



# **Projet de département informatique : Réseaux hybrides haute performance pour applications critiques**

Étudiants : Emilie Enfroy, Tom Marini

Encadrant : Laurent Ciarletta

École des Mines de Nancy  
département informatique, deuxième année

2022-2023

# Sommaire

|   |           |
|---|-----------|
| <b>1. Introduction</b>                          | <b>3</b>  |
| <b>2. Caractéristiques d'un réseau</b>          | <b>4</b>  |
| <b>3. Les types de Wi-Fi</b>                    | <b>4</b>  |
| <b>4. Wi-Fi mesh</b>                            | <b>5</b>  |
| a. Principe                                     | 5         |
| b. Implémentation choisie : BATMAN              | 6         |
| c. Asus AiMesh                                  | 7         |
| <b>5. Raspberry Pi et installation BATMAN</b>   | <b>8</b>  |
| a. Configuration du Raspberry Pi                | 8         |
| b. Connexion ssh                                | 9         |
| c. Installation de BATMAN                       | 10        |
| d. Test de performance                          | 13        |
| <b>6. Capteur de température/humidité DHT11</b> | <b>14</b> |
| <b>7. Installation finale</b>                   | <b>17</b> |
| <b>8. Conclusion et perspectives</b>            | <b>20</b> |
| a. Le projet                                    | 20        |
| b. Pistes d'améliorations du projet             | 21        |
| <b>9. Références et bibliographie</b>           | <b>22</b> |
| <b>10. Liste du matériel</b>                    | <b>22</b> |

# 1. Introduction

Un réseau est un élément fondamental du transport de l'information, il fournit l'accès des utilisateurs à internet, il connecte de nombreux services entre eux, il assure la connectivité des équipements informatiques... En somme, il est la plateforme de transit de données provenant d'usages très diversifiés. Un réseau doit par conséquent répondre aux exigences de ses applications, par exemple en étant rapide, sécurisé, robuste, à grande bande passante, à grande couverture...

Ce projet consiste à travailler sur les **réseaux hybrides**, c'est-à-dire mêlant plusieurs types de technologies de réseau et d'accès à internet. L'objectif des réseaux hybrides est surtout de s'adapter aux différentes gammes de connexions en proposant des technologies moins coûteuses pour les connexions les moins critiques, tout en réservant des technologies plus coûteuses et performantes pour des applications spécifiques gourmandes en débit, nécessitant une très faible latence ou qui exigent une grande robustesse. C'est par exemple le cas des calculs en temps réel (edge/fog computing) et des applications en robotique.

**L'objectif** de ce projet est de mettre en place un **réseau de capteurs** (température, pression, hygrométrie) dans l'école. Nous pourrons déployer un réseau avec une antenne **satellite**, et ainsi la solution pourra se présenter comme un réseau de secours à l'école ou dans des environnements complexes.

Nous avons utilisé un certain **matériel** disponible ou commandé au techlab :

- une antenne satellite Starlink modèle Standard
- des routeurs mesh Asus rt-ax92u
- des mini-ordinateurs Raspberry Pi 3 B+ 2017
- des capteurs de températures DHT11 et de batteries portables
- cartes SD, câbles RJ45 (ethernet), ...

Une **liste complète du matériel** utilisé est donnée à la fin du rapport.

Notre installation est la suivante : nous connectons les capteurs de température aux mini-ordinateurs portables Raspberry Pi alimentés par batterie puis nous envoyons les données dans nos tables grâce au **réseau mesh** installé sur les Raspberry Pi (BATMAN) ou bien sur les routeurs AiMesh Asus. L'accès à internet est soit fourni par l'école via l'Université de Lorraine, soit fourni par satellite.

## 2.Caractéristiques d'un réseau

Comme évoqué en introduction, il existe des applications bien spécifiques qui nécessitent des modes de communication avec certaines caractéristiques. Voici une liste de quelques performances utiles pour caractériser les réseaux :

- **Débit** ou bande passante: la quantité d'information par unité de temps, par exemple 50 Mbits/s.
- **Latence** : le temps pour échanger de l'information (un paquet) d'un point à un autre du réseau.
- **Jitter** : variation de la latence dans le temps.
- **Fiabilité** : capacité à délivrer les paquets sans erreur.
- **Robustesse** : la capacité du réseau à offrir une connexion performante à chacun des nœuds du réseau même en cas de panne.
- **Sécurité**
- **Couverture**

Un réseau informatique est aussi caractérisé par le protocole qui effectue le routage. C'est un point important pour la suite du projet puisque nous utiliserons le protocole BATMAN pour effectuer le routage sur des mini-ordinateurs Raspberry Pi.

L'un des objectifs est de proposer un réseau flexible et robuste, c'est-à-dire qui maintient une connexion et des bonnes performances même en cas de panne de l'un des nœuds du réseau. On peut se demander quel type de connexion réseau est pertinent pour ce projet. On aura ici à faire à un **réseau local** (LAN Local Area Network). En vue d'applications robotiques ou de domotique, il est préférable que ce réseau soit sans fil. Bien qu'il existe plusieurs types de technologies sans fil (RFID, NFC, Bluetooth, ZigBee, WiFi, 5G, Cellulaire, SigFox, LoRa) [1], la solution la plus répandue pour ce type de projet reste tout de même le **WiFi** qui est présent sur la plupart de nos appareils.

## 3.Les types de WiFi

Il existe plusieurs standards de WiFi qui sont des déclinaisons de la norme IEEE 802.11 tel que le WiFi 6 (802.11.ax) dernier en date. Chaque déclinaison exploite différemment les types d'ondes radio 2.4GHz et 5GHz et traite différemment les signaux.

En plus des standards WiFi, il existe différents modes de connexion. Les plus courants sont le mode infrastructure et le mode ad hoc.

- Dans le mode **infrastructure**, chaque terminal se connecte au réseau par l'intermédiaire d'un point d'accès. Un utilisateur à la capacité de changer de point d'accès s'il change de zone de couverture.
- Dans le mode **ad hoc**, tous les terminaux ont la capacité d'échanger des données deux à deux. Autrement dit, chaque machine est à la fois client et routeur. La topologie est en mode pair à pair.

