Master 1 Informatique

Rapport de projet TER

Amélioration des faiblesses de Bluetooth Low Energy

Quentin Guardia - Redouane Otmani - Ludivine Bonnet

Année universitaire : 2020-2021

Encadrants :Ahmed MEHAOUA - Osman SALEM

Résumé

Dans un environnement de plus en plus connecté, nous avons été amenés à nous interroger sur la sécurité des objets qui nous entourent. Et en particulier des objets connectés par Bluetooth Low Energy (BLE). Ainsi, dans le cadre du projet tutoré de Master 1, nous avons étudié la sécurité de BLE. Après avoir décortiqué les mécanismes qui se cachent derrière ce protocole, nous avons recensé les attaques connues à ce jour. Nous en avons mis une en pratique, qui consiste à sniffer des paquets échangés. Enfin, nous proposons dans ce rapport divers moyens de sécuriser les objets connectés par BLE et ainsi une partie de notre environnement.

Table des matières

Table des matières

Table des figures

Liste des tableaux

Introduction **1**

Chapitre I. Présentation de BLE

Introduction 2
I.1 Modèles de communication 2
I.2 L'advertising 2
I.3 Le protocole GATT et structures de données 4
Conclusion 5

Chapitre II. Sécurité

Introduction 6
II.1 Chiffrement 6
II.1.1 LE Legacy Connections et LE Secure Connections 6
II.2 Randomisation des adresses MAC 8
II.3 Liste blanche 8
Conclusion 8

Chapitre III. Attaques possibles et propositions de protection

Introduction 10
III.1 Attaques sur advertisement 10
III.1.1 Principe 10
III.1.2 Attaques 10
III.1.3 Contre-mesures 11
III.2 Interception passive 11
III.2.1 Principe 11
III.2.2 Attaques 11
III.2.3 Contre-mesures 12

III.3 Interception active 12
III.3.1 Principe 12
III.3.2 Attaques 13
III.3.3 Contre-mesures 14
III.4 Appareils en libre accès 14
III.4.1 Principe 14
III.4.2 Attaques 14
III.4.3 Contre-mesures 15
III.5 Attaques lors de l'appairage 15
III.5.1 Principe 15
III.5.2 Attaques 15
III.5.3 Contre-mesures 16
III.6 Contournement de la liste blanche 16
III.7 Violation de confidentialité 16
Conclusion 17

Chapitre IV. Attaque BLE par sniffing
Introduction 18
IV.1 Préparation du serveur GATT 18
IV.2 Préparation du client GATT 19
IV.3 Paramétrage de l'appareil pirate 19
IV.3.1 Drivers 19
IV.3.2 Installation de l'API AdaFruit 19
IV.3.3 Détermination du Serial Port 20
IV.3.4 Python et pySerial 20
IV.4 Interception des données (sniffing) 20
IV.4.1 Lancement du serveur GATT (Raspberry) 20
IV.4.2 Lancement de l'API (ordinateur pirate) 21
IV.4.3 Connexion du client (smartphone) 21
IV.4.4 Écoute passive du serveur (ordinateur pirate) 22
Conclusion 23

Conclusion **24**

Références **25**

Glossaire **28**

Table des figures

I.1 Communication entre Central et Peripherals 3
I.2 3 canaux utilisés pour l'advertising 3
I.3 Diagramme d'un GATT Heart Rate Service 5

III.1 MITM avec GATTacker 12

IV.1 Sniffer BlueFruit 18
IV.2 AdaFruit Sniffer API 21
IV.3 BLE Central 22
IV.4 Wireshark affichant la fréquence cardiaque 23

Liste des tableaux

II.1 Modèles d'association selon le matériel [Zhang19] 7

Introduction

Contexte et motivation

L'Internet des Objets (IoT) fait référence à la connexion des objets au réseau Internet. Les capteurs occupent une place grandissante dans notre société. Ils sont utilisés dans divers domaines, de la santé à la géolocalisation en passant par l'industrie. L'essor des objets connectés implique une adaptation des modes de transmission. C'est pourquoi certains réseaux sans fil comme LoRa ou BLE ont vu le jour. On s'attardera sur ce dernier le long du projet. Après avoir été développé par Nokia dès 2006, il s'intègre à Bluetooth 4.0 sous le nom que l'on connaît aujourd'hui, ou également sous le nom de Bluetooth Smart. Cependant, comme pour toute technologie, des failles existent. Il est de notre volonté de les comprendre pour les maîtriser.

Contributions et organisation du rapport

On présentera dans une première partie les généralités et technicités de BLE. C'est-à-dire sur quelles modalités les appareils communiquent entre eux. Puis nous étudierons dans un second temps les failles de BLE connues à ce jour, avec les implémentations de sécurisation possibles. Enfin, nous mettrons en pratique une attaque, avec des propositions de sécurisation pour terminer.

