Université de Mons
Faculté des sciences
Département d'Informatique
Service de réseaux et télécommunications

Réseau Wi-Fi multi-sauts sur plateforme ESP

Rapport de projet

Directeur :
Bruno Quoitin

Auteur :
Arnaud Palgen

Année académique 2019-2020

Introduction

L'objectif du projet est de concevoir un réseau Wi-Fi multi-sauts composé de micro-contrôleurs Wi-Fi. L'ESP32 d'Espressif sera utilisé en raison de son très faible coût.
Le projet tiendra compte de la consommation énergétique des noeuds du réseau du fait qu'ils soient alimentés par batterie.
Le réseau créé sera ainsi testé pour en évaluer ses fonctionnalités. Il sera également étudié pour notamment analyser la durée du vie du réseau sur batterie, analyser la longueur des chemins établis, etc.

Table des matières

Chapitre 1

Etat de l'Art

1.1 Choix de l'environnement

Trois environnements s'offrent à nous :

1. MicroPython
   Selon le site officiel de MicroPython [2], MicroPython est une implémentation simple et efficace de Python 3. inclut un petit sous-ensemble de la bibliothèque standard Python et est optimisé pour fonctionner sur des microcontrôleurs.
   MicroPython est open source et facile à utiliser. Cependant, il n'est pas assez bas niveaux pour ce projet. Nous n'utiliserons donc pas cet environnement.
2. Arduino
3. IDF
   IDF est l'environnement du constucteur de l'ESP32 (Espressif). La documentation est complète mais le code source n'est pas disponible. Seuls les fichiers d'entêtes le sont.

Nous choisirons l'environnement IDF

1.2 Protocols de routage

Dans cette section nous discuterons des différents protocoles de routage envisageable.
Tout d'abord nous allons étudier le protocole d'Espressif, ESP-MESH . Ensuite nous étudierons et comparerons les principaux protocoles MESH de la littérature (AODV et DSR). Enfin nous pourrons choisir un protocole à implémenter sur ESP32.

Les protocoles de routages MESH peuvent être diviser en deux catégories :

1. Proactifs : Les noeuds maintiennent une/des table(s) de routage qui stockent les routes vers tous les noeuds du réseau. Ils envoient régulièrement à travers le réseau pour échanger et mettre à jour l'information de leurs voisins.
2. Réactifs : Ces protocoles établissent une route uniquement quand des paquets doivent être transférés.

Nous ne choisirons pas de protocole proactif pour se projet car comme ils gardent beaucoup d'informations en mémoire, ils passeront difficillement à l'échelle. Cependant, le temps mis pour transférer des données et plus long que les protocols proactifs, car il est nécessaire d'établir un chemin.

1.2.1 ESP MESH

ESP-MESH est le protocol du constructeur Espressif permettant d'établir un réseau mesh avec des ESP32. Cette section explique le fonctionnement de ce protocole.
La topologie ici utilisée est l'arbre. La racine de l'arbre est la seule interface entre le réseau ESP-MESH et le reste du réseau.

Construction d'un réseau

1. Élection de la racine
   - Sélection automatique
     Chaque noeud se trouvant à l'état idle va transmettre son addresse MAC et la la valeur de son RSSI avec le routeur via des beacons.
     Simultanément, chaque noeud scan les beacons des autres noeuds.
     Si un noeud en détecte un autre avec un RSSI plus fort, il va transmettre le contenu des ce beacon (càd voter pour ce noeud).
     Ce processus sera répété pendant un nombre minimum d'itérations.
     Après toutes les itérations, chaque noeud va calculer le ratio

     $$\frac{\textit{nombre de votes}}{\textit{nombre de noeuds participants à l'élection}}$$

     Si ce ratio est au dessus d'un certain seuil, ce noeud deviendra la racine.
   - Sélection par l'utilisateur
     La racine se connecte au routeur et elle ainsi que les autres noeuds, oublient le processus délection.

2. Formation de la deuxième couche
   Les noeuds dans l'état idle à portée de la racine vont s'y connecter et devenir des noeuds intermédiaires.
3. Formation des autres couches
   Les noeuds dans l'état idle à portée de noeuds intermédiaires vont s'y connecter. Si plusieurs parents sont possibles, un noeud choisira son parent selon deux critères :

   1. La couche sur laquelle se situe le candidat parent : le candidat se trouvant sur la couche la moins profonde sera choisi.

   2. Le nombre d'enfants du candidat parent : si plusieurs candidats se trouvent sur la couche la moins profonde, celui avec le moins d'enfants sera choisi.

   Une fois connecté, les noeuds deviendront des noeuds intermédiaires si le nombre maximale de couche n'est pas atteint. Sinon, les noeuds de la dernière couche deviendront automatiquement des feuilles, empêchant d'autres noeuds dans l'état idle de s'y connecter.

Pour éviter les boucles, un noeud ne va pas se connecter à un noeud dont l'adresse MAC se trouve dans sa table de routage.

