Rapport de Projet

**P3 : Sécurisation de l'Internet des Objets par surveillance globale**



Ji Yang

Encadrants : Thomas Vantroys
Xavier Redon
Alexandre Boé

# Contexte

## La description

Le développement de l'Internet des Objets amène de nouveaux problèmes en terme de sécurité. En effet, ces objets, en général petits, n'embarquent que très peu d'éléments de sécurisation. Par ailleurs, leur mise à jour est souvent très difficile voire impossible. Un défaut de sécurité sur ce type d'objets permet à des utilisateurs malveillants de réaliser des attaques de grande envergure, du fait du grand nombre d'objets (par exemple ici).

Comme il est difficile de sécuriser l'objet par lui même (coût élevé, performances des objets limitées, ...), il est nécessaire de trouver d'autres solutions. Pour cela, nous proposons de développer :

1.une plateforme d'écoute sur les bandes ISM 868 MHZ, 2,4 GHz essentiellement ;

2.un "modèle" normal correspondant à un trafic de données non malveillant ;

3. différents scénarios d'attaques et leurs signatures associées.

## Objectifs

Le but du projet est à réaliser un système permettant de récupérer et analyser les package sous le protocole zigbee. Après l’analyse, il peut indiquer si le réseau est sous l'attaque et associer le scénario et l'attaque. Le projet a pour but de construire un réseau d’objet’, attaquer  le réseau et récupérer et surveiller le réseau. Je chois le zigbee comme le protocole de réseau à construire le réseau d'objet.

Le projet peut être séparer à des parties suivantes.

1. Construire la communication avec quelques arduino et Xbee par le protocole zigbee.

2. Faire communiquer les arduino et récupérer les messages.

3. Attaquer le réseau par les manière : 'DDos' , 'Replay'.

4. Analyser les messages et chercher les caractères de la communication sans attaque et la communication sous attaque.

5. Selon les analyses, donner une solution à indiquer si le réseau subit l'attaque.

6. Associer les attaques et les scénarios

## Matériaux

Pour le projet, j’ai utilisé :