Chapitre I

Présentation de BLE

Introduction

BLE est une technologie sans fil, à basse consommation, basée sur Bluetooth, utilisée pour connecter des appareils entre eux. Il consomme environ moitié moins d'énergie que Bluetooth, car moins de données sont échangées. Aussi, les appareils peuvent rester en « sleep mode » jusqu'à la prochaine interaction. C'est pour ces avantages qu'il est utilisé par des capteurs, qui ont besoin d'une part d'envoyer peu de données, d'autre part d'une implémentation légère et une faible consommation. De plus, grâce à ces avantages, les applications utilisant BLE peuvent fonctionner sur de longues périodes, des mois, voire des années. Les standards Bluetooth sont gérés par le Bluetooth Special Interest Group (Bluetooth SIG). Ainsi, nous allons détailler comment plusieurs appareils peuvent communiquer avec cette technologie.

I.1 Modèles de communication

BLE compte deux types de modèle de connexion possible.

Maître et esclave :

Le maître, aussi appelés « Central », a pour but d'initialiser la connexion avec un ou plusieurs esclaves. À contrario, un esclave, ou « Peripheral », ne peut pas initialiser de connexion et ne peut se connecter qu'à un seul maître. Il exécute uniquement les ordres de celui-ci. Il se fait repérer grâce aux paquets advertising qu'il envoie. On associe généralement le Central à l'ordinateur ou au smartphone. Il consomme plus que le Peripheral.

Broadcaster et Observer :

Le Broadcaster n'accepte aucune connexion, et détecte les Observers, pour éventuellement envoyer des paquets advertising. L'Observer se fait, tout comme le Peripheral, détecter grâce à des paquets advertising envoyés à intervalles réguliers. Jusqu'à une demande de connexion.

I.2 L'advertising

Ainsi, l'Observer et le Peripheral arrêtent d'envoyer des paquets d'advertising lorsqu'ils reçoivent un paquet spécifique, leur indiquant qu'ils sont connectés à un Broadcaster ou un Central. D'ailleurs, on parle de mode connecté lorsque l'échange est bidirectionnel, ce qui est le cas pour le Central et l'Observer. Et de mode advertising pour les connexions à sens unique, pour le Peripheral et le Broadcaster.

Figure I.1. Communication entre Central et Peripherals

BLE fonctionne grâce à des ondes radio ultra haute fréquence, sur une bande 2.4 GHz à 2.8 GHz. Et ce avec 40 canaux physiques, contre 80 pour Bluetooth, pour un multiplexage en fréquence et en temps grâce à la couche L2CAP. Par un simple calcul, on trouve que l'écart entre deux canaux est de 2 MHz. Les appareils en mode advertising envoient des paquets de 31 bytes à intervalle régulier. Et ce uniquement sur 3 des canaux : le 37, le 38 et le 39. Les autres canaux sont réservés pour les échanges de données entre appareils.[Bluetooth]

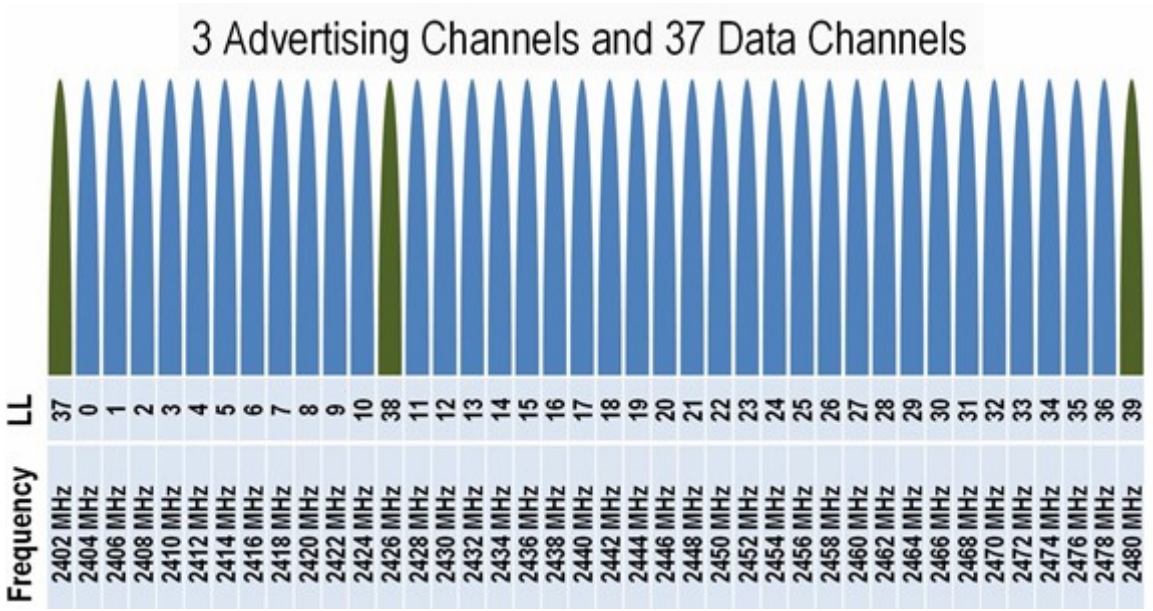

Figure I.2. 3 canaux utilisés pour l'advertising

Généralement, les appareils envoient leurs services, terme expliqué dans la partie suivante, leur nom et les informations sur le constructeur.