Routage

1. Table de routage
   Chaque noeud possède sa table de routage. Soit p un noeud, sa table de routage contient les addresses MAC des noeuds du sous-arbre ayant p comme racine, et également celle de p.
   Elle est partitionnée en sous-tables qui correspondent au sous-arbres des enfants de p.
2. Protocole de routage
   Quand un paquet est reçu,
   — Si l'adresse MAC du paquet est dans la table de routage et si elle est différente de l'adresse du noeud l'ayant reçu, le paquet est envoyé à l'enfant correspondant à la sous-table contenant l'adresse.
   — Si l'adresse n'est pas dans la table de routage, le paquet est envoyé au parent.
   ESP-MESH utilise un mécanisme de vérification de chemin pour détecter les boucles. Si une boucle arrive, un parent va prévenir son enfant et initier une déconnexion.

Mise sous tension asynchrone
La structure du réseau peut être affectée par l'ordre dans lequel les noeuds

sont mis sous tension. Les noeuds ayant une mise en tension retardée suivront les deux règles suivantes :

1. Si une racine existe déja, le noeud ne vas pas essayer d'élir une nouvelle racine même si son RSSI avec le routeur est meilleur. Il va rejoindre le réseau comme un noeud dans l'état idle.
   Si le noeud est la racine désignée, tous les autres noeuds vont rester dans l'état idle jusqu'a ce que le noeud soit mis en tension.
2. Si le noeud devient un noeud intermédiraire, il peut devenir le meilleur parent d'un autre noeud (cet autre noeud changera donc de parent).
3. Si un noeud dans l'état idle a un parent prédéfini et que ce noeud n'est pas sous tension, il ne vas pas essayer de se connecter à un autre parent.

Défaillance d'un noeud

— Défaillance de la racine
  Si la racine tombe, les noeuds de la deuxième vont d'abord tenter de s'y reconnecter. Après plusieurs essais ayant échoué, les noeuds de la deuxième couche vont entammer entre eux, le processus d'élection d'un nouvelle racine.
  Si la racine ainsi que plusieurs couches tombent, le processus d'élection sera initialisé sur la couche la plus haute.
— Défaillance d'un noeud intermédiaire
  Si un noeud intermédiaire tombe, ses enfants vont d'abord tenter de s'y reconnecter. Après plusieurs essais ayant échoué, ils se connecteront au meilleur parent disponible.
  S'il n'y a aucun parent possible, ils se mettront dans l'état idle.

Changement de racine

Un changement de racine n'est possible que dans deux situations :

1. La racine tombe. (voir point précédent)
2. La racine le demande. Dans ce cas, un processus d'élection de racine sera initialisé. La nouvelle racine élue enverra alors une switch request à la racine actuelle qui répondra par un aquitement. Ensuite la nouvelle racine se déconnectera de son parent et se connectera au routeur. L'ancienne racine se déconnectera du routeur et rentrera dans l'état idle pour enfin se connecter à un nouveau parent.

Paquets ESP-MESH

Les paquets ESP-MESH sont contenu dans une trame wifi. Une transmission

mutli-sauts utilisera un paquet ESP-MESH transporté entre chaque noeuds par un paquet wifi différent.
La figure 1.1 montre la structure d'un paquet ESP-MESH :

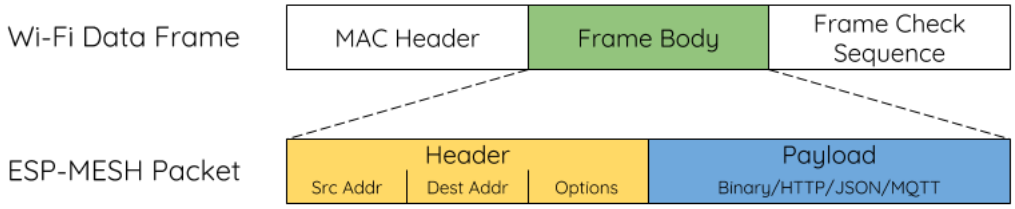

FIGURE 1.1 – Paquet ESP-MESH [1]

Le header d'un paquet ESP-MESH contient les adresses MAC source et destination ainsi que diverse options.
Le payload d'un paquet ESP-MESH contient les données de l'application.

Multicasting
Le multicasting permet d'envoyer simultanément un paquet ESP-MESHà plusieurs noeud du réseau. Le multicasting peut être réalisé en spécifiant

— Soit un ensemble d'adresses MAC
Dans ce cas, l'adresse de destination doit être `01:00:5E:xx:xx:xx`
Cela signifie que le paquet est un pquet multicast et que la liste des adresses peut être obtenue dans les options du header.
— Soit un groupe préconfiguré de noeuds
Dans ce cas, l'adresse de destination du paquet doit être l'ID [1] du groupe et un flag MESH_DATA_GROUP doit être ajouté.