4 arduino

4 Xbee


1 Xbee adapteur

4 Arduino_xbee_shield

1 telosb

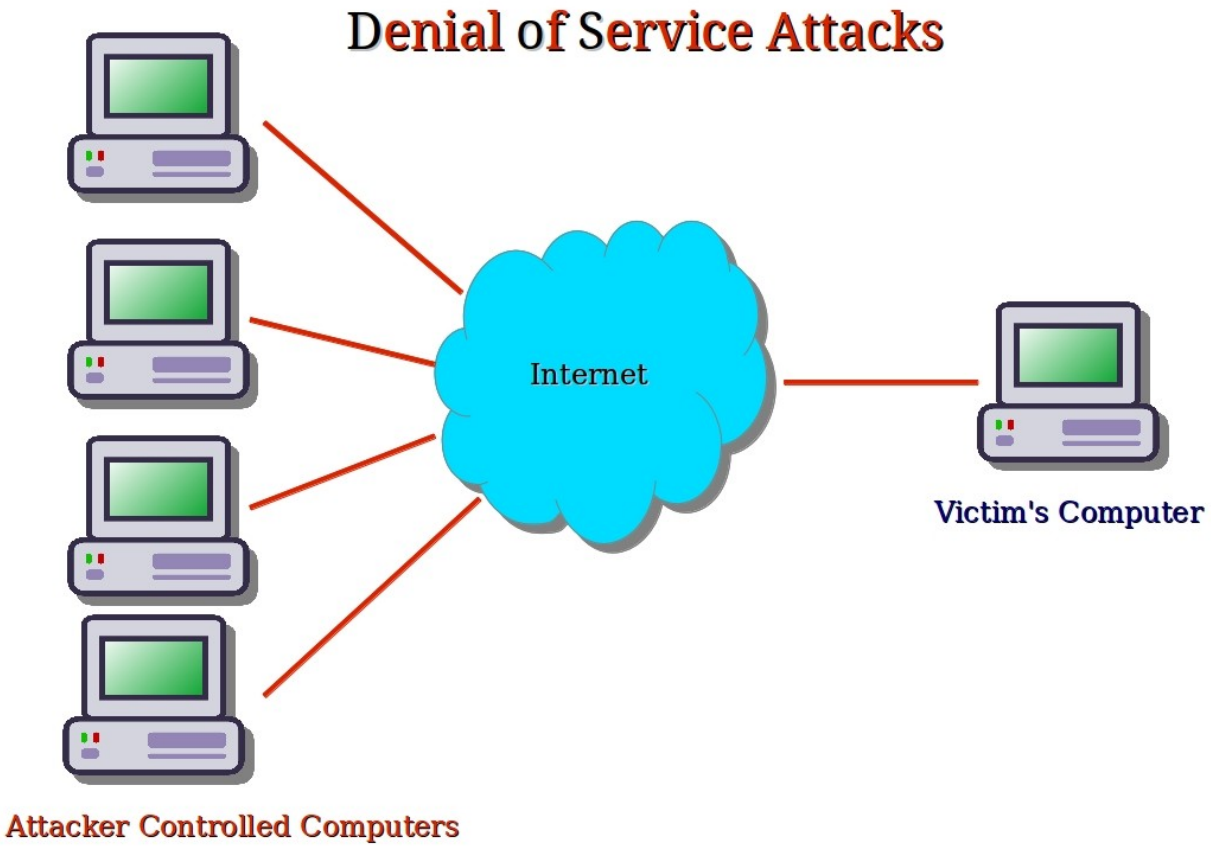

jtag
user button
reset button
TSR photodiode
PAR photodiode
SHT11 humidity / temp
6 pin expansion
10 pin expansion
LEDs
USB-serial
reset support
(bottom)
TI MSP430 F1611
ST M25P80 flash
serial ID
CC2420
IEEE 802.15.4 radio
PIFA Antenna
Telos

# Background

## Attaque DOS[1]

Une attaque par déni de service (abr. DoS attack pour Denial of Service attack en anglais) est une attaque informatique ayant pour but de rendre indisponible un service, d'empêcher les utilisateurs légitimes d'un service de l'utiliser. À l'heure actuelle la grande majorité de ces attaques se font à partir de plusieurs sources, on parle alors d'attaque par déni de service distribuée (abr.DdoS attack pour Distributed Denial of Service attack).

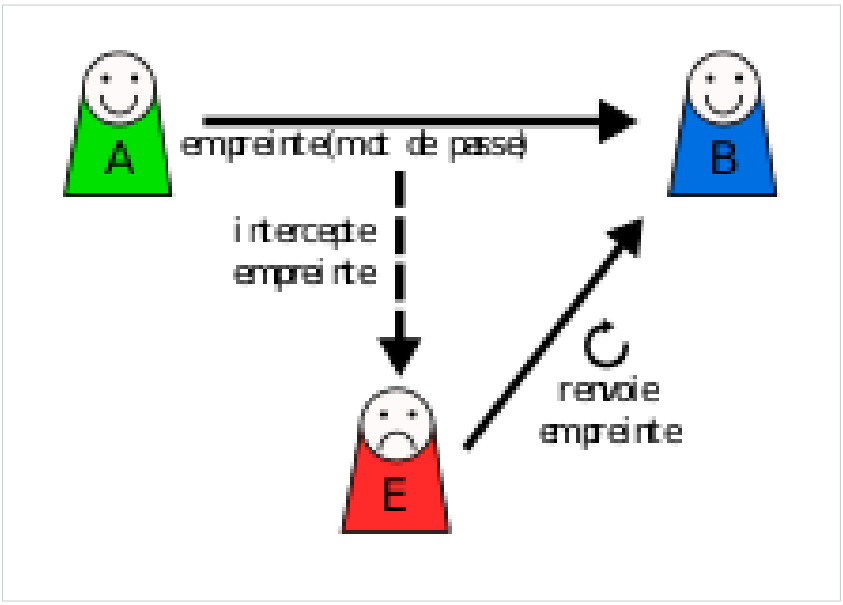
En général, la caractère la plus grande est que dans un court temps, il y a beaucoup plus de packages envoyés que au cas pas d'attaque.




Au début Dos est juste pour les réseau de l'ordinateur, mais maintenant de plus en plus des attaques Dos sont utilisé contre le réseau d'objet. Une attaque célèbre est l'attaque de 'Mirai'[2].

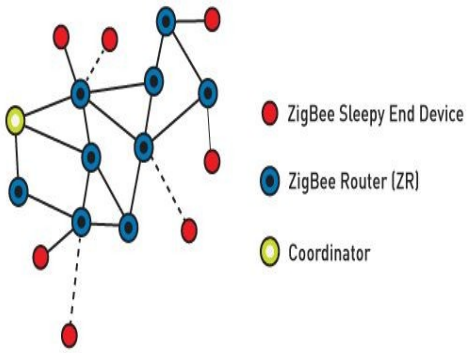
## Attaque par rejeu (Replay attack)[3]

Une attaque par rejeu (en anglais, replay attack ou playback attack) est une forme d'attaque réseau dans laquelle une transmission est malicieusement répétée par un attaquant qui a intercepté la transmission. Il s'agit d'un type d'usurpation d'identité.