De son côté, le Central alterne entre scanner (écouter) les paquets d'advertising et en

envoyer. Il scanne pour savoir s'il trouve le Peripheral qu'il cherche puis il envoie afin de commencer l'échange avec le Peripheral. Le processus de scan est coûteux donc généralement le scan ne fonctionne pas indéfiniment.

I.3 Le protocole GATT et structures de données

Le protocole GATT est utilisé lors de la communication entre un maître et un esclave. Plus précisément, l'esclave est un serveur GATT, et le maître est un client GATT se connectant au serveur. Ainsi, le client fait une requête pour obtenir des données nommées attributes, au serveur. Les attributes sont sous forme de characteristics, services ou encore descripteur. Les characteristics regroupent des descripteurs. Un des descripteurs, optionnel, est la valeur finale demandée, comme le pouls. Il y a nécessairement deux autres descripteurs, un pour déclarer la characterestic et un autre pour apporter plus d'information sur la valeur envoyée, comme son unité par exemple.

Les services sont des ensembles de characteristics. Le client attend que le serveur retourne ses données complétées. Il peut s'agir d'un groupe de characteristics donnant des informations sur plusieurs paramètres du corps par exemple.
Chaque attribute est reconnu par un UUID (Universally Unique IDentifier) qui permet de définir universellement la nature de l'information ; par une valeur qui instancie le descripteur, comme l'unité du poul, les permissions éventuelles (lecture, écriture, les deux ou aucunes) et le handle, une valeur hexadécimale qui permet de reconnaître chaque attribute au sein du paquet.[Woolley16]

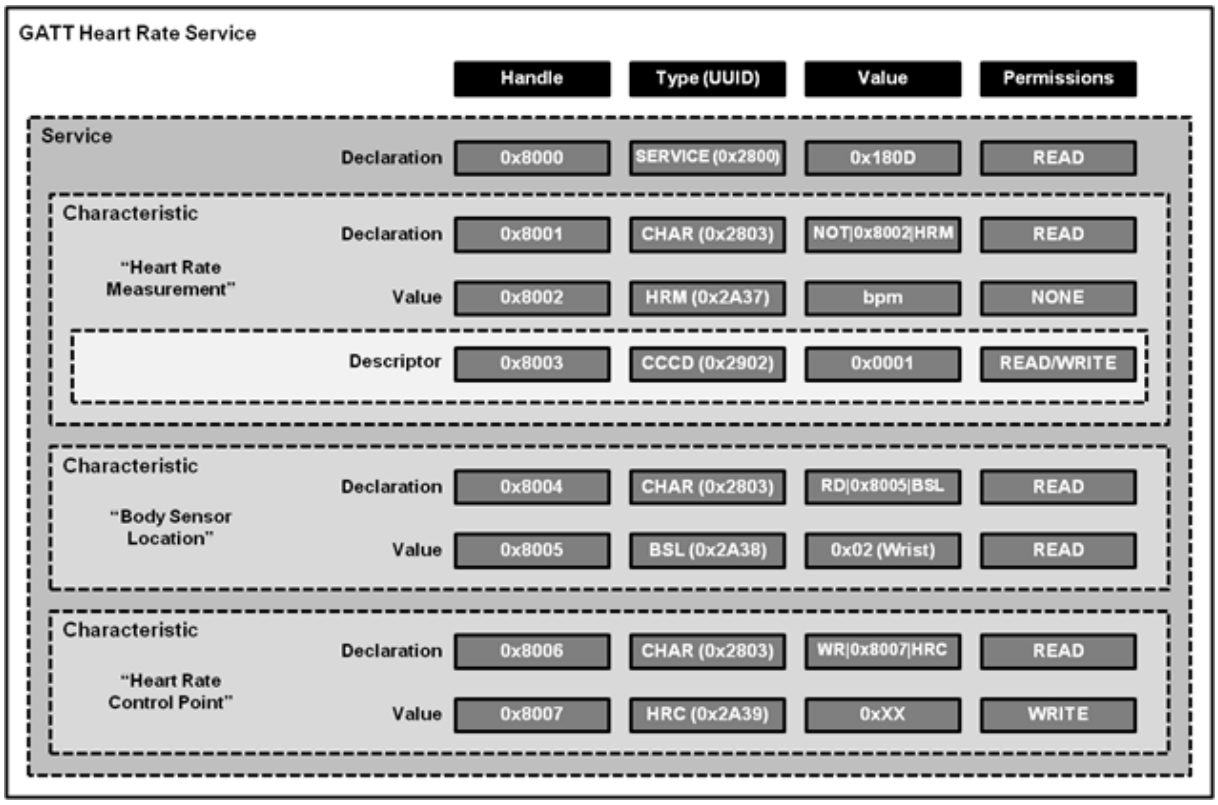

Figure I.3. Diagramme d'un GATT Heart Rate Service

Globalement, un client GATT peut, via les characteristics, faire deux types de requête : write et read. Respectivement pour obtenir confirmation ou une valeur de la part du serveur. Le client peut également avoir un droit d'écriture avec write.