Broadcasting
Le broadcasting permet de transmettre un paquet ESP-MESH à tous les noeuds du réseau. Pour éviter de gaspiller de la bande passante, ESP-MESHutilise les règles suivantes :

1. Quand un noeud intermédiare reçoit un paquet broadcast de son parent, il va le transmettre à tous ses enfants et en stocker une copie
2. Quand un noeud intermédiaire est la source d'un paquet broadcast, il va le transmettre à son parent et à ses enfants

1. Dans un réseauESP-MESH, chaque groupe a un ID unique

3. Quand un noeud intermédiaire reçoit un paquet d'un de ses enfants, il va le transmettre à ses autres enfants, son parent et en stocker une copie
4. Quand une feuille est la source d'un paquet broadcast, elle va le transmettre à son parent
5. Quand la racine est la source d'un paquet broadcast, elle va le transmettre à ses enfants
6. Quand la racine reçoit un paquet broadcast de l'un de ses enfants, elle va le transmettre à ses autres enfants et en stocker une copie
7. Quand un noeud reçoit un paquet broadcast avec son addresse MAC comme adresse source, il l'ignorer
8. Quand un noeud intermédiaire reçoit un paquet broadcast de son parent, qui a été à l'origine transmis par l'un de ses enfants, il va l'ignorer

Contrôle de flux

Pour éviter que les parents soient submergés de flux venant de leurs enfants, chaque parent va assigner une fenêtre de réception à chaque enfant. Chaque noeud enfant doit demander une une fenêtre de réception avant chaque transmission. La taille de la fenêtre peut être ajustée dynamiquement. Une transmission d'un enfant vers un parent se déroule en plusieurs étapes :

1. Le noeud enfant envoit à son parent une requête de fenêtre. Cette requête contient le numéro de séquence du paquet en attente d'envoi.
2. Le parent reçoit la requête et compare le numéro de séquence avec celui du précédent paquet envoyé par l'enfant. La comparaison est utilisée pour calculer la taille de la fenêtre qui est transmise à l'enfant.
3. L'enfant transmet le paquet en accord avec la taille de fenêtre. Une fois la fenêtre de réception utilisée, l'enfant doit renvoyer une demande de fenêtre.

Performances

Espressif fournit les performances d'ESP-MESH pour un réseau de 100 noeuds avec un nombre maximum de couches de 6 et un nombre d'enfants maximum par noeuds de 6.

Temps de construction du réseau	$<$ 60 secondes
Latence par saut	10 à 30 millisecondes
Temps de réparation du réseau	Si la racine tombe : $<$ 10 secondes Si un noeud enfant tombe : $<$ 5 secondes

TABLE 1.1 – Performances d'ESP-MESH [**?**]

Discussion
A première vue, une topologie en arbre n'est pas robuste car si la racine tombe, tout le reste du réseau est déconnecté. Cependant le processus d'élection d'une nouvelle racines semble efficace selon les résulats fournis par Espressif. Un point négatif du prorocole est que les tables de routage contiennent tous le sous arbre des noeuds. On imagine donc difficilement utiliser ce protocole pour un nombre élevé de noeuds.

1.2.2 AODV

AODV définis 3 types de messages :

1. Route request (*RREQs*)

2. Route Remplies (*RREPs*)
3. Route Errors (*RERRs*)

Ces messages sont reçus via UDP. Certains messages come les RREQ on besoin d'être diffusés dans le réseau. Dans ce cas, le TTL du header IP est utlisé pour indiquer la plage de diffusion.
Tant qu'il existe une route valide entre le noeud source et destination, AODV ne joue aucun rôle. Quand des données doivent être transmisent vers une nouvelle destination, le noeud source broadcast un RREQ pour trouver une route vers la destination. La route sera déterminée quand le RREQ atteindra la destination ou quand il atteindra un noeud intermédiaire avec une route assez "fraîche" pour la destination. On entend par route assez "fraîche", une route valide pour la destination qui a un numéro de séquence plus grand ou égal à celui contenu dans le RREQ. La route est rendu disponible en transmettant en unicast un RREP à la source du RREQ.
Chaque noeud qui reçoit un RREQ garde en mémoire la route vers la source de ce RREQ. De cette manière, les RREP peuvent être transmis par unicast. Les noeuds surveillent le status du lien avec les next-hops des routes actives. Quand une rupture d'un lien d'une route active est détectée, un RERR est utilisé pour notifié les autres noeuds de la perte de ce lien.
Un RER indique quelles destinations ne sont plus atteignables par ce lien.

Pour ce faire, chaque noeud maintient une liste de ces précurseurs qui contient les adresses IP de ses voisins qui peuvent être utlisés comme next-hop pour une destination.

1.2.3 DSR

work in progress...

1.3 Limitations

work in progress...

1.3.1 ESP NOW

work in progress...

Bibliographie

[1] ESP-MESH api guide. https://docs.espressif.com/projects/esp-idf/en/latest/api-guides/mesh.html/, 2018. [Accès en ligne le 11 décembre 2019].

[2] MicroPython. https://micropython.org/, 2018. [Accès en ligne le 11 décembre 2019].