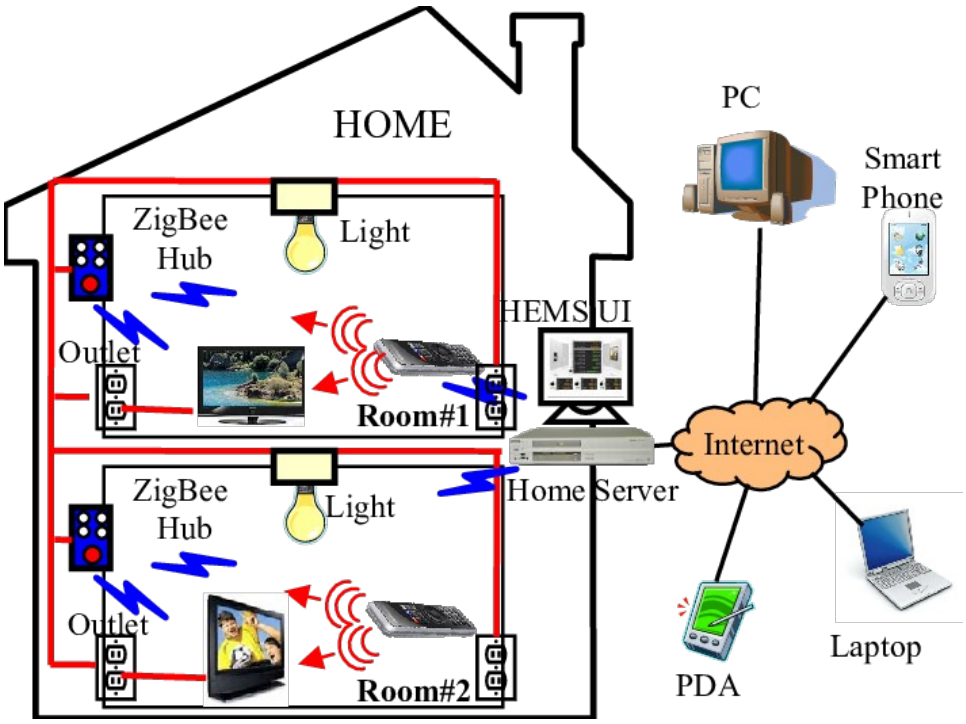



## Zigbee[4]

ZigBee est un protocole de haut niveau permettant la communication d'équipements personnels ou domestiques équipés de petits émetteurs radios à faible consommation ; il est basé sur la norme IEEE 802.15.4pour les réseaux à dimension personnelle (Wireless Personal Area Networks : WPAN).




Le plus grande avantage du Zigbee est dont puissance est petite. Ça va sauver beaucoup d’énergie, et il a haut niveau de écurité. Maintenant Zigbee est un des plus importants et les plus utilisé protocole du réseau d’objet.

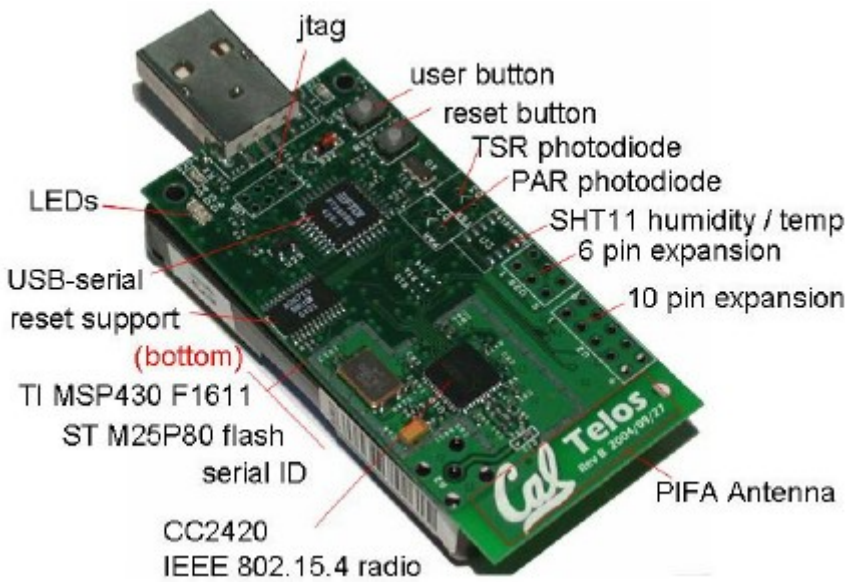



## Xbee

Xbee est un appareil de l'entreprise Digi, nous pouvons l'utiliser à envoyer et récupérer des package sous le protocole Zigbee, il a 2 modes, AT et API. Quand on utile Xbee en mode AT, nous pouvons recevoir juste de l'autre Xbee dont Chanel, PIND, et l'adresse de réseau sont tous correspondant à lui, c'est à dire que une Xbee ne puet recevoir que le message d'une Xbee dont adresse de destination est même que son propre adresse. La mode AT est souvent utilisé à la communication 'un point à un point', par contre, la mode API est souvent utilisé à la communication 'multiple point à multiple point', est les message de la mode API est plus facile à traité par l'ordinateur. J'ai 4 arduinos, donc je chois la mode API à communiquer.

## Telosb

Telosb est un apparail de Old Crossbow. Il est Composé de plusieurs capteurs, pour mon projet, j'ai l'utiliser à récupérer tous les package dans le réseau.




## Killerbee[5]

Killerbee est un cadre et un outil à attaquer Zigbee et réseau d'IEEE 802.15.4. Il donne plusieurs fonctions,  par example, sniffer, Ddos, Attaque par rejeu obtenir le mot de passe etc.