De son côté, le serveur peut envoyer une notification au client, sans attendre de retour. Cependant, si un ACK est requis, alors on parlera plutôt d'indication.

Conclusion

Les appareils BLE respectent une hiérarchie. Il y a en effet un client et un serveur, un maître et un esclave. Le client est généralement le maître. En guise d'illustration, un smartphone peut se connecter en BLE à une station météo pour recevoir la température. Dans ce cas de figure, le smartphone est le maître, ou client GATT, et la station météo l'esclave, ou serveur GATT. La communication par BLE est particulière. En effet, seulement quelques variables sont échangées dans ce qu'on appelle des characteristics. Ce qui fait la légèreté de BLE. Bien sûr, BLE offre un certain niveau de sécurité. C'est ce que nous allons voir dans la partie suivante.

Chapitre II

Sécurité

Introduction

On sait qu'un utilisateur malveillant, s'il est assez proche des appareils BLE, peut intercepter certains paquets. Pour éviter des attaques, les protocoles ont été complexifiés pour assurer l'intégrité, l'authenticité et la confidentialité des données échangées. Nous présenterons ici les méthodes de chiffrement, de randomisation des adresses MAC et du fonctionnement de la liste blanche.

II.1 Chiffrement

II.1.1 LE Legacy Connections et LE Secure Connections

Pour les appareils munis de Bluetooth 4.0 et 4.1, on parle de LE Legacy Connections. L'un des appareils envoie une requête d'appairage à l'autre. De là, les appareils s'échangent leurs informations : leurs moyens d'entrées/sortie, comme la présence ou non d'un clavier, les modalités pour l'authentification du premier appairage et la liaison lors des réappairages futurs (bonding) et la taille maximale de la clé qui va être utilisée. À noter que pour l'instant, rien n'est encore chiffré.
Puis, toujours lors de l'appairage, un des appareils génère une clé temporaire (TK) qui sera sue des deux appareils. Une confirmation de la clé est faite grâce à l'échange de valeurs aléatoires, échangées et chiffrées puis déchiffrées. Avec la TK et une valeur aléatoire, une nouvelle clé est générée : la STK. La connexion sera chiffrée avec cette clé, au niveau de la couche de liaison. Éventuellement, une LTK peut être échangée pour le bonding.

Pour les appareils armés de Bluetooth 4.2 ou supérieur, qui peuvent cependant utiliser la méthode précédemment décrite, on parle plus de LE Secure Connections. Il n'y a plus de clés TK et STK, mais une clé LTK, dont le calcul est basé sur des paires de clés générées à partir d'une courbe elliptique Diffie-Hellman. Cette méthode est bien plus sûre.[Ren16]

II.1.1.1 Génération de clés pour LE Legacy Connections

- Just Works : Avec cette méthode, la TK est initialisée à 0. Il est cependant facile de trouver par force brute la STK.
- Out of Band (OOB) Pairing : La TK est échangée par une autre technologie comme NFC. Il y a ici deux avantages. D'abord, la clé échangée peut être plus longue, jusqu'à 128 bits. Ensuite, BLE et l'autre moyen de connexion sont indépendants et donc bien plus sûrs. C'est la méthode la plus fiable. Tant que le second moyen de

communication est sûr, alors tout l'échange est sûr.

- Passkey : Une chaîne de 6 chiffres est échangée par divers moyens possibles. Par exemple elle peut être affichée à l'écran et validée par les deux utilisateurs. Cette méthode est plutôt sûre. Cependant, elle s'est déjà montrée vulnérable.[Rosa13]

II.1.1.2 Génération de clés pour LE Secure Connections

- Just Works : Une fois que les appareils ont échangé leurs clés publiques, l'appareil qui n'a pas initié la connexion produit un nonce, qui est une valeur aléatoire, et l'utilise pour créer une valeur de confirmation Cb. Puis, Cb et le nonce sont envoyés à la machine qui a initié la connexion. En même temps, cette dernière produit également un nonce avant de l'envoyer. Elle utilise aussi le nonce qu'elle a reçu pour créer sa propre valeur de confirmation Ca. Si les deux valeurs correspondent, alors la connexion peut avoir lieu. La courbe elliptique Diffie Hellman apporte une sécurité supplémentaire. Cependant, cette méthode reste la plus vulnérable.
- Out of Band (OOB) Pairing : Cette fois, les clés publiques, nonces et valeurs de confirmations sont échangés par d'autres moyens que Bluetooth, comme NFC.
- Passkey : Les deux appareils utilisent les clés publiques échangées, le code à 6 chiffres affichés sur les deux écrans et les nonces de 128 bits pour authentifier la connexion. Ici, chaque bit de la passkey est échangé puis confirmé comme dans la méthode de Just Works. Une fois chaque bit mutuellement confirmé, l'échange sécurisé commence.
- Comparaison numérique : Cette méthode reprend Just Works et ajoute une autre étape. Après le Just Works vérifié, chaque machine génère 6 chiffres en utilisant les nonce. Ces 6 chiffres s'affichent alors et l'utilisateur doit vérifier qu'ils correspondent bien à la valeur attendue.