## 4. Wi-Fi mesh

### a. Principe

Un réseau **mesh** est un réseau ad hoc évolué où certaines machines vont pouvoir jouer le rôle de routeur en plus d'être des points d'accès. Cela change totalement du réseau ad hoc dans lequel deux nœuds trop éloignés ne peuvent pas échanger des données même s'il voient des nœuds intermédiaires. On parle alors d'un réseau à **topologie maillée**. Cela permet à nœud du réseau de pouvoir passer par des nœuds intermédiaire pour envoyer des données.

Le principal **avantage** du réseau mesh est sa résilience. En effet, même si un nœud central tombe en panne alors les autres nœuds vont pouvoir communiquer en empruntant d'autres routes au détriment d'un peu de performance. Ceci évite d'isoler une partie du réseau. Le cerveau du réseau mesh répartit la charge de routage sur les nœuds les plus performants ce qui délesté les terminaux de la tâche de routage. Cela veut aussi dire que le réseau passe facilement à l'échelle sans configuration supplémentaire majeure. Un autre avantage est que tous les nœuds peuvent être mobiles, y compris les routeurs principaux.

Finalement, le réseau mesh est plus flexible mais moins performant qu'un réseau d'infrastructure lorsqu'il s'agit des connexions directes. La flexibilité permet de couvrir une zone de manière plus robuste aux pannes avec des terminaux mobiles.

La solution WiFi mesh se répand dans les réseaux domestiques qui nécessitent une grande couverture. Elle a l'avantage d'être simple à mettre en place contrairement à certains types de répéteurs et amplificateurs de signal (pas tous). Le passage de la connexion d'un nœud du réseau mesh à l'autre est automatique donc transparent pour l'utilisateur et le routage est adaptatif, le réseau choisi automatiquement la meilleure route pour optimiser les performances. Le réseau est robuste et passe facilement à l'échelle, en cas de panne (suppression d'un nœud) ou d'ajout d'un nouveau nœud dans le réseau, la technologie mesh permet de rediriger automatiquement les données pour maintenir la connexion.

## b. Implémentation choisie : BATMAN

Plusieurs **implémentations** du réseau mesh existent. Elles sont basées sur la norme IEEE (Institute of Electrical and Electronics Engineers) 802.11s qui spécifie les réseaux mesh. En consultant la page wikipedia sur les réseaux mesh sans fils [2] on s'aperçoit que de nombreuses implémentations de ce type de réseau existent. **BATMAN** (Better Approach to Mobile Adhoc Networking) est présenté et se veut être une amélioration du réseau ad hoc développé par l'équipe **MANET** (Mobile Adhoc Network) de l'IETF (Internet Engineering Task Force). C'est également cette implémentation que nous recommande notre tuteur M. Ciarletta.



Logo du projet BATMAN

L'objectif de BATMAN est de définir un nouveau protocole de **routage**, c'est-à-dire la manière d'envoyer les ressources entre les nœuds du réseau ou encore la manière de définir le chemin emprunté par les paquets pour aller d'un point à l'autre du réseau.

Le projet BATMAN part du constat que les protocoles de base de routage sont très peu adaptés au réseau ad hoc MANET. Le protocole OLSR devait résoudre une partie de ces problèmes et permettre de former des réseaux maillés à grande échelle mais c'est finalement le projet BATMAN qui apportera des corrections stables.

Le **principe de routage** proposé par les développeurs de BATMAN repose sur le partage d'informations en empruntant les meilleures connexions de chaque nœud, chacun ayant à disposition les routes des messages qu'il a reçu. En effet, chaque nœud se signale régulièrement en broadcast (diffusion d'un message à tous les nœuds) et ses voisins relaient ce signalement aux autres ce qui permet au réseau complet de connaître ce nœud. Pour trouver le meilleur chemin vers un nœud, le nœud émetteur passe alors en revue les différentes routes par lesquelles ce nœud s'est signalé et choisit la meilleure. Le voisin qui lui a transmis la meilleure route est alors sélectionné pour l'envoi du paquet. Puis le processus se répète de nœud en nœud jusqu'à la destination.

Imaginons qu'un nœud veuille communiquer avec un nœud de son voisinage mais qui est très distant. Alors les performances de la route ne seront pas optimales : perte de paquets, force du signal (RSSI Received Signal Strength Indicator en dB) trop faible du fait de la distance, congestion réseau potentielle du nœud... On peut donc imaginer que le réseau va choisir une route plus efficace même si elle est moins directe. À l'inverse, si finalement il n'y

a que des nœuds peu performants en tant que routeurs ou alors réservés à d'autres applications, le réseau peut choisir la route directe. Voilà tout l'enjeu des réseaux mesh.

Pour évaluer la **force du signal** on peut utiliser le module airport sur macOS en exécutant cette commande :

```
"/System/Library/PrivateFrameworks/Apple80211.framework/Versions/Current/Resources/airport -I | grep CtlRSSI"
```

On peut aussi scanner tous les points d'accès :

```
"/System/Library/PrivateFrameworks/Apple80211.framework/Versions/Current/Resources/airport -s"
```

### c. Asus AiMesh



Nous disposons au techlab de 10 **routeurs rt-ax92u AiMesh** fabriqués par l'entreprise multinationale taïwanaise d'équipements informatiques et électroniques Asus. Ce modèle de routeur fait partie de la gamme de produits réseau et se présente comme un produit de haute qualité. La technologie AiMesh permet de créer un réseau mesh de topologie maillée couplé à de l'analyse de données pour améliorer la gestion du réseau et notamment le routage grâce à des analyses de la qualité du signal, de la latence et la congestion des routeurs du réseau. Cela peut par exemple permettre d'anticiper les besoins en bande passante, changer de bande entre 2.5GHz, 5GHz. L'objectif de ces modèles est principalement d'assurer une meilleure couverture WiFi dans une maison ou des bureaux, tout en garantissant une connexion stable et performante. Le débit théorique peut aller jusqu'à plusieurs Gb/s. La technologie embarquée par ces routeurs n'est évidemment pas open source et elle est propriétaire.