# Avantage

Il y a des produits à surveiller le réseau d’objet, par exemple, Forescout[6]et Bitdefender[7]. Ils utilisent quelques astuces pour identifier rapidement l’utilisateur, le propriétaire, le système d’exploitation, la configuration de l’appareil, les logiciels, les services, l’état des patchs et la présence d’agents de sécurité. Leur produits peuvent savoir si la périphérie est dangereux selon leur base de donnée des caractères des attaques du IOT. Ils peuvent aussi offrir les service d'afficher tous les information de périphérie connectés au réseau pour aider leur client à observer et contrôler le réseau. Leur produit est bon, mais ils ne peuvent supporter que les protocoles: wifi , bluetooch, 4G, Ethernet, IP etc.

Par contre,rapport à mon projet, mon produit a pour le but de la sécurité de IOT dont protocole est Zigbee et il surveille le réseau de IOT en à deux aspects. Un est la surveillance du contenu de messages envoyé dans le réseau. L'autre est la surveillance de la caractère physique -- puissance du chaîne.

ET il est facile à utiliser. Juste insérer le telosb à l’ordinateur ou rasperry et lancer un script de python.

# Evolution du projet

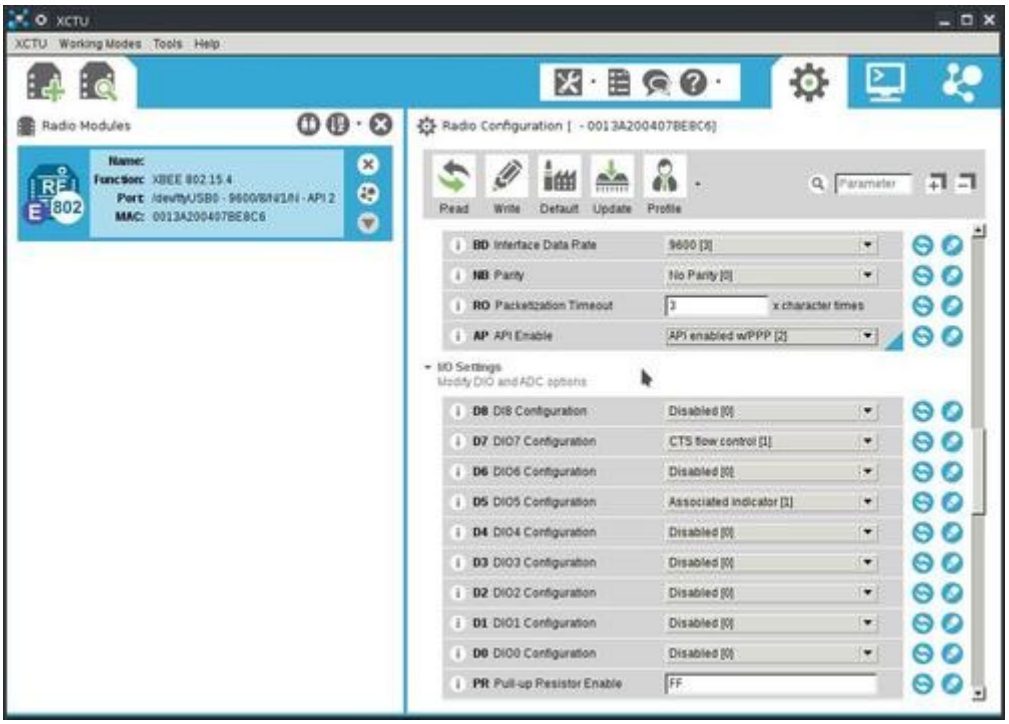
## Organisation du projet

J’ai fait seul le projet. Tous les documents, wiki, programmations sont fait par moi. Mais j’ai discuté avec professeur et les autres camarades, même si on était pas au même groupe.

## construire une communication du réseau d'objet

### *Tester les Xbees*

J’ai utilisé le logiciel XCTU[8] à tester les xbees. j’ai configuré leur adresses de source, leur adresses de destination , PIND, channel leur mode de communication(AT et API) etc. , et j’ai observé que ils peuvent communiquer bien à la manière API, avec XCTU.




Vous pouvez trouvé le vidéo de la teste ici[10].

### *Programmation à arduino*

J’ai utiliser langue C à coder la programmation. La communication est simple, mais practical: il y a 3 arduino(on va s'appeler A,B,C à la suite) avec xbee dans le réseau, dont 3 se communiquent, une est comme le coordinateur, les autres 2 sont comme ‘end_point devices’. Le fonctionnement est: 1. A envoie un message 'allumer B' à B, et reste allumé 3 seconds, après il éteint 3 seconds, et envoie un message 'allumer B' à B. 2. Quand B récupère 'allumer B' de A, il va allumer le Led, et après, envoyer 'allumer C à C. Quand B récupère 'éteindre B' de A, il va éteindre le Led, et après, envoyer 'éteindre C à C. 3. Quand c récupère 'allumer c' de B, il va allumer le Led. Quand c récupère 'éteindre c’ de B, il va éteindre le Led, et après, envoyer 'éteindre C à C.