Pour récapituler, voici les modèles d'association employés selon le matériel utilisé.

A1 \ **A2**	**Écran**	**Écran, Oui/Non**	**Clavier**	**Pas de Clavier/Écran**	**Clavier, Écran**
Écran	Just works	Just works	Passkey	Just works	Just works
Clavier	Passkey	Passkey	Passkey	Just works	Passkey
Pas de clavier/Écran	Just works	Just works	Just works	Just works	Just works
Clavier/Écran	Passkey	Comparaison numérique	Passkey	Just works	Comparaison numérique

Tableau II.1. Modèles d'association selon le matériel [Zhang19]

II.2 Randomisation des adresses MAC

Comme pour les autres types de connexion, un appareil BLE est identifié grâce à une adresse MAC de 48 bits. De nos jours, quatre formats sont identifiés :

- Format public IEEE : Les 24 bits de poids forts désignent l'entreprise, et les 24 autres sont à définir par l'entreprise. Ici l'adresse ne peut pas changer, ce qui n'offre pas une sécurité maximale.
- Statique aléatoire : L'adresse MAC est mise à jour aléatoirement lors de chaque nouveau démarrage. Ainsi, une certaine protection est apportée si l'appareil est redémarré régulièrement.
- Privée résoluble aléatoire : Cette méthode est utilisée à condition qu'une Identity Resolving Key (IRK) soit échangée lors de l'appairage. Ici, un appareil crée une adresse MAC aléatoire à partir de l'IRK, qui est transmise dans un paquet d'advertisement. L'autre appareil peut en déduire l'adresse originale afin de continuer la communication. Une nouvelle adresse est générée périodiquement, ce qui apporte une certaine protection.
- Privée non-résoluble aléatoire : Cette fois, l'adresse est convertie en un nombre aléatoire. Ce nombre peut être renouvelé très régulièrement, ce qui apporte une forte sécurité.

In fine, le fait que seul la machine appairée puisse deviner l'adresse MAC originale à partir d'une adresse randomisée évite le phénomène de tracking.[Woolley15]

II.3 Liste blanche

La liste blanche, ou whitelist en anglais, est une méthode permettant aux appareils de filtrer les autres appareils selon leurs adresses MAC. La politique de filtrage détermine de quelle manière un annonceur (advertiser) peut traiter la demande de scan ou connexion. Lorsque les politiques de filtrage de l'advertising sont appliquées, les appareils peuvent uniquement autoriser les demandes de scan ou connexion de la liste blanche. Pour ce faire, un appareil doit maintenir une liste blanche localement, qui stocke l'adresse et son type. Ainsi, un appareil n'a pas besoin de répondre à chaque demande de connexion, car la plupart sont filtrées.

Conclusion

Les données sont chiffrées de bout en bout. Il faut donc que le pirate intercepte les données quand elles sont encore susceptibles d'être transmises en clair, soit au moment de

l'appairage. Ensuite, deux appareils connectés en BLE sont en général assez proches. Un pirate aurait donc besoin de s'immiscer physiquement entre les deux appareils, ce qui est rarement possible. Pour ces raisons, il ne semble pas aisé de réaliser une attaque sur BLE.
En pratique, tous les appareils BLE n'appliquent pas forcément les méthodes de sécurisation précédemment mentionnées. En effet, cela peut être jugé inutile dans certains cas. Par exemple sur certains systèmes de paiement en caisse. Ou bien difficile à mettre en place. Pour illustrer, on peut concevoir qu'il est difficile de mettre en place un système d'appairage très sécurisé lorsque plusieurs Peripherals rentrent en jeu.
De plus, il est possible de sécuriser la couche application. D'autant plus que, même la connexion entre deux appareils chiffrée, les données une fois sur l'appareil sont disponibles sur toutes les applications.

Nous verrons dans la partie suivante les failles persistantes malgré toutes les sécurisations évoquées.

Chapitre III

Attaques possibles et propositions de protection

Introduction

Malgré les mesures de sécurité précédemment énoncées, certaines attaques sont connues à ce jour. Elles vont d'une simple interception passive de données à l'usurpation d'identité et au déni de service. Comme il ne s'agit plus de faille zero day, des réponses ont déjà été proposées à ces attaques. Nous les présentons ci-dessous les attaques avec leurs solutions.