Une interface de gestion et de configuration du réseau est bien évidemment fournie avec les routeurs. Les utilisateurs ont la possibilité de prioriser le trafic suivant la machine, de configurer des pare-feu et du contrôle parental. Le réseau passe facilement à l'échelle, on peut y ajouter de nouveaux routeurs aisément. La sécurité est aussi gérée par les routeurs.

Nous exploitons ces routeurs en y installant des capteurs de température DHT11 par l'intermédiaire d'un mini-ordinateur Raspberry Pi. C'est une alternative au réseau BATMAN installé sur les Raspberry Pi qui peut présenter des avantages au niveau de gestion du réseau. Une fois le réseau configuré, il suffit de gérer les routeurs via l'interface Asus.

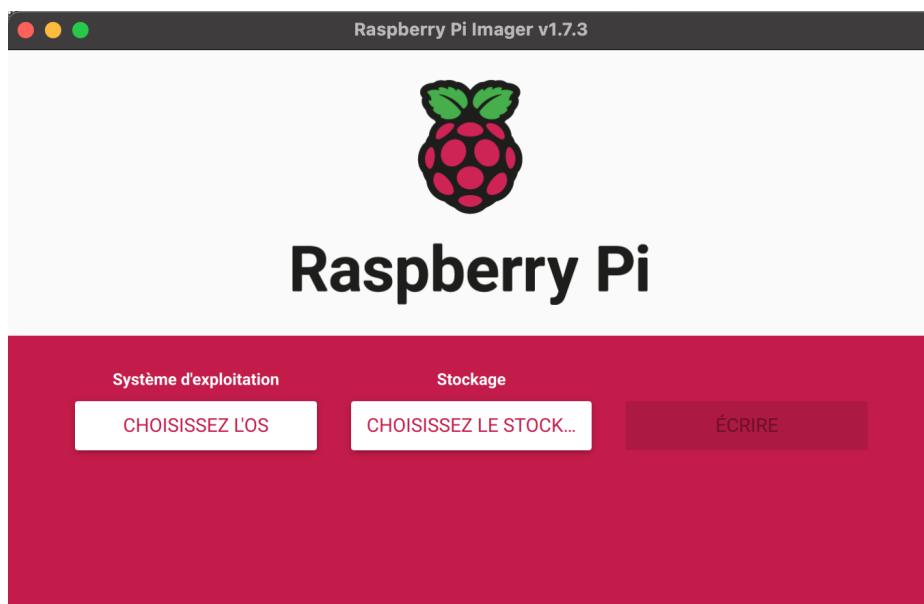
Le gros bémol est que ces routeurs consomment bien plus et sont beaucoup plus imposant que le réseau formé seulement par les Raspberry Pi avec BATMAN. En effet, les spécifications des alimentations indiquent 19V et 1.75A(max) pour les routeurs Asus et 5.1V et 2.5A(max) pour les Raspberry Pi. Ceux-ci sont d'ailleurs très consommateurs comparés aux cartes Arduino UNO 5V et 100mA(max). Cependant les Arduino UNO n'ont pas de systèmes d'exploitation permettant d'installer des protocoles de routage tels que BATMAN. Certains produits proposés par Arduino permettent de le faire mais ils consomment plus d'énergie. Quoi qu'il en soit, le hardware pourrait être judicieusement choisi de manière à ne pas surdimensionner l'installation. C'est le même principe que les réseaux hybrides : il s'agit de disposer de la bonne technologie au bon endroit et de manière proportionnée à la tâche à accomplir.

## 5.Raspberry Pi et installation BATMAN

### a. Configuration du Raspberry Pi

Pour configurer un mini-ordinateur Raspberry Pi, il faut commencer par installer le **système d'exploitation** qui va permettre de lancer des commandes dans un terminal et d'exécuter du code. Nous avons choisi l'OS (Operating System) *Raspberry Pi 32 bits Desktop* mais c'est tout à fait possible de réaliser l'installation sur une version plus légère. Nous avons essayé avec la version *Raspberry Pi 32 bits Lite* mais cela pose des difficultés pour configurer le réseau BATMAN et exploiter les données des capteurs, il manque des packages que nous n'avons pas l'habitude d'installer.

Pour installer l'OS, il faut formater une carte **micro SD** à l'aide d'un logiciel tel que *imager*.



Interface de formatage de la carte micro SD  
sur le logiciel *Imager*

Choisissez un OS, il se téléchargera. Allez dans les paramètres pour activer le ssh et bien noter 3 informations : **identifiant**, **mot de passe** et **nom d'hôte**.

Important : Une fois formatée, ouvrir le contenu de la carte SD et y insérer un fichier vide nommé “ssh” à la racine. Le fichier peut être simplement créé par la commande “`touch ssh`” sur MacOs . Cela sera utile pour la **connexion en ssh** (Secure Shell) au Raspberry Pi. Le protocole ssh permet d'établir une connexion à distance de manière sécurisée à un terminal afin d'effectuer des commandes comme si on était véritablement sur le terminal de la machine distante.

Ensuite, il ne reste plus qu'à insérer la carte dans le Raspberry Pi et brancher le **câble d'alimentation**.

Pour observer le démarrage (boot), il suffit de brancher le Raspberry Pi à un écran par **câble HDMI**. Un **clavier** filaire est également indispensable.

## b. Connexion ssh

Brancher le Raspberry Pi en ethernet sur le même réseau local que votre machine (même point d'accès). C'est en général votre machine qu'il faut chercher à connecter au même réseau local que le Raspberry Pi.

La commande de connexion est : `ssh <id>@<hostname>.local`  
ou bien :`ssh <id>@<ip_address>`

Si la première ne marche pas, c'est un problème lié au DNS.

Dans ce cas, effectuer “`arp -a`” pour sonder les adresses IP (Internet Protocol) de votre réseau local.

## c. Installation de BATMAN

En cherchant des tutoriels pour installer un réseau mesh BATMAN sur des Raspberry Pi avec la requête google “batman raspberrypi”, un repository GitHub apparaît dans les premiers résultats. Il a été réalisé par un certain “binnes” et propose un tutoriel complet [4]. Le tutoriel est vraiment très bien construit et propose même de mettre en œuvre le projet d’IoT avec les capteurs DHT11 !

Il se décompose en 3 parties :

1. Création d'une passerelle (gateway) qui connecte le point d'accès du réseau domestique et le réseau mesh. Les nœuds mesh sont également configurés.

Un pont est créé pour permettre de relier le réseau mesh à des nœuds qui n'ont pas été configurés en mesh.

2. Connexion de la passerelle (gateway) en Wi-Fi au point d'accès et non plus en ethernet.
3. Projet d'IoT avec les capteurs de température DHT11.

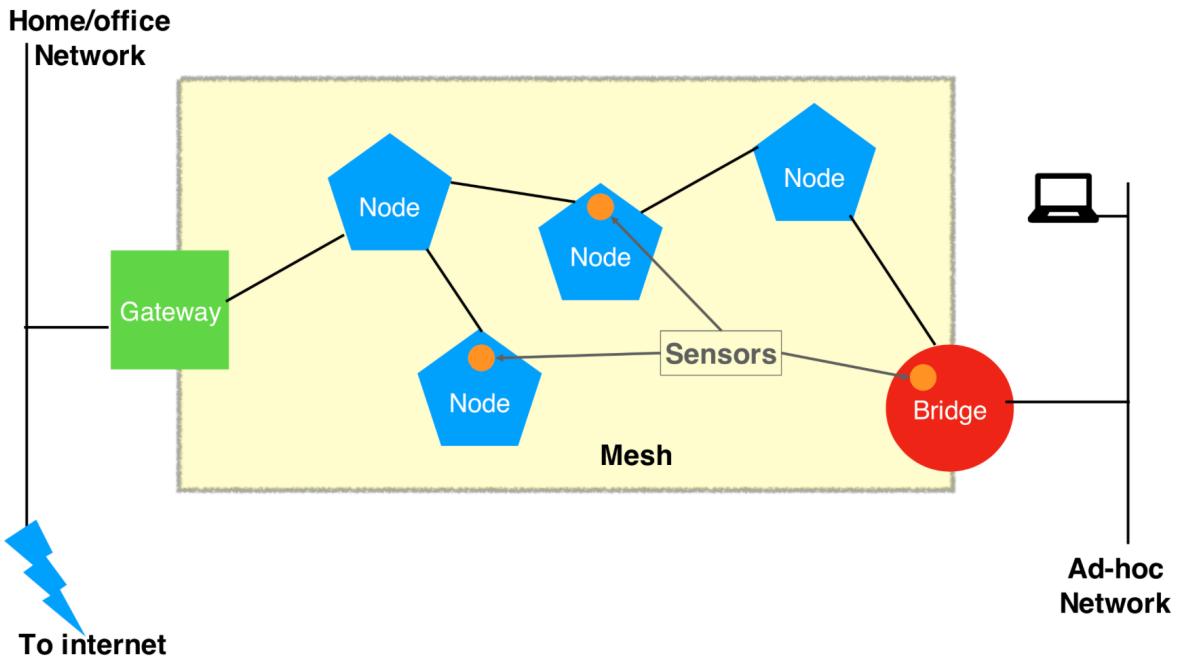


Schéma de l'architecture du réseau Mesh installé grâce au tutoriel  
(source : <https://github.com/binnes/WiFiMeshRaspberryPi/>)

Il est intéressant de faire la distinction entre des deux éléments du réseau :

- **Gateway (Passerelle)** : Une gateway est un dispositif utilisé pour connecter des réseaux distincts. Elle agit comme une interface entre deux réseaux hétérogènes, permettant à des dispositifs appartenant à des réseaux différents de communiquer entre eux. La gateway est généralement utilisée pour connecter un réseau local (LAN) à un réseau étendu (WAN), comme Internet. Elle traduit les protocoles, gère les adresses IP et facilite le transfert de données entre les réseaux. En bref, la gateway permet aux dispositifs d'un réseau de communiquer avec des dispositifs d'un autre réseau.
- **Access Point (Point d'accès)** : Un access point est un dispositif utilisé pour étendre un réseau sans fil. Il permet aux dispositifs compatibles Wi-Fi de se connecter à un réseau local (LAN) ou à un réseau étendu (WAN) sans fil. L'access point crée un point d'accès sans fil au réseau en transmettant les données entre les dispositifs sans fil et le réseau filaire. Il

agit comme un pont entre les dispositifs sans fil et le réseau câblé, permettant aux utilisateurs d'accéder au réseau et de partager des ressources.

Une gateway est utilisée pour connecter des réseaux distincts, tandis qu'un access point est utilisé pour étendre un réseau sans fil. La gateway est responsable de la connectivité entre différents réseaux, tandis que l'access point facilite la connectivité sans fil des dispositifs à un réseau existant.

**Voici les commandes effectuées pour l'installation depuis un terminal connecté au même réseau local que le noeud à configurer :**

```
ssh -o "UserKnownHostsFile=/dev/null" -o "StrictHostKeyChecking=no"  
pi@node6.local  
sudo apt-get update && sudo apt-get upgrade -y  
#sudo reboot -n  
sudo apt-get install -y batctl  
touch ~/start-batman-adv.sh  
nano ~/start-batman-adv.sh
```

Coller ce code dans le fichier ouvert :

```
"  
#!/bin/bash  
# batman-adv interface to use  
sudo batctl if add wlan0  
sudo ifconfig bat0 mtu 1468  
  
# Tell batman-adv this is a gateway client  
sudo batctl gw_mode client  
  
# Activates batman-adv interfaces  
sudo ifconfig wlan0 up  
sudo ifconfig bat0 up  
"  
chmod +x ~/start-batman-adv.sh  
sudo nano /etc/network/interfaces.d/wlan0
```