Donc le période de allumer-éteindre est 6 secondes.

Pour contrôler le Xbee, j’ai utiliser un packages Xbee-Arduino[9] . Avec le package, nous pouvons traiter la communication facilement. Par exemple,

Le code de récupère les packages est suivant :

```
if (xbee.readPacket(500)) {
    // got a response!
    // should be a znet tx status
    if (xbee.getResponse().getApiId() == TX_STATUS_RESPONSE) {
        xbee.getResponse().getTxStatusResponse(txStatus);
        // get the delivery status, the fifth byte
        if (txStatus.getStatus() == SUCCESS) {
            // success.  time to celebrate
        } else {
        }
    }
} else if (xbee.getResponse().isError()) {
} else {
}
```

envoyer un packages :

```
xbee.send(tx);
```

```
// after sending a tx request, we expect a status response
// wait up to 5 seconds for the status response
if (xbee.readPacket(500)) {
    // got a response!
    if (xbee.getResponse().getApiId() == TX_STATUS_RESPONSE) {
        xbee.getResponse().getTxStatusResponse(txStatus);
        // get the delivery status, the fifth byte
        if (txStatus.getStatus() == SUCCESS) {
            // success.  time to celebrate
        } else {
        }
    }
} else if (xbee.getResponse().isError()) {
} else {
}
```

Vous pouvez trouvé tous mes programmation à la fin de papier.

Et une vidéo de démonstration ici[11].

### *Etudier à attaquer le réseau d'objet*

J'ai chercher les arts d'attaquer le IOT avec DDos, et je vais attaquer le IOT par 2 façon, le premier est d'utiliser un arduino à envoyer très fréquemment les trames à un arduino de le IOT, et tester si le IOT se communique normalement, si non, de quelle fréquence, elle n'est pas fonctionne normalement. L'autre façon est d'envoyer les trame avec très grande de taille, et tester si le IOT se communique normalement, si non, à quel taille de trame, elle n'est pas fonctionne normalement.

Pour le réaliser, je trouve 2 manière. La première est à utiliser le logiciel XCTU qui nous onne le API à controler le trame, la destination, la fréquence etc. Le deuxième manière est à utiliser le

package[8] de python de Xbee Nous pouvons écrire la programmation à contrôler les trame à envoyer, récupérer ou changer les paramètres.

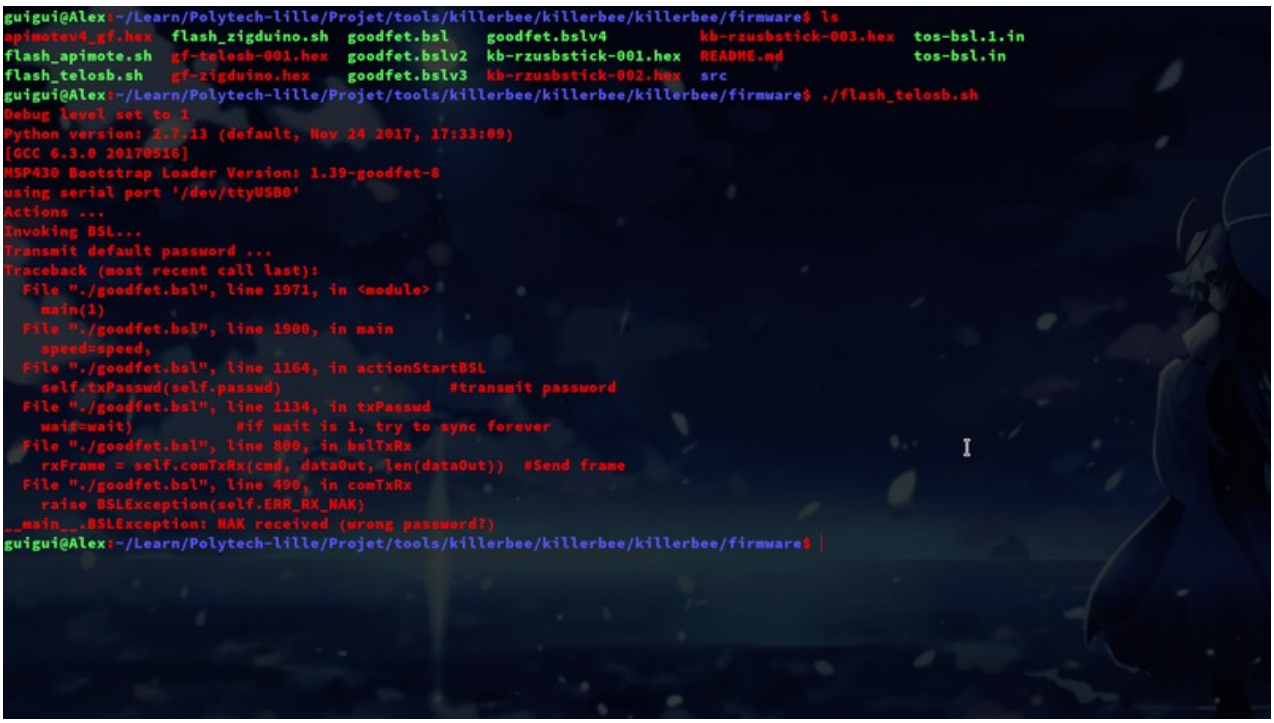
Au début, j'ai pensé que si j'allais envoyer très fréquemment  packages au réseau, la communication va être dérangé, mais en fait, la communication d'IOT a fonctionné bien. Parce que la fréquence de allumer-étendre de led était aussi 6 secondes.