III.1 Attaques sur advertisement

III.1.1 Principe

Les paquets d'advertisement émis par les appareils connectés peuvent être usurpés et tromper le smartphone visé ou le Central de manière générale. Dans une optique d'optimisation de la batterie, les appareils augmentent souvent l'intervalle entre deux advertisements. Ainsi, un pirate peut brouiller l'appareil et récupérer ses paquets d'advertisement qu'il enverra à une fréquence bien plus haute et quand il veut au Central. Il suffit en effet que le premier paquet reçu par le Central vienne de la machine pirate pour que l'attaque réussisse. De plus, quand la machine est en mode connecté, avec la machine pirate par exemple, alors elle n'envoie plus d'advertisement. Ce qui peut favoriser la tâche du pirate, qui a pour but d'envoyer des paquets au Central à la place du Peripheral original. Ainsi l'attaque la plus simple est le déni de service. Il suffit au pirate de se faire passer pour le Central, dans le but d'intercepter les paquets du Peripheral. De la sorte, on empêche facilement la connexion entre le Peripheral et le Central.

III.1.2 Attaques

- Déni de service en domotique : De plus en plus, les smartphones sont connectés à des objets de la maison : ampoule, serrure, etc. Le smartphone garde en mémoire l'adresse MAC des objets connectés. Il suffit à un pirate d'usurper l'adresse MAC d'un objet connecté, d'établir la connexion avec le smartphone et envoyer de faux paquets (contenant « on » à la place de « off » par exemple). Ainsi, le smartphone ne peut plus contrôler l'objet connecté original.[Benchoff16]
- Compromission d'un antivol : Sur le même modèle, il existe des bagages dont l'antivol est connecté par BLE au smartphone. Si un pirate usurpe l'adresse MAC de l'antivol et envoie des paquets contenant les informations opposées au vrai, alors la sécurité du bagage est compromise.[Jasek17]
- Abus de beacons : Un beacon est un petit boîtier BLE qui détecte la proximité d'un smartphone. Il y a une application qui permettait de gagner des points en

visitant des lieux. Les points étaient échangeables contre des cadeaux. L'application a rencontré des problèmes car il était possible d'usurper les paquets envoyés par les beacons. Cela avait pour conséquence de faire gagner des points à des smartphones immobiles. Il fallait pour cela connaître la valeur de certains descripteurs de beacons originaux : UUID, coordonnées GPS, etc.[Jeon18]

III.1.3 Contre-mesures

Pour contrer cela, on a introduit des systèmes de signature et de chiffrement supplémentaires pour les valeurs broadcastées. Par exemple, certains vendeurs ont fait en sorte que les paquets contiennent une valeur pré-définie modifiée périodiquement. Seule l'application smartphone du vendeur permet de vérifier la valeur du paquet. Cependant ce genre de système a ses limites. Il faut qu'il soit compatible avec le matériel et le logiciel. Et parfois une connexion internet est requise. Enfin, le vendeur ne fait pas forcément vérifier son algorithme de chiffrement par des experts car ce sont des implémentations non-officielles. Il existe donc toujours un risque. Le plus sûr reste de ne pas faire confiance aux paquets d'advertising pour des choses importantes.

III.2 Interception passive

III.2.1 Principe

Les paquets non chiffrés peuvent facilement être victimes d'espionnage. Beaucoup de matériel accessible permet de réaliser une telle attaque. C'est le cas de la clé Ubertooth vendue par Great Scott Gadget [IMA17] ou un dongle utilisant le module BLE nRF51822 de Nordic Semiconductor [Nordic]. Le matériel, accompagné du système logiciel correspondant, permet de sniffer les paquets grâce à des applications comme Wireshark. Cela reste possible via des attaques actives comme on le verra dans la partie d'après.

III.2.2 Attaques

- Smart finder : « smart finder » est un objet connecté qui bénéficie d'une authentification à 6 chiffres. Ces 6 chiffres sont envoyés en clair par le smartphone vers l'objet connecté, par sécurité. Il était possible de voir ses 6 chiffres en clair grâce à l'outil GATTacker.
- Gestion de Beacons : En général, les beacons bénéficient d'un mot de passe statique. Plus précisément, chaque appareil a son propre mot de passe configuré. Pour se connecter, il se peut donc que l'application associée sur smartphone envoie le mot de passe en clair.

- OTP authentication token : « One Time Password » permet à un appareil de se connecter avec un mot de passe à usage unique, avec 6 chiffres en BLE. Il a cependant été observé à l'aide de sniffers que parfois les mots de passe ne sont pas chiffrés. Ce n'est pas forcément dangereux lors d'une attaque passive, car une fois utilisé le mot de passe n'est plus valable. En revanche, lors d'une attaque active, un tiers peut récupérer le mot de passe pour usurper l'identité du client.[Jung15]

III.2.3 Contre-mesures

Les attaques évoquées sont similaires et requièrent donc des solutions similaires. Déjà, comme on peut trivialement le deviner, il est important de chiffrer toutes les characteristics, en particulier avec les outils de sécurité offerts par la couche de liaison de Bluetooth. On peut également implémenter le chiffrement sur les couches plus hautes selon le protocole propriétaire. Il faudrait alors tester la robustesse des implémentations pour plus de sûreté. En parallèle, il faudrait s'assurer de l'identité de chaque appareil avant l'appairage, en plus de la confidentialité du message.