Coller ce code dans le fichier ouvert :

Note : l'essid, ici "raspberrypi-mesh-techlab", est à bien conserver identique pour chaque configuration

```
"  
auto wlan0  
iface wlan0 inet manual  
    wireless-channel 1  
    wireless-essid raspberrypi-mesh-techlab  
    wireless-mode ad-hoc  
"  
echo 'batman-adv' | sudo tee --append /etc/modules  
echo 'denyinterfaces wlan0' | sudo tee --append /etc/dhcpcd.conf  
sudo nano /etc/rc.local  
Insérer /home/pi/start-batman-adv.sh & avant exit 0
```

Si il s'agit d'un noeud : sudo shutdown -h now

**Pour le gateway seulement :**

```
sudo apt-get install -y dnsmasq  
sudo nano /etc/dnsmasq.conf
```

**Coller ce code dans le fichier ouvert :**

```
"  
interface=bat0  
dhcp-range=192.168.199.2,192.168.199.99,255.255.255.0,12h  
update start-batman-adv.sh  
"  
nano ~/start-batman-adv.sh
```

**Coller ce code dans le fichier ouvert :**

```
"  
#!/bin/bash  
# batman-adv interface to use  
sudo batctl if add wlan0  
sudo ifconfig bat0 mtu 1468  
  
# Tell batman-adv this is an internet gateway  
sudo batctl gw_mode server  
  
# Enable port forwarding  
sudo sysctl -w net.ipv4.ip_forward=1  
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE  
sudo iptables -A FORWARD -i eth0 -o bat0 -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT  
sudo iptables -A FORWARD -i bat0 -o eth0 -j ACCEPT  
  
# Activates batman-adv interfaces  
sudo ifconfig wlan0 up  
sudo ifconfig bat0 up  
sudo ifconfig bat0 192.168.199.1/24  
"  
sudo shutdown -h now
```

**Tests après installation**

```
ifconfig ⇒ bat0 présent  
iwconfig ⇒ essid configuré dans wlan0  
sudo batctl if ⇒ wlan0 active  
sudo batctl n ⇒ les voisins configurés apparaissent
```

Remplacer le nom des adresses MAC des voisins par un nom personnalisé

```
sudo nano /etc/bat-hosts  
sudo touch /etc/bat-hosts
```

**Coller un contenu de la forme :**

```
"  
MAC_ad_1 Name1  
MAC_ad_2 Name2  
...  
"
```

## d. Test de performance

Voici les principales commandes à exécuter pour tester les performances du réseau installé dans la partie précédente.

- “ifconfig”, “netstat”
- “nload” permet de voir l’activité (débit) sur chaque configuration indiquée dans ifconfig
- “nmap”
- “arp -a”
- “iperf -s” lance le serveur puis “iperf -c <ip\_address> -p <server\_port>”  
Affiche le délai d’émission/réception d’un point vers un autre, le débit et la quantité de donnée envoyée pour le test
- “traceroute <ip\_address>”  
Affiche la route empruntée par les paquets et les latences pour chaque paquet.  
L'affichage ressemble à :

```
traceroute to <dest> (<dest_IP>), <max_hops> hops max, <packet_size> byte packets
1 <router1_IP> <time1> ms <time2> ms <time3> ms
2 <router2_IP> <time1> ms <time2> ms <time3> ms
```

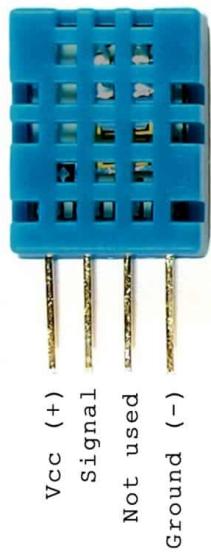
Il y a un temps de latence <time> pour chaque paquet testé.

- “ping <ip\_address>”  
Affiche le temps de latence. Les paquet envoyés sont de la forme ICMP (Internet Control Message Protocol)

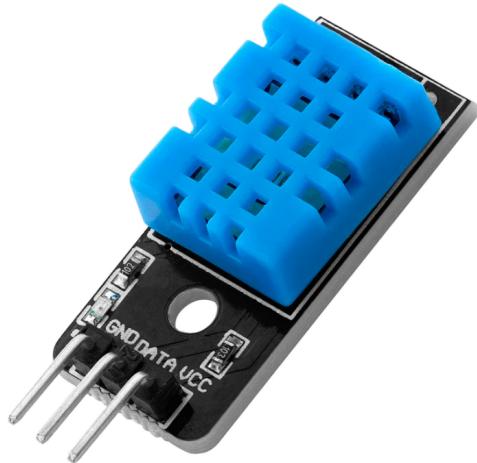
Il est intéressant de réaliser ces tests lorsque le réseau est pleinement configuré. Il est possible de réaliser des simulations de pannes en éteignant certains nœuds pour confirmer le caractère mesh du réseau. En effet, le routage doit réarranger les routes de connexions entre les nœuds et cela sera visible avec la commande “traceroute”.

## 6. Capteur de température/humidité DHT11

Le capteur DHT11 sert à mesurer la température et l’humidité. Il est compatible avec des Raspberry Pi et des Arduino. Il faut noter que la correspondance est légèrement différente sur notre modèle par rapport au schéma de montage. Notre modèle correspond à l'image de droite où le capteur est monté sur un plaque donnant lieu à 3 pattes. **Attention l'ordre n'est pas le même.**



Capteur à 4 pattes du schéma de montage



Capteur KY-015 DHT 11 az delivery

Voici le schéma de montage qui a été appliqué :