Je pense que c'est parce que le débit de Xbee est petit, donc , l'influence de Ddos n'est pas très évidente.

### *Etudier le telosb*

Pour surveiller le réseau, c'est nécessaire à  trouver une manière à 'sniffer' les trames de communication de IOT. Au début, je voulais utiliser Xbee comme le 'sniffeur', j'ai trouvé un package de Xbee qui est écrit de Python[6]. Il donne des fonctions, par exemple, avec lui, on peut configurer les paramètres de Xbee(adresse de réseau, PIND, Channel, mot de passe etc, le changement de mode API et AP, etc), afficher tous les appareils dans un réseau et obtenir leur configuration, et envoyer et recevoir les message entre le réseau. Mais, quand je l'ai utilisé, j'ai trouver un problème que chaque Xbee peut recevoir seulement les trames dont destination est lui. Après, j'ai trouvé une solution que j'ai afficher tous les adresse de appareil du réseau, et j'ai changé continuellement l'adresse de source de mon Xbee aux adresse et récupérer les messages à l'adresse, et après, je peux sniffer le réseau. Mais, quand je l'ai fait, j'ai trouvé un problème que quand le Xbee a changé son propre adresse, il a pris près que 0.3 second qui a causé la perte de message. Donc ça n'a fonctionné pas. Après, j'ai utiliser telosb à sniffer le réseau.

Pour utilier le telosb, le première chose qu'on va faire est à télécharger le firmware de killerbee[5], mais il a échoué:




Après, j'ai cherché sur internet et j'ai trouvé que le problème était à cause de la version de goodfet. donc j'ai changé le fichier de goodfet et il a marché, quelques détail est sur la site suivante[9]

Et j'ai essayé les packets d'outil de killerbee, par exemple, 'sniffer', 'Ddos' etc, mais, ça n'a fonctionné pas. En fait, le telosb n'a récupéré ou envoyé rien avec les outils donné. Je ne comprends pas pourquoi il n'a fonctionné pas correctement. Après, j'ai trouvé que avec le package 'goodfet', je peux sniffer le réseau. Donc j'ai écrit une programmation pour sniffer le réseau, et j'ai trouvé que le format du trame n'était pas identité que le format du trame de Xbee, donc je pense que le problème est à cause du format de trame différent.

Un exemple du format de trame de telosb est suivant:

0x17 -> length (总字节数 - 1)

0x61 -> la tête de trame(LOW)

0x88 -> la tête de trame(HIGH)

0xd -> frameID(不确定, 它随报文顺序,而增加)

0x34 -> PIND(LOW)

0x12 -> PIND(HIGH)

0x3 -> Addr16 of sender (LOW)

0x0 -> Addr16 of sender (HIGH)

0x1 -> Addr16 of receiver (LOW)

0x0 -> Addr16 of receiver (HIGH)

0xeb -> numeno des trames envoyés(LOW)

0x0 -> numeno des trames envoyés(LOW)

0x68 |-

0x65 |-

0x6c |-

0x6c |-

0x6f |- -> data

0x20 |-

0x77 |-

0x6f |-

0x72 |-

0x6c |-

0xe7 -> checkSum(LOW)

0xa9 -> checkSum(HIGH)

Pour obtenir les data correctement, j'ai écrit une classe 'parser_trame' à alalyser les trames et une classe 'Protecter' à analyser les datas et juger s'il y une attaque, dans cet parti, je vais utiliser 2 détection, un est contre Ddos, et l'autre est contre l'attaque 'répète'. Et j'ai écrit un démo, il a fonctionné pas mal.

J'ai deux algorithmes de la détection de l'attaque de Ddos.

Le premier est à compter le numéro moyen de trame dans une période, si la valeur est supérieur à la valeur moyen, c'est à dire qu'il y une attaque de Ddos. C'est logic, parce que en géneral, la charactère le plus marqué de l'attaque DDos est que le nombre du message une plus que le nombre du message en général.

Le deuxième est à mesurer la puissance du channel, je récupère la valeur de 'rssi'[https://fr.wikipedia.org/wiki/RSSI] du Xbee pour évaluer la puissance, la relation entre rssi et la puissance pour le telosb est environ: 'puissance = rssi -40'. Si la puissance obtenu est superieur à la puissance moyan sans attaque, c'est à dire qu'il y a une attaque.