III.3 Interception active

III.3.1 Principe

Contrairement à l'interception passive, le pirate se fait passer à la fois pour le Central pour recevoir les paquets du Peripheral, et à la fois pour le Peripheral pour envoyer les paquets au Central. Il use pour cela de la modification de son adresse MAC, par exemple possible avec le programme bdaddr. Ainsi les paquets transitent par la machine pirate, et le pirate peut les modifier et les ré-envoyer. Le logiciel GATTacker permet de faire cela, voici un schéma où l'on voit l'IoT communiquant avec le Central en pensant qu'il s'agit du smartphone, et le smartphone échangeant avec le Peripheral à la place de l'IoT. On parle ici d'attaque Man In The Middle (MITM).[Jasek]

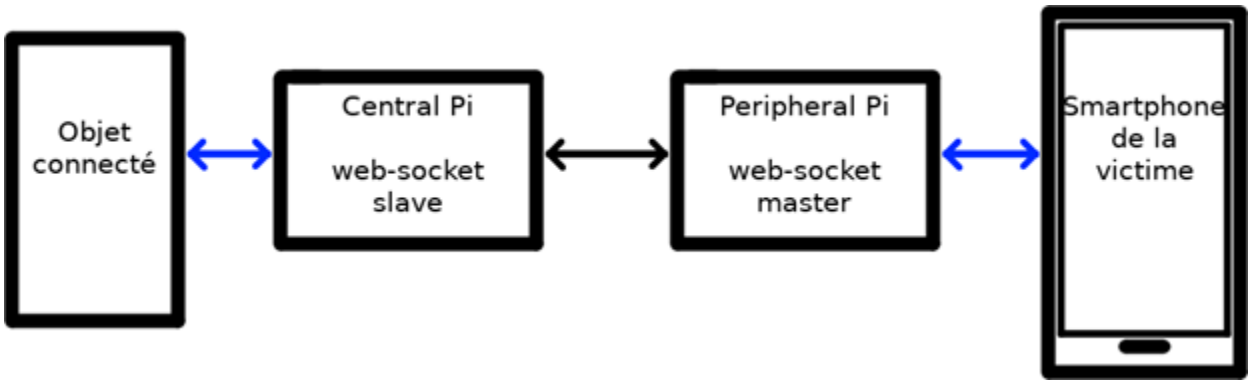

Figure III.1. MITM avec GATTacker

III.3.2 Attaques

- Manipulation de données : Cette faille existe sur certains terminaux de paiement connectés à un smartphone. Les données échangées entre le terminal et le smartphone sont bien chiffrées. En revanche, certaines commandes, en particulier celles permettant d'afficher du texte sur le terminal, ne le sont pas. En analysant le contrôle de redondance cyclique sur le smartphone grâce au texte en clair, il est possible d'afficher n'importe quel texte sur le terminal par MITM. Couplé à de l'ingénierie sociale, il est théoriquement possible d'arnaquer le vendeur, en affichant « Paiement accepté » sur le terminal après paiement alors que ce n'est pas le cas.
- Injection de commande : Il est déjà arrivé qu'un système intelligent de voiture soit piraté par injection de commande [Posch20]. Plus exactement, le smartphone de la victime se connecte à la voiture, pour la verrouiller par exemple, sur un modèle d'authentification défi-réponse. Ensuite, l'application communique de manière non chiffrée avec la voiture grâce à la session. Un pirate peut, sans altérer le processus d'authentification, intercepter la session et envoyer des commandes à la voiture, y compris changer les clés de session pour prendre le contrôle total de la voiture. Cela est facilité par le fait que l'implémentation de la sécurité est souvent réalisée par le fabriquant lui-même, ce qui n'est pas gage de robustesse. De surcroît, l'application mobile envoie ses données automatiquement à la voiture grâce à des processus en arrière-plan. Ainsi, si le smartphone est trop loin de la voiture pour que cette dernière le détecte, mais que le pirate est à mi-distance et peut se connecter aux deux, alors il s'agit d'une situation à risque. On parle de Bluesnipping lorsque le pirate cherche à agrandir sa portée Bluetooth, ce qui peut être utilisé dans ce genre d'attaque.
- Replay : On peut utiliser cette attaque grâce à des serrures intelligentes, qui en ont déjà été victimes, comme les premières générations d'August Smart Lock. Ici, les échanges sont bien chiffrés. En revanche, il n'y a pas de sécurité contre le replay, c'est-à-dire le renvoie de paquets déjà envoyés, même chiffrés. Concrètement, la victime peut recevoir un paquet indiquant que la serrure est bien verrouillée alors que ce n'est pas le cas. Pour ce faire, une faiblesse est utilisée. Pour chaque défi donné, le smartphone envoie une même clé de chiffrement. Il est donc aisé pour le pirate d'écouter l'échange entre la serrure et l'application vulnérable, d'abord des clés, puis de la communication. La fois suivante, pendant le processus d'authentification, l'appareil pirate peut se faire passer pour la serrure et envoyer les mêmes défis que la serrure aurait envoyé. Le pirate ayant connaissance de la clé, il peut déchiffrer tout l'échange. Et c'est à ce moment qu'il peut jouer le replay, en renvoyant un paquet à l'utilisateur indiquant que la serrure est bien verrouillée, alors que ce n'est pas le cas.