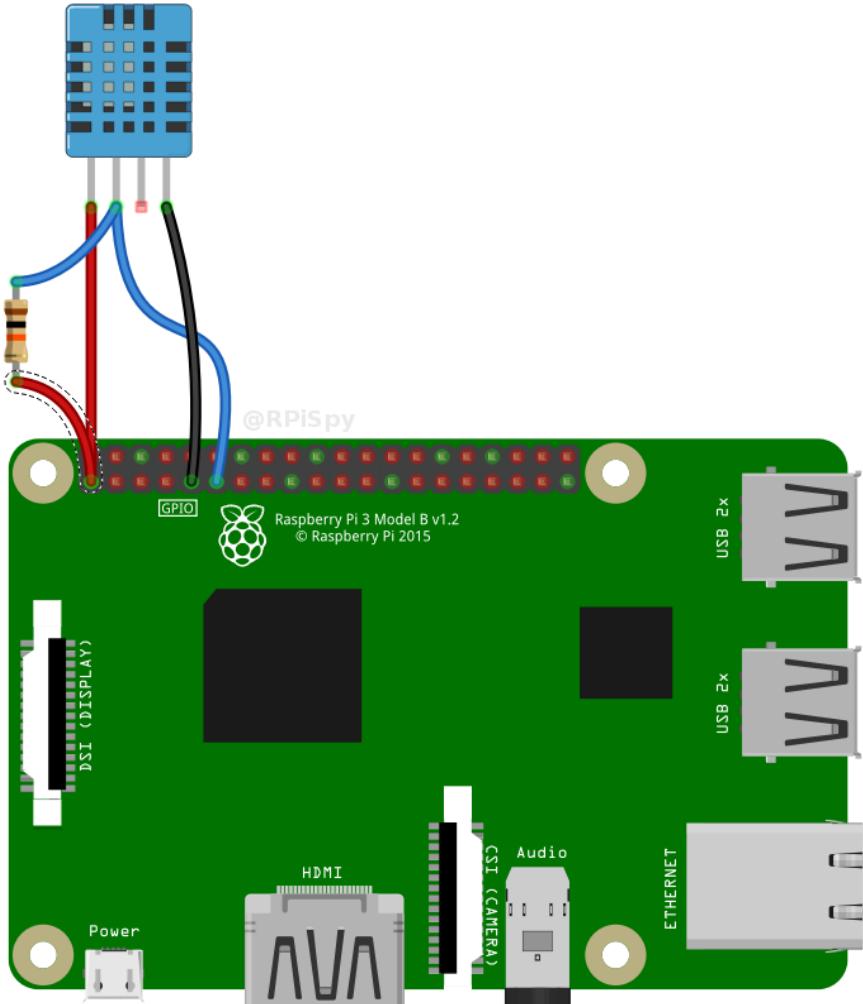


Schéma de montage de capteur DHT11 sur Raspberry Pi 3 B+ 2017  
 (source : [4])

Les données du capteur sont accessibles en programmant en python. Il suffit d'utiliser les bibliothèques de *Adafruit\_CircuitPython\_DHT*.

Pour l'installation et l'import de la librairie *adafruit\_dht*, il faut suivre les indications du repository GitHub officiel [https://github.com/adafruit/Adafruit\\_CircuitPython\\_DHT](https://github.com/adafruit/Adafruit_CircuitPython_DHT).

Voici les commandes à exécuter :

```
sudo apt-get update && sudo apt-get upgrade -y
sudo apt install python3-pip
pip install --upgrade pip
sudo apt install libgpiod2
sudo pip3 install adafruit-circuitpython-dht
```

Veillez bien à utiliser python3. Pour tester sa bonne installation :

```
python3 -V  
which python3  
pip -V  
which pip
```

Si jamais vous voulez utiliser un environnement virtuel de python, il peut être intéressant de travailler avec le gestionnaire de versions *virtualenv*.

```
"pip install virtualenv"  
"cd <dir>"  
"virtualenv <name_file_to_place_env>" (souvent <name_file_to_place_env>=venv)  
"source <name_file_to_place_env>/bin/activate"
```

Vérifier la bonne configuration avec :

```
"which python3"
```

Voici un script qui permet d'envoyer les données du capteur sur notre base de données supabase :

```
from supabase import create_client  
  
import time  
  
import board  
  
import adafruit_dht  
  
dhtDevice = adafruit_dht.DHT11(board.D17)  
  
url = SUPABASE_URL  
  
key = SUPABASE_KEY  
  
supabase = create_client(url, key)  
  
sensor_name = "nodeTom"  
  
def publish_sensor_data(temperature, humidity, sensor_name):  
  
    data = {"temperature": temperature,  
            'humidity': humidity, 'sensor_name': sensor_name}  
  
    supabase.table('measures').insert(data).execute()  
  
while True:  
  
    try:  
  
        temperature_c = dhtDevice.temperature  
  
        humidity = dhtDevice.humidity  
  
        publish_sensor_data(temperature_c, humidity, sensor_name)  
  
        print("temp", temperature_c, "humidity", humidity)  
  
    except RuntimeError as error:  
  
        # Errors happen fairly often, DHT's are hard to read, just keep going
```

```

print(error.args[0])
time.sleep(2.0)
continue

except Exception as error:
    dhtDevice.exit()
    raise error

time.sleep(15.0)

```

Pour un premier test, essayez déjà d'afficher dans le terminal le résultat toutes les 2 s.

## 7. Installation finale

Voici finalement la “valise de déploiement” que nous proposons. Le schéma ci-dessous rend compte de l’architecture globale du réseau mis en place. Ce réseau est le support de l’échange de données entre des appareils IoT (Internet of Things) pouvant être des capteurs, comme réalisé dans ce projet, ou des applications en robotique par exemple.

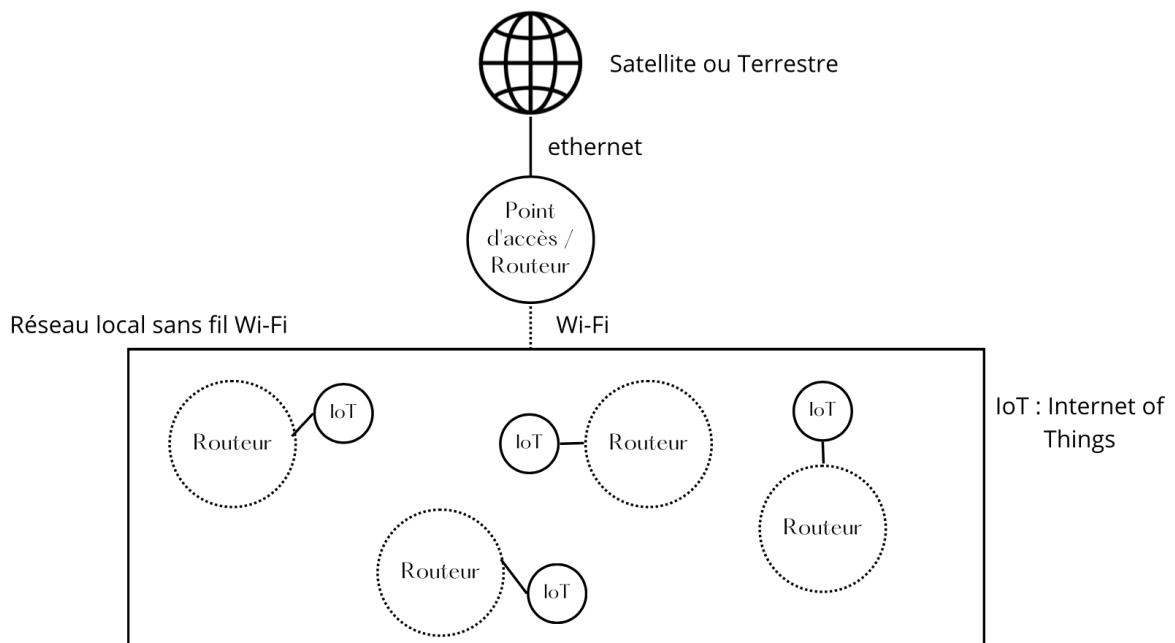


Schéma de l’installation de la “valise de déploiement”

Notre projet IoT consiste à placer une assemblée de capteurs de températures / humidité dans l’école chacun connecté à des mini-ordinateurs Raspberry Pi faisant aussi office de routeur. Les données des capteurs sont envoyées sur une base de données et affichées sur un site internet. Le but est de pouvoir consulter les mesures de température et d’humidité à distance. L’intérêt de toute cette installation est que les capteurs sont facilement mobiles

grâce à l'aspect embarqué des Raspberry Pi. De plus, le réseau possédant un routage mesh, cela permet de configurer très facilement, de disposer les noeuds facilement dans l'espace souhaité, d'étendre la couverture car tous les noeuds peuvent être routeurs dans un réseau mesh et de rendre le réseau résilient et auto-soignant.

Le réseau est très facilement configurable et étendable par ajout de nœud car le protocole BATMAN intègre automatiquement un nouveau nœud dans le réseau une fois ce nœud configuré.

Les nœuds peuvent être très facilement disposables dans l'espace. En effet, chaque nœud du réseau étant potentiellement routeur, il est beaucoup plus facile de trouver une route pour transmettre les paquets. Il n'y pas seulement un routeur central, tout le réseau peut effectuer du routage. Cet avantage se fait bien entendu au détriment de la performance des routeurs. En fait, les nœuds non performants ne deviennent routeurs que lorsque c'est vraiment nécessaire.

La couverture du réseau se trouve facilement étendable dans le sens où il n'y pas a nécessairement besoin d'augmenter la puissance d'émission des signaux des routeurs pour transmettre des données sur de plus longues distances. Il suffit de placer des routeurs intermédiaires facilement intégrables au réseau. Bien sûr, cela peut ne pas être la bonne solution si ces routeurs intermédiaires sont là uniquement pour répondre à un besoin de réseau. Le but est toujours d'apporter une solution dimensionnée et adaptée au problème. Dans notre cas d'application avec les capteurs de température, il peut sembler pertinent de créer plusieurs points d'accès à des endroits stratégiques dans l'école. Si cela n'est pas réalisé, alors on risque fortement d'engorger le réseau pour que les informations rejoignent un seul point d'accès.

Le réseau est résilient et auto-soignant dans le sens où si un nœud tombe en panne et qu'il sert de routeurs pour acheminer les messages d'autres nœuds alors le réseau pourra probablement trouver une autre route qui sera certes moins performante.

Il est également possible de connecter des mini-ordinateurs Raspberry Pi sur les routeurs Asus AiMesh dans un réseau parallèle au réseau BATMAN. Nous n'avons pas essayé de coupler les deux technologies mais c'est possible en paramétrant un pont (bridge). L'explication est donnée dans le tutoriel d'installation de BATMAN exposé en partie 5.c.

Quelles sont les caractéristiques de notre réseau ?

- Débit : Le débit est limité au débit maximal des Raspberry Pi, c'est à dire environ 100 Mb/s.  
On peut vérifier cela avec la commande “iperf”.  
Le débit dépendra bien sûr de la route considérée
- Latence : On peut évaluer la latence avec la commande “iperf” également.

- Résilience : Le réseau est résilient dans le sens où il y a un réarrangement des routes en cas de panne d'un nœud. On teste cette capacité avec la commande "traceroute".

Malheureusement, cela n'affiche pas les routes intermédiaires mais seulement le terminal de destination. Il faudrait chercher autour du protocole ICMP

- Fiabilité : Le taux d'erreur de transmission de paquets peut être évalué avec la commande ""

Concernant le point d'accès à internet, **l'antenne satellite Starlink** permet effectivement de fournir un connexion stable et performante. Les conditions météorologiques doivent être dégagées.



Photo de l'installation de l'antenne Starlink et son routeur

Voici les performances de l'antenne Starlink Standard mesurées avec nperf sur smartphone iPhone 8 :

- Débit de téléchargement maximum et moyen : 265 Mb/s et 204 Mb/s
- Débit d'envoi maximum et moyen : 19 Mb/s et 12 Mb/s
- Latence minimale et moyenne : 41 ms et 48 ms

Cela correspond à des gammes de bande passante et de latence qu'on pourrait avoir dans un réseau domestique classique.