Avant de mesurer, je vous donne aussi deux fonction pour évaluer la valeur du nombre de message et de la puissance moyen sans attaque.

La détection de l'attaque de 'répète' est basé en une hypothèse que pour tous mes trame, il y a une signe correspondante au temps, ainsi on tous les trames vont être diffirent même si quelsques trames ont les mème data.

Après, il récupère continuellement les trames dans le réseau, et les transforme au format de Hash et les stocker. Après chaque fois qu'il récupère un trame, il va le transformer au format de Hash et vérifier s'il il y a un valeur dont hash est mème, si oui, c'est à dire qu'il y a une attaque de 'répète'. Mais, il y a un cas spécial que un appareil veut envoyer un message à l'autre appareil, mais, il est échoué, donc il répète une fois le même data(avec même signe), en fait, c'est pas l'attaque. Donc j'ai ajouté un délais, par exemple 5s, si il y a un message dont Hash est existé dans la base de donné mais l'intervalle entre le temps ou les 2 Hash sont récupéré est intérieur à 5s, il va penser qu'il n'y a pas d'attaque.

### *Améliorer l'algorithme*

J'ai fait une démonstration à Professeur Redon et Professeur Vantroys, et Nous avons fait une communication sur le projet, Monsieur Vantroys m'a dit qu'il y a un problème à mon algorithme de la détection de Ddos par compter les numéro de message. Mon algorithme ne peut pas détection le Ddos si le nombre moyen est intérieur à la valeur de seuil, mais dans une petite durée, le nombre de message est beaucoup. En fait, c'est une attaque, mais avec mon algorithme précédant, on ne peut pas le détecter. Donc à ce semestre, je cherche une meilleure manière à détecter la Ddos. Après quelques jours de chercher, j'ai trouvé une manière de détecter Ddos qui est utiliser à détecter Ddos sur Internet. Cette manière utiliser un indicateur qui s'appelle 'Hurst exponent'[12]. L'indicateur est correspondant à auto-similarité. L'idée de l'algorithme est calculer continuellement l'indicateur, s'il y a pas d'attaque, l'indicateur va rester à un niveau et en général la valeur est entre 0.5 et 1. Mais, s'il y a une attaque, la auto-similarité va être influencé, la valeur d'indicateur va diminuer, donc on peut dire que quand l'indicateur de Hurst est intérieur à une valeur de seuil, il y a une attaque.

L'algorithme à calculer l'indicateur de Hurst est suivant(copié de Wiki[13]):

1. Calculate the mean;

$$m = \frac{1}{n}\sum_{i=1}^{n} X_i \,.$$

2. Create a mean-adjusted series;

$$Y_t = X_t - m \quad \text{for } t = 1, 2, \ldots, n\,.$$

3. Calculate the cumulative deviate series $Z$ ;

$$Z_t = \sum_{i=1}^{t} Y_i \quad \text{for } t = 1, 2, \ldots, n\,.$$

4. Compute the range $R$ ;

$$R(n) = \max(Z_1, Z_2, \ldots, Z_n) - \min(Z_1, Z_2, \ldots, Z_n)\,.$$

5. Compute the standard deviation $S$ ;

$$S(n) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - m)^2}.$$

6. Calculate the rescaled range $R(n)/S(n)$ and average over all the partial time series of length $n$.

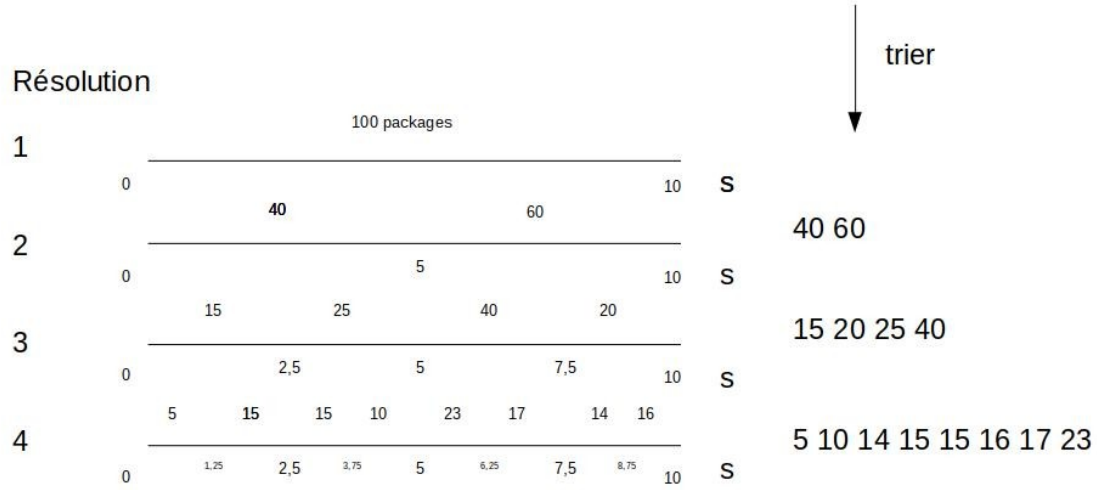
J'ai réaliser l'algorithme, et je l'ai implémenter à ma programmation, mais malheuresement ça n'a fonctionné pas. Parce que si on veut utiliser l'algorithme, c'est obiligatoire que les variance n'equale pas 0. Mais, pour mon data, il y a des situation ou la variance equale 0. Donc je ne peux pas l'utiliser.

Après, j'ai pensé une nouvelle idée que pour une liste de data, je peux les diviser à plusieur dessous-niveau, et pour chaque dessous-niveau, je mesurer leur valuer et l'utilise à juger s'il y a des attaque.

Par exemple, pour les data pendant 10s, je les diviser à 2 partie, la première partie est les data de première 5 secondes, la deuxième partie est les data de deuxième 5 secondes, et on peut le diviser encore. Après, on va avoir les nombres de message à chaque niveau et chaque segment. Après je récupère les data sans attaque, avec les data, j'utilise la loi normale évaluer la distribution du nombre de message à chaque niveau et chaque segment. Quand on fait la surveillance, ont calculer le nombre de message de chaque segment, si le nombre est supérieur à la valeur moyen du loi normale qui est correspondante à ce segment, on va calculer le quotient de la densité de probabilité à ce valeur et la densité de probabilité à la valeur moyen, après on fait la sommation tous les quotient à chaque segment dans une période(avant de la sommation, les quotient multiplie au poids qui est correspondant au niveau). A la fin, si le quotient est intérieur à un seuil, c'est à dire qu'il n'y a pas de attaque, or il y a des attaque. La semaine prochaine, je vail le réaliser. La procédure est suivant:

Préparation:

```
0. Choisir la duré de la préparation t;
1. Choisir un résolution r (par exemple 4);
2. Récuperer les package du réseau dans une périod p (par exemple 10s);
3. for i in 1 : résolution:
      Deviser le période à (2 ** (résolution - 1) ) segment;
      Calculer le nombre de package à chaque segment;
      Les trier par l'ordre ascendant;
4. Composer les data et les stocker
5. if le temps de fonctionne tt > t:
      go to 2;
   else:
      go to 6;
6. calculer la valeur moyenne 'm' et la variance 's' des data à chaque segment à chaque
résolution
```




surveiller:

```
0. Obtenir les data de préparation: t, r, m, s, choisir un seuil k;
1 Y = 0
2. for i in 1 : r:
      Deviser le période à (2 ** (résolution - 1) ) segment;
      Calculer le nombre de package à chaque segment comme x;
      Les trier par l'ordre ascendant;
```

$$yi = \frac{1}{si\sqrt{2\pi}} e^{-\frac{(x-mi)^2}{2si^2}} / \frac{1}{si\sqrt{2\pi}} e^{-\frac{(mi-mi)^2}{2si^2}} * \frac{1}{2*t},$$

```
3. calculer les indicateur :
      if x > mi:
        Y += yi;
4. if Y > k :
      print 'there is une attaque;
5. if le temps de fonctionne tt > t:
      go to 2;
   else:
      go to 6;
```

```
6. print 'no attaque'
```

En fait, ça fonctionne bien. Il peut détecter l’attaque de Dos au cas que l’assaillant envoie continuellement les packages avec la fréquence très haute, mais aussi au cas que l’assaillant  envoie beaucoup de packages nuisibles à une durée courte, comme Ldos(Low rate Denial of Service).

# Conclusion

Ce Projet étant de grande ampleur, il n’est pas terminé, il y a pas défaut, par exemple, le telosb ne peut récupérer que 80 % de package quand il y a 100 packages pendant 1 seconde, et l’algorithme peut aussi être amélioré.

Pour moi, j’ai obtenu les connaisse du Protocole Zigbee, le python, la programmation en Arduino et quelque outil d’attaquer le réseau.

# Référence


1. https://fr.wikipedia.org/wiki/Attaque_par_d%C3%A9ni_de_service
2. https://fr.wikipedia.org/wiki/Mirai_(logiciel_malveillant)
3. https://fr.wikipedia.org/wiki/Attaque_par_rejeu
4. https://fr.wikipedia.org/wiki/ZigBee
5. https://github.com/riverloopsec/killerbee
6. https://www.forescout.com/
7. https://www.bitdefender.fr/
8. https://www.digi.com/products/xbee-rf-solutions/xctu-software/xctu
9. https://github.com/andrewrapp/xbee-arduino
10. https://www.youtube.com/watch?v=uhft9i8MHr8
11. https://www.youtube.com/watch?v=LZTDQbQ1uCc
12. https://en.wikipedia.org/wiki/Hurst_exponent
13. https://en.wikipedia.org/wiki/Hurst_exponent