III.3.3 Contre-mesures

Il s'agit principalement des mêmes contre-mesures préconisées pour lutter contre l'interception passive. On peut en plus ajouter une fonctionnalité permettant de vérifier qu'il n'y a pas eu de tentative d'interception ni de modification des données. Il existe pour cela des algorithme de Contrôle de Redondance Cyclique (CRC) ou de checksum, assurant l'intégrité des données. En d'autres termes, il faut s'assurer de l'authenticité des characteristics.

III.4 Appareils en libre accès

III.4.1 Principe

Il est courant qu'un appareil soit disponible pour toute tentative de connexion avec un appareil inconnu. Ce qui est encore plus risqué en l'absence chiffrement préalable. Cependant, un pirate peut trouver des failles et les exploiter, bien qu'elles ne soient pas génériques.

III.4.2 Attaques

- Interfaces non-sécurisés : Des modules peuvent se connecter à certaines interfaces sans authentification, et envoyer des requêtes GATT comme write ou notify. Selon l'interface, il se peut que certaines requêtes dérèglent l'interface et nuisent à son bon fonctionnement, selon le système embarqué. On peut détecter ces interfaces qui ne demandent pas de connexion. Ainsi elles sont susceptibles de recevoir des attaques.
- Brute force : La majorité de Peripherals et beacons ne limitent pas le nombre de tentatives de connexion. Sur 6 chiffres, un pirate peut réaliser une attaque par brute force sans difficulté.[Rose18]
- Carence d'aléatoire : Certains modules n'intègrent pas de générateur de nombre aléatoire. Les développeurs sont obligés de se baser sur certaines variables préexistantes, pour générer des nombres peu aléatoires. Par exemple, certains modules multiplient leur Serial Number avec la température ambiante [Grant21]. Cela peut être critique. En effet, toujours lors d'une authentification défi-réponse, si un pirate devine la valeur du challenge, alors il peut réaliser une attaque par MITM.
- Trop de services disponibles sans authentification : Trop de services peuvent être accessibles. Même si individuellement l'accès à un seul service ne représente pas de danger, l'accès à tous les services peut révéler des informations censées êtres confidentielles.
- Mauvaise gestion des entrées : Cela rejoint le premier point. Dans certains cas, envoyer des valeurs aberrantes en guise de characteristics peut compromettre le com-

portement de l'appareil en question. Cela arrive quand les variables reçues ne sont pas contrôlées.

- Plus généralement, tous les défauts de logique : Voici un exemple parmi tant d'autres. Il peut arriver qu'un appareil stocke plusieurs clés pour différentes connexions. Un utilisateur souhaitant se connecter doit alors donner l'index de la clé ainsi que sa valeur. Si un utilisateur malveillant entre un index out of range et prédit la valeur écrite en mémoire à cette adresse-là, alors il peut se connecter à l'appareil.

III.4.3 Contre-mesures

Il suffit ici d'être consciencieux et de bien tester les systèmes lors de leur conception. En reprenant les exemples ci-dessus, on peut limiter l'accès à certaines données ou vérifier chaque entrée. De même, on peut limiter la durée d'activité de l'appareil, pour réduire la plage de possibles attaques.

III.5 Attaques lors de l'appairage

III.5.1 Principe

Comme on l'a vu, chiffrer un échange est primordial. Pour chiffrer un échange, il faut d'abord appairer les deux systèmes. Cependant, il peut résider des failles lors de cette étape, qui peuvent même être exploitées par ingénierie sociale.

III.5.2 Attaques

- Failles de Just Works : Just Works est fréquemment utilisé lors de l'appairage. Au moment de la liaison entre un objet connecté et un smartphone, le pirate peut s'approcher de l'appareil pour tenter d'accéder aux characteristics et ainsi créer une nouvelle liaison. Alors, le pirate peut créer un clone de l'objet afin de se faire passer pour lui. De là, le smartphone se connecte au clone. S'il ne vérifie pas la présence de l'adresse MAC dans sa liste, c'est gagné. Le smartphone n'ayant pas de clés LTK associées à l'adresse MAC pour chiffrer, mais étant bien connecté à l'appareil, il va communiquer en chiffrant avec ce qu'il a : le néant. En revanche, si la présence de l'adresse MAC est vérifiée, alors le pirate doit la cloner. Mais cela s'arrête là car l'attaquant ne dispose pas de la LTK, ce qui le déconnecte rapidement.[Lounis20]
- Code PIN : Comme expliqué ci-dessus, un pirate peut cloner un appareil lors de la liaison. L'échange n'étant pas sécurisé, le smartphone se déconnecte automatiquement. Le pirate désactive alors son clone et peut sniffer l'échange lors du processus d'appairage. Lors de l'appairage suivant, le pirate sera en mesure de craquer le code PIN grâce à Crackle afin de récupérer la LTK [Crackle14]. C'est ainsi qu'une

interception active peut commencer.

- Mode SCO : C'est le mode Secure Connection Only qui déconnecte l'appