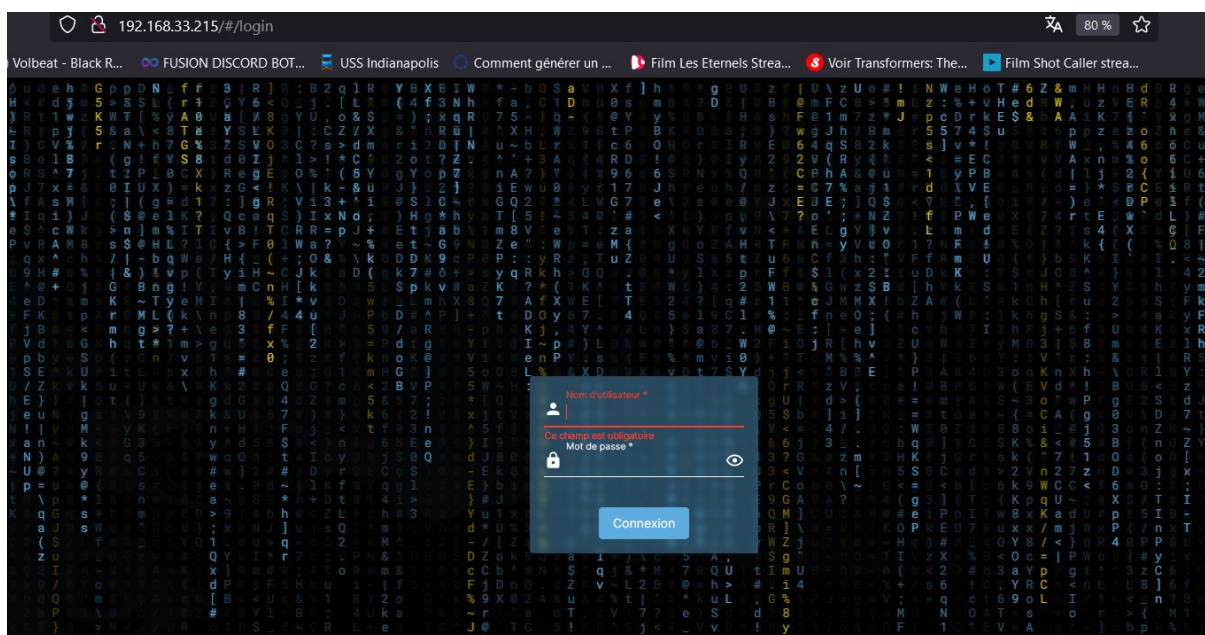


Figure 41 : écran de connexion a OpenMediaVault sur un navigateur.

Notre VM était équipée avec 2 processeurs, 2 Go de RAM, et 3 disques DUR. Nous avons utilisé le disque dur de 10 Go afin d'installer l'image ISO d'OpenMediaVault. Les deux autres disques ont été utilisé pour faire stockage de type RAID (Mirroring, soit RAID1). Le RAID est un ensemble de techniques de virtualisation du stockage permettant de répartir des données sur plusieurs disques durs afin d'améliorer soit les performances, soit la sécurité ou la tolérance aux pannes de l'ensemble du ou des systèmes. Les différents types de RAID sont résumés dans le tableau suivants :

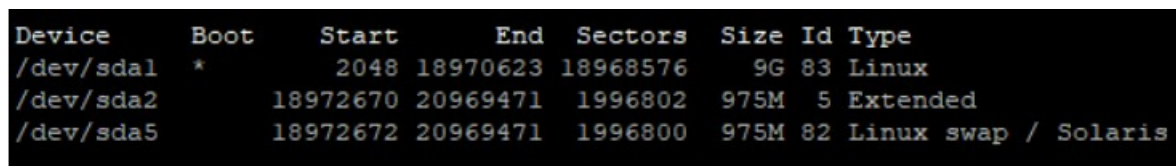
RAID 0	<ul style="list-style-type: none"> <li>■ Data striped across hard disk drives for maximum write performance</li> <li>■ No actual data protection</li> </ul>
RAID 1	<ul style="list-style-type: none"> <li>■ Synchronously mirrors all data from each HDD to an exact duplicate HDD</li> <li>■ No data lost if HDD faults or fails</li> <li>■ Typically highest-performing RAID level at the expense of lower usable capacity</li> </ul>
RAID 2	<ul style="list-style-type: none"> <li>■ Data protected by error correcting codes</li> <li>■ Parity HDD requirements proportional to the log of HDD number</li> <li>■ Somewhat inflexible and less efficient than RAID 5 or RAID 6 with lower performance and reliability</li> <li>■ Not widely used</li> </ul>
RAID 3	<ul style="list-style-type: none"> <li>■ Data is protected against the failure of any HDD in a group of N+</li> <li>■ Similar to RAID 5, but blocks are spread across HDDs</li> <li>■ Parity is bitwise vs. RAID 5 block</li> <li>■ Parity resides on a single HDD rather than being distributed among all disks</li> <li>■ Random write performance is quite poor, and random read performance fair at best</li> </ul>
RAID 4	<ul style="list-style-type: none"> <li>■ Similar to RAID 3, stripes data across many HDDs in blocks instead of RAID 3 bytes to improve random access performance</li> <li>■ Data protection is provided by a dedicated parity HDD</li> <li>■ Similar to RAID 5 except uses dedicated parity instead of distributed parity</li> <li>■ Dedicated parity HDD remains a bottleneck, especially for random write performance</li> </ul>
RAID 5	<ul style="list-style-type: none"> <li>■ Most common RAID</li> <li>■ Provides RAID 0 performance with more economical redundancy</li> <li>■ Stripes block data across several HDDs while distributing parity among the HDDs</li> <li>■ Uses HDDs more efficiently, providing overlapped read/write operations</li> <li>■ Provides more usable storage than RAID 1 or RAID 10</li> <li>■ Data protection comes from parity information used to reconstruct data of a failed drive</li> <li>■ Minimum of three and usually five HDDs per RAID group</li> <li>■ Rebuilds cause lower storage system performance</li> <li>■ Potential total RAID group data loss if second drive fails during rebuild</li> <li>■ Read performance tends to be lower than other RAID types because parity data is distributed on each HDD</li> </ul>

Figure 42 : Tableau résumant les types de RAID



## Partie 5: Application à une transmission multimédia : Serveur NAS

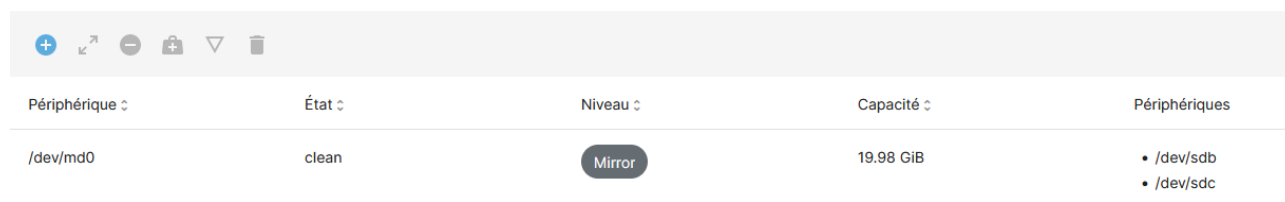
Afin que ce serveur NAS fonctionne, il a fallu configurer différents services : RAID, SAMBA. Après avoir installé RAID 1, nous pouvons vérifier le fonctionnement correct du RAID grâce à la commande `fdisk -l`, qui permet d'obtenir diverses informations à propos des disques durs de la VM.



```
Device      Boot      Start         End      Sectors  Size Id Type
/dev/sda1   *          2048    18970623   18968576    9G 83 Linux
/dev/sda2             18972670   20969471    1996802   975M  5 Extended
/dev/sda5             18972672   20969471    1996800   975M 82 Linux swap / Solaris
```

Figure 43 : Affichage des disques durs

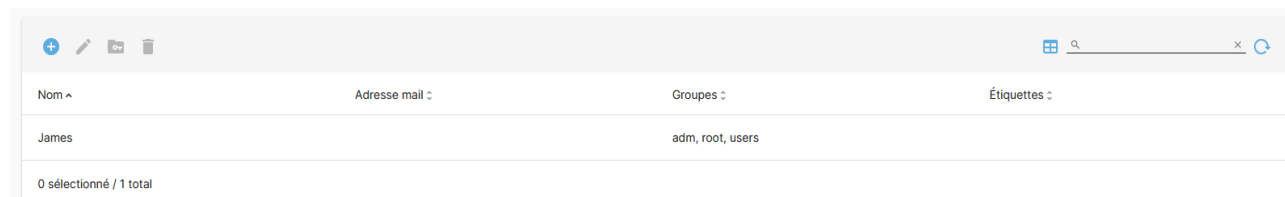
Nous pouvons également vérifier cette configuration par l'interface graphique :



Périphérique	État	Niveau	Capacité	Périphériques
/dev/md0	clean	Mirror	19.98 GiB	<ul style="list-style-type: none"><li>/dev/sdb</li><li>/dev/sdc</li></ul>

Figure 44 : Affichage du RAID dans OpenMediaVault

Par la suite, nous avons créé un utilisateur dans les menus d'OpenMediaVault. Cet utilisateur nous a permis d'effectuer différents tests sur les partages mis en place.



Nom	Adresse mail	Groupes	Étiquettes
James		adm, root, users	

0 sélectionné / 1 total

Figure 45 : page présentant notre utilisateur.

Nous avons par la suite créé deux dossiers partagés. Nous avons renseigné les informations suivantes :

- Nom : photos et vidéo
- Partition utilisée : notre partition RAID
- Le chemin d'accès au répertoire associé
- Des permissions afin de restreindre les accès aux partages

## Partie 5: Application à une transmission multimédia : Serveur NAS

Une fois que nous avons créé ces dossiers partagés, nous avons activé le service SAMBA, afin de pouvoir se connecter sur ces partages depuis le réseau. Nous avons choisi un nom pour notre machine et nous avons choisi les dossiers que nous allons partager. Nous avons coché la case héritage des autorisations. Cela permettait de reprendre les droits des utilisateurs sur le répertoire principal et de les affecter aux partages.

Une fois les manipulations effectuées, la mise en place de SAMBA, nous sommes parvenus au résultat suivants :

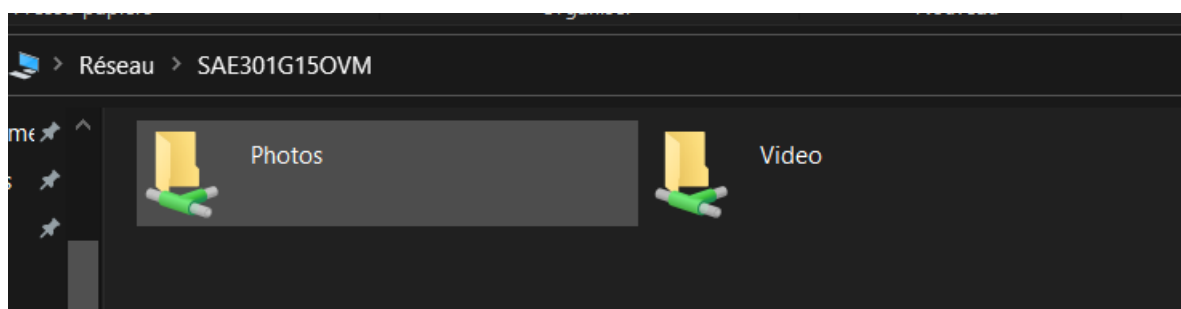


Figure 46 : les deux partages « vidéo et « photos » sont visibles.

Nous nous sommes servis de cette infrastructure pour faire les calculs suivants et relever les informations suivantes :

- Le type de fichier que nous avons choisi lors du formatage est : EXT4.
- Déduire le débit d'un téléversement et de téléchargement d'un fichier de 132 Mo en mesurant le temps pris par l'opération.

Nous nous sommes basé, pour cela, de la formule suivantes : Débit = Taille du fichier(en Bits)/temps.

- Calculs
  - Pour le téléversement,
    - le dépôt du fichier dure 11,61 secondes
    - Formule :  $\text{Débit} = \text{Taille} / \text{Temps} = 132\,706 / 11,61 = 11,4 \text{ Mbit/s}$ .
  - Pour le téléchargements,
    - Le téléchargement du fichier dure 10,9 secondes
    - Formule :  $\text{Débit} = \text{Taille} / \text{Temps} = 132\,706 / 10,9 = 12,1 \text{ Mbit/s}$ .

Nous n'avons pas réussi à installer le service FTP sur la VM avec OpenMediaVault. Cependant, nous avons dialogué avec d'autres groupes l'ayant réussie, et après comparaison de leurs valeurs et des nôtres, nous déduisons qu'il y a pas de différences significatives entre les débits des protocoles Smb et FTP. Nous remarquons également que le débit en download est un peu plus rapide que le débit en upload.

## Partie 5: Application à une transmission multimédia : Serveur VPN

Dans cette partie, nous avons mis en place un tunnel VPN. Le principe d'un tunnel VPN est d'encapsuler des données puis de les chiffrer pour empêcher des personnes externes d'accéder aux échanges. Nous avons mis en place 1 seul serveur VPN : Wireguard, et commencé à configurer un deuxième serveur VPN, OpenVPN. Cependant, nous n'avons pas pu le faire fonctionner du a sûrement des erreurs de configurations. Nous aurions pu le terminer, mais nous ne l'avons pas fait dans le temps imparti.

Afin d'installer wireguard, nous avons suivi les instructions du site : <https://www.wireguard.com/quickstart>. Nous avons donc créé une nouvelle interface sur chacune des deux machines : Le RPI, et la VM. Cette interface sera utilisée pour communiquer entre ces deux machines.

```
pi@SAE301-15rpi:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether dc:a6:32:ad:56:0f brd ff:ff:ff:ff:ff:ff
    inet 192.168.33.15/24 brd 192.168.33.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::4135:8e2b:96cc:c7cb/64 scope link
        valid_lft forever preferred_lft forever
3: wlan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether dc:a6:32:ad:56:10 brd ff:ff:ff:ff:ff:ff
4: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue state UNKNOWN group default qlen 1000
    link/none
    inet 10.0.0.2/32 scope global wg0
        valid_lft forever preferred_lft forever
pi@SAE301-15rpi:~$
```

Figure 47 : les interfaces de notre Raspberry. On remarque la nouvelle interface wg0

Nous avons donc installé les différents paquets nécessaires à l'installation du serveur VPN sur le client (RPI) et sur le serveur (VM). Ensuite, nous avons généré une clé publique et une clé privée pour chaque machine.

Une fois la clé publique et privée générée, nous avons utilisé la commande suivante sur les deux machines afin de relier ces deux machines dans un tunnel VPN.

```
valid_lft forever preferred_lft forever
root@SAE301-15vm:~# wg set wg0 peer vhb523z0P7z0j8l1Xsvxia/68KB0c0PpC1iHtGqXdE4= allowed-ips 10.0.0.2/32 endpoint 192.168.33.15:38950
```

Dans cette figure, nous appairons l'autre machine, soit la VM, à l'aide de **peer** et autorisons l' IP 10.0.0.2/32 sur le point d'entrée/sortie : 192.168.33.15, soit notre adresse IP sur l'interface enp0s3 ;

Une fois cette commande effectuée, nous pouvons vérifier que la configuration est correcte grâce à la commande **sudo wg**

```
pi@SAE301-15pi:~$ sudo wg
interface: wg0
  public key: vHbs23z0P7z0jB1lXsvxia/68KB0c0PpClIhTGqXdE4=
  private key: (hidden)
  listening port: 50072

peer: hsLcNerNj1manJsL7BJcR0W9ULMOgHbSGTEiKskDIDc=
  endpoint: 192.168.33.115:34958
  allowed ips: 10.0.0.1/32
```

Nous pouvons utiliser la commande **sudo netstat -plantu** pour voir le port du vpn si il est actif.

```
pi@SAE301-15pi:~$ sudo netstat -plantu
Connexions Internet actives (serveurs et tables)
Proto Recv-Q Send-Q Adresse locale Adresse distante Etat PID/Program name
tcp 0 0 127.0.0.1:631 0.0.0.0:* LISTEN 20285/cupsd
tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN 569/sshd: /usr/sbin
tcp 0 0 0.0.0.0:5900 0.0.0.0:* LISTEN 560/vncserver-x11-c
tcp 0 0 192.168.33.15:5900 192.168.33.236:3998 ESTABLISHED 560/vncserver-x11-c
tcp 0 0 127.0.0.1:40341 127.0.0.1:51442 ESTABLISHED 1089/vncserverui
tcp 0 208 192.168.33.15:22 192.168.33.236:3287 ESTABLISHED 1916/sshd: pi [priv
tcp 0 0 127.0.0.1:51442 127.0.0.1:40341 ESTABLISHED 560/vncserver-x11-c
tcp6 0 0 :::1:631 :::* LISTEN 20285/cupsd
tcp6 0 0 :::22 :::* LISTEN 569/sshd: /usr/sbin
tcp6 0 0 :::4949 :::* LISTEN 460/perl
tcp6 0 0 :::5900 :::* LISTEN 560/vncserver-x11-c
udp 0 0 224.0.0.251:5353 0.0.0.0:* 15088/chromium-brow
udp 0 0 224.0.0.251:5353 0.0.0.0:* 15088/chromium-brow
udp 0 0 0.0.0.0:5353 0.0.0.0:* 392/avahi-daemon: r
udp 0 0 0.0.0.0:57620 0.0.0.0:* 392/avahi-daemon: r
udp 0 0 0.0.0.0:631 0.0.0.0:* 20289/cups-browsed
udp 0 0 0.0.0.0:38950 0.0.0.0:* -
udp 0 0 0.0.0.0:68 0.0.0.0:* 466/dhcpd
udp6 0 0 :::42164 :::* 392/avahi-daemon: r
udp6 0 0 :::5353 :::* 392/avahi-daemon: r
udp6 0 0 :::546 :::* 466/dhcpd
udp6 0 0 :::38950 :::* -
```

Figure 47 : les différents ports actifs du RPI

Nous observons que le port du VPN est 38950. Effectivement, nous ne voyons pas de nom du programme, et nous pouvons également voir que ce port correspond bien au VPN au moment de la configuration par la commande suivante :

```
valid_lrt forever preferred_lrt forever
root@SAE301-15vm:~# wg set wg0 peer vHbs23z0P7z0jB1lXsvxia/68KB0c0PpClIhTGqXdE4= allowed-ips 10.0.0.2/32 endpoint 192.168.33.115:38950
```

Voici une capture d'écran des cartes réseaux des deux machines. Nous remarquons la nouvelle interface wg0 qui correspond donc au VPN.

Nous avons par la suite effectué un ping et voici une capture des trames lors du ping sur wireshark :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.0.2	10.0.0.1	ICMP	84	Echo (ping) request id=0x0008, seq=84/21504, ttl=64 (reply in 2)
2	0.000115326	10.0.0.1	10.0.0.2	ICMP	84	Echo (ping) reply id=0x0008, seq=84/21504, ttl=64 (request in 1)
3	1.001683357	10.0.0.2	10.0.0.1	ICMP	84	Echo (ping) request id=0x0008, seq=85/21760, ttl=64 (reply in 4)
4	1.001729790	10.0.0.1	10.0.0.2	ICMP	84	Echo (ping) reply id=0x0008, seq=85/21760, ttl=64 (request in 3)
5	2.003337038	10.0.0.2	10.0.0.1	ICMP	84	Echo (ping) request id=0x0008, seq=86/22016, ttl=64 (reply in 6)
6	2.003392516	10.0.0.1	10.0.0.2	ICMP	84	Echo (ping) reply id=0x0008, seq=86/22016, ttl=64 (request in 5)
7	3.004302681	10.0.0.2	10.0.0.1	ICMP	84	Echo (ping) request id=0x0008, seq=87/22272, ttl=64 (reply in 8)
8	3.004360602	10.0.0.1	10.0.0.2	ICMP	84	Echo (ping) reply id=0x0008, seq=87/22272, ttl=64 (request in 7)
9	4.005921482	10.0.0.2	10.0.0.1	ICMP	84	Echo (ping) request id=0x0008, seq=88/22528, ttl=64 (reply in 10)
10	4.005986459	10.0.0.1	10.0.0.2	ICMP	84	Echo (ping) reply id=0x0008, seq=88/22528, ttl=64 (request in 9)
11	5.007584337	10.0.0.2	10.0.0.1	ICMP	84	Echo (ping) request id=0x0008, seq=89/22784, ttl=64 (reply in 12)

Frame 1: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface wg0, id 0  
 Interface id: 0 (wg0)  
 Encapsulation type: Raw IP (7)  
 Arrival Time: Oct 19, 2023 16:20:08.126808822 CEST  
 [Time shift for this packet: 0.000000000 seconds]  
 Epoch Time: 1697725208.126808822 seconds  
 [Time delta from previous captured frame: 0.000000000 seconds]  
 [Time delta from previous displayed frame: 0.000000000 seconds]  
 [Time since reference or first frame: 0.000000000 seconds]  
 Frame Number: 1  
 Frame Length: 84 bytes (672 bits)  
 Capture Length: 84 bytes (672 bits)

Figure 49 : trames du ping chiffré

Nous devons observer sur wireshark une trame avec comme protocole « wireguard ». Cependant, surement du à une erreur de configuration, notre ping se fait en ICMP. Nous étions censé obtenir une trame ayant un protocole « wireguard » au lieu de ICMP.

Notre protocole de chiffrement était le RSA (Rivest-Shamir-Adleman). RSA est un système de chiffrement à clé publique, ce qui signifie qu'il utilise deux clés distinctes pour chiffrer et déchiffrer les données : une clé publique et une clé privée. Les données chiffrées avec la clé publique ne peuvent être déchiffrées qu'avec la clé privée correspondante, et vice versa. Cela permet une communication sécurisée sur des réseaux non sécurisés, comme Internet.

Notre clé de chiffrement publique du Raspberry était la suivante :

**vhbs23z0P7zOjB11Xsvxia/68KBOc0PpCIhTGqXdE4=**

Notre clé de chiffrement privée du Raspberry était la suivante :

**IDLLDzszto3VTLhc7ggOR7y7n9JsZOD4Wg2Y8nhWpEQ=**

Notre clé de chiffrement publique de la VM était la suivante :

**hsLcNerNjImanJsL7BJcR0W9ULMOgHbSGTEiKskDIDc=**

Notre clé de chiffrement privée de la VM était la suivante :

**yMYtvttCjC0SoEckm/i+1GKsmx9Re81E7odUmq3ajmk**

Nous avons effectué des tests de débit en utilisant iperf3, d'abord sans VPN, puis avec la connexion VPN activée. Les résultats montrent une différence de débit relativement faible et non significative. Plus précisément, nous avons mesuré une vitesse de 94 Mbits/s sans VPN et de 90 Mbits/s avec VPN, soit une différence d'environ 4 Mbits/s.

Il en découle que le débit, qu'il y ait un VPN activé ou non, est pratiquement identique. Les résultats de nos tests de débit sont présentés ci-dessous, avec le terminal de gauche

correspondant au serveur VPN et celui de droite au client VPN. La première mesure a été réalisée sans VPN, tandis que la deuxième a été effectuée avec VPN.

```

pi@SAE301-15rpi:~$ iperf3 -s
Server listening on 5201
Accepted connection from 192.168.33.115, port 33850
[ 5] local 192.168.33.15 port 5201 connected to 192.168.33.115 port 35160
[ ID] Interval      Transfer    Bitrate
[ 5] 0.00-1.00    sec 10.8 MBytes 90.9 Mbits/sec
[ 5] 1.00-2.00    sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 2.00-3.00    sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 3.00-4.00    sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 4.00-5.00    sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 5.00-6.00    sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 6.00-7.00    sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 7.00-8.00    sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 8.00-9.00    sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 9.00-10.00   sec 11.2 MBytes 94.1 Mbits/sec
[ 5] 10.00-10.26  sec 2.95 MBytes 94.1 Mbits/sec
[ ID] Interval      Transfer    Bitrate
[ 5] 0.00-10.26   sec 115 MBytes 93.8 Mbits/sec
Server listening on 5201
Accepted connection from 10.0.0.1, port 58140
[ 5] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 58156
[ ID] Interval      Transfer    Bitrate
[ 5] 0.00-1.00    sec 10.3 MBytes 86.8 Mbits/sec
[ 5] 1.00-2.00    sec 10.7 MBytes 90.1 Mbits/sec
[ 5] 2.00-3.00    sec 10.7 MBytes 90.1 Mbits/sec
[ 5] 3.00-4.00    sec 10.7 MBytes 90.1 Mbits/sec
[ 5] 4.00-5.00    sec 10.7 MBytes 90.0 Mbits/sec
[ 5] 5.00-6.00    sec 10.7 MBytes 90.0 Mbits/sec
[ 5] 6.00-7.00    sec 10.7 MBytes 90.0 Mbits/sec
[ 5] 7.00-8.00    sec 10.7 MBytes 90.1 Mbits/sec
[ 5] 8.00-9.00    sec 10.7 MBytes 90.1 Mbits/sec
[ 5] 9.00-10.00   sec 10.7 MBytes 90.1 Mbits/sec
[ 5] 10.00-10.18  sec 1.96 MBytes 90.1 Mbits/sec
[ ID] Interval      Transfer    Bitrate
[ 5] 0.00-10.18   sec 109 MBytes 89.7 Mbits/sec
Server listening on 5201

root@SAE301-15vm:~# iperf3 -c 192.168.33.15
Connecting to host 192.168.33.15, port 5201
[ 5] local 192.168.33.115 port 35160 connected to 192.168.33.15 port 5201
[ ID] Interval      Transfer    Bitrate    Retr  Cwnd
[ 5] 0.00-1.00    sec 13.6 MBytes 114 Mbits/sec  0    604 KBytes
[ 5] 1.00-2.00    sec 12.4 MBytes 104 Mbits/sec  0    1.15 MBytes
[ 5] 2.00-3.00    sec 11.2 MBytes 94.4 Mbits/sec  0    1.71 MBytes
[ 5] 3.00-4.00    sec 11.2 MBytes 94.4 Mbits/sec  0    2.27 MBytes
[ 5] 4.00-5.00    sec 11.2 MBytes 94.4 Mbits/sec  0    2.83 MBytes
[ 5] 5.00-6.00    sec 11.2 MBytes 94.4 Mbits/sec  0    3.14 MBytes
[ 5] 6.00-7.00    sec 11.2 MBytes 94.4 Mbits/sec  0    3.14 MBytes
[ 5] 7.00-8.00    sec 11.2 MBytes 94.2 Mbits/sec  0    3.14 MBytes
[ 5] 8.00-9.00    sec 10.0 MBytes 83.9 Mbits/sec  0    3.14 MBytes
[ 5] 9.00-10.00   sec 11.2 MBytes 94.4 Mbits/sec  0    3.14 MBytes
[ ID] Interval      Transfer    Bitrate    Retr
[ 5] 0.00-10.00   sec 115 MBytes 96.3 Mbits/sec  0
[ 5] 0.00-10.26  sec 115 MBytes 93.8 Mbits/sec
iperf Done.
root@SAE301-15vm:~# iperf3 -c 10.0.0.2
Connecting to host 10.0.0.2, port 5201
[ 5] local 10.0.0.1 port 58156 connected to 10.0.0.2 port 5201
[ ID] Interval      Transfer    Bitrate    Retr  Cwnd
[ 5] 0.00-1.00    sec 11.9 MBytes 100 Mbits/sec  0    574 KBytes
[ 5] 1.00-2.00    sec 12.8 MBytes 107 Mbits/sec  0    1.10 MBytes
[ 5] 2.00-3.00    sec 10.0 MBytes 83.9 Mbits/sec  0    1.58 MBytes
[ 5] 3.00-4.00    sec 11.2 MBytes 94.4 Mbits/sec  0    1.58 MBytes
[ 5] 4.00-5.00    sec 11.2 MBytes 94.4 Mbits/sec  0    1.58 MBytes
[ 5] 5.00-6.00    sec 10.0 MBytes 83.9 Mbits/sec  0    1.58 MBytes
[ 5] 6.00-7.00    sec 11.2 MBytes 94.4 Mbits/sec  0    1.58 MBytes
[ 5] 7.00-8.00    sec 10.0 MBytes 83.9 Mbits/sec  0    1.66 MBytes
[ 5] 8.00-9.00    sec 11.2 MBytes 94.4 Mbits/sec  0    1.66 MBytes
[ 5] 9.00-10.00   sec 11.2 MBytes 94.4 Mbits/sec  0    1.66 MBytes
[ ID] Interval      Transfer    Bitrate    Retr
[ 5] 0.00-10.00   sec 111 MBytes 93.1 Mbits/sec  0
[ 5] 0.00-10.18  sec 109 MBytes 89.7 Mbits/sec
iperf Done.
root@SAE301-15vm:~#

```

Figure 51 : a gauche, le test iperf3 sur le RPI en tant que serveur, à droite, en tant que client



## Partie 5: Application à une transmission multimédia : Serveur web sécurisé

Dans cette section, notre objectif était de sécuriser notre serveur web de manière à ce que seules les personnes autorisées puissent y accéder avec un mot de passe, et empêcher ainsi tout accès non autorisé. Nous avons essayé plusieurs configurations. Cependant, aucune d'elle n'a fonctionné. Nous avons demandé à d'autres groupes, et voici à quoi les fichiers de configuration nginx devaient ressembler et la page web.

Nous devons rajouter ces lignes de codes dans le fichier `/etc/nginx/sites-enabled/default` afin de pouvoir passer en https. Ces lignes de codes proviennent du groupe de Antonin, Julien, et Robin.

```
listen 443 ssl;
```

```
ssl_certificate /home/tg/Vidéos/server.crt;
```

```
ssl_certificate_key /home/tg/Vidéos/server.key;
```

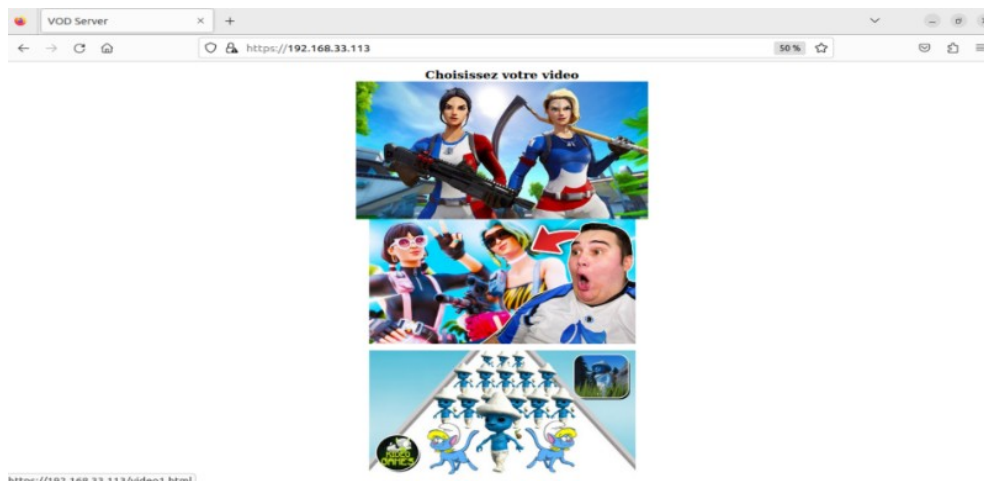
Nous devons également ajouter un fichier permettant l'authentification au site web. Pour cela, nous aurions dû ajouter les lignes suivantes dans le fichier `/etc/nginx/sites-enabled/default`, qui proviennent toujours du groupe d'Antonin, de Julien, et de Robin.

```
auth_basic "Zone sécurisée, veuillez vous authentifier";
```

```
auth_basic_user_file /home/tg/motdepasse;
```

Nous n'aurions plus qu'à réaliser la commande : `htpasswd`

A la fin, nous aurions dû arriver au résultat suivant, qui provient du groupe de Lilian, Maxence, Antoine



Nous observons que c'est bien en https par le cadenas avec un triangle noir, au lieu d'un cadenas barré d'un trait rouge.

## Conclusion

Au cours de cette Saé, nous avons mis en pratique nos compétences en réseau pour mettre en place des services et en télécommunications afin d'analyser des données liées au débit. De plus, nous avons dû recourir à des ressources telles que l'anglais pour la lecture de documentation technique et la gestion de projet afin de s'organiser de façon à être productif.

Nous avons rencontré certains problèmes tels que :

- Notre seconde VM ne voulait pas télécharger le plugin ftp pour OpenMediaVault, nous empêchant momentanément de réaliser des calculs, des mesures de débit et des interprétations. Malgré l'aide de Monsieur Millet, et d'autres groupes, nous n'avons pas réussi à installer ce plugin sur OpenMediaVault.

- Lors de la création du serveur vidéo, la configuration des fichiers afin de lire les vidéos était un peu compliquée mais grâce à l'entraide avec d'autres groupes, nous avons trouvé des solutions et résolu ce problème.

- La configuration du serveur HTTPS. En effet, malgré l'aide de plusieurs groupes, nous n'avons pas réussi à configurer le serveur web en https.

- Notre VPN marchait le jeudi, mais au moment de la vérification le vendredi avec Monsieur Millet, celui-ci ne marchait plus. Cependant, nous avons des traces que celui-ci fonctionnait.

Pendant cette Saé, nous avons pu collaborer avec d'autres groupes, ce qui a encouragé l'entraide et nous a permis d'obtenir un nouveau point de vue sur un problème donné. Cette expérience a renforcé notre aptitude à travailler efficacement en équipe. Nous avons également dû résoudre des problèmes en cherchant activement des informations par nous-mêmes. Cette expérience nous a permis d'acquérir des connaissances que nous pourrions utiliser dans les projets à venir et dans notre future vie professionnelle.