Cette antenne pourrait bien servir à proposer un accès à internet dans des zones mal couvertes du type forêts ou montagnes. Elle est simple à mettre en place et ne nécessite que quelques minutes pour trouver le satellite. L'antenne est motorisée et peut donc s'orienter dans la bonne direction.

## 8. Conclusion et perspectives

### a. Le projet

Ce projet se concrétise par la création d'une valise de déploiement d'une assemblée de capteurs montés sur un réseau Wi-Fi mesh. L'utilité de cette solution réside dans sa capacité à fournir un réseau proportionné aux applications IoT telles que la mise en place de capteurs ou alors d'applications domotique, surveillance et en robotique. Pouvoir fournir un service adapté à chaque technologie représente l'intérêt des réseaux hybrides. Par exemple, des applications en temps réel telles que de l'analyse d'images nécessitent évidemment des performances bien plus élevées que l'exploitation de capteurs de températures. Il serait dommage de devoir surdimensionner tout le réseau pour assurer le fonctionnement des applications haute performance. Les technologies de réseau intelligentes comme le mesh permettent d'arriver à cette fin puisque les capacités de routage sont optimales pour les applications les plus gourmandes.

Une partie de notre travail n'a pas été évoquée dans ce projet mais c'est parce qu'elle ne nous a pas permis d'avoir un réseau fonctionnel. Pendant quelques séances, nous avons été épaulés par Yohan Hernandez, étudiant aux Mines de Nancy en cybersécurité et qui suit le master réseau de l'Université de Lorraine. Il a travaillé avec le chercheur Emmanuel Nataf pour proposer une installation du réseau mesh BATMAN via la "simsennet-toolbox". Malgré le temps accordé à cette partie, nous n'avons pas réussi à faire fonctionner le réseau avec cette méthode.

Il est important de noter également que notre compréhension du protocole BATMAN n'est en réalité qu'une compréhension de haut niveau à partir de nos lectures de pages web. Nous n'avons pas du tout contribué au projet batman-adv, ni même jeté un coup d'œil au code source. Cela s'explique par le fait que l'installation est déjà laborieuse pour des étudiants en 2A du département informatique de Mines Nancy et que notre objectif est surtout l'intégration de ces technologies et non leur développement. Néanmoins, comprendre le fonctionnement d'un tel projet par le code source serait intéressant et formateur.

## b. Pistes d'améliorations du projet

Il aurait pu être intéressant du point de vue de la mise à l'échelle de déployer une **solution d'orchestration** de type Kubernetes. Cela sert à installer très facilement les dépendances et de gérer les mises à jour sur tous les nœuds du réseau sans avoir à tout refaire à la main. En plus, c'est une activité très en vogue en ce moment dans les entreprises de solutions numériques qui serait formatrice, pratique et utile à la formation des étudiants. Cette solution se dirige également vers les applications type edge et fog computing dans lesquelles il faut répartir la charge de calcul entre différentes machines de manière intelligente, planifiée, orchestrée.

Les nœuds du réseau sont actuellement des Raspberry Pi mais il serait intéressant de varier le type de nœud. Par exemple, des étudiants pourraient tenter de reproduire l'installation d'un réseau mesh type BATMAN ou autre sur **ARDUINO UNO** qui sont, comme évoqué dans la partie 4.c beaucoup moins consommateurs d'énergie et plus légers. C'est une étape en plus vers l'hybridation des réseaux.

Il est primordial d'évaluer les **performances** du réseau que nous avons construit dans plusieurs configurations et cas d'usages. C'est seulement après ces retours d'expérience qu'on pourra valider la pertinence du réseau.

Le logiciel Cisco Packet Tracer permet de visualiser et de **simuler des réseaux** sur un ordinateur sans avoir besoin d'équipement physique. Par conséquent, cet outil pourrait permettre de tester des configurations réseaux. Il comporte plusieurs intérêts :

- Cela permet d'apprendre des compétences en réseaux, IoT et en Cybersécurité. Il ne dispense bien sûr pas la pratique des réseaux sur des routeurs, des switches, des firewalls et des serveurs physiques.
- Il permet aux professionnels de résoudre des problèmes virtuellement. Il sert aussi à expérimenter, gérer et sécuriser des infrastructures réseaux.

Cisco Systems propose également une plateforme Cisco Networking Academy qui permet de suivre des tutoriels et découvrir les réseaux/télécoms.

## 9. Références et bibliographie

[1] "Technologie sans fil", Farnell, 2023, url:

<https://fr.farnell.com/wireless-technology#:~:text=Un%20r%C3%A9seau%20WLAN%20permet%20de. aux%20r%C3%A9seaux%20LAN%20et%20WLAN.>

[2] "Wireless Mesh Network", Wikipedia, 2023, url:

[https://en.wikipedia.org/wiki/Wireless\\_mesh\\_network#Protocols](https://en.wikipedia.org/wiki/Wireless_mesh_network#Protocols)

[3] Raspberry Pi mesh network BATMAN, binnes, novembre 2021, url:  
<https://github.com/binnes/WiFiMeshRaspberryPi/>

[4] "Capteur de température et d'humidité DHT11 et Raspberry Pi", Tomas, 27 juin 2021  
<https://www.raspberryme.com/capteur-de-temperature-et-dhumidite-dht11-et-le-raspberry-pi>

## 10. Liste du matériel

- 11 Raspberry Pi (étiquettes : 8 nodes, 1 gateway, 1 rasp-info, 1 rasp L. Ciarletta)
- 28 micro SD avec adaptateur à chaque fois
- 2 adaptateurs microSD → USB
- 9 alim de raspberry pi
- 1 switch + alim
- Caméra pour Raspberry Pi
- Antenne Starlink
- Routeurs ASUS x10
- Câbles ethernet empruntés au techlab
- Câbles HDMI empruntés au techlab

Commandé au 2nd semestre:

- 6 batteries dont 2 grosses avec vraie prise
- 9 cartes SD Lexar
- Capteur de qualité de l'air (particules ≤ 2.5 micromètres)  
modèle : nova PM sensor SDS011  
5005-815C  
22,06,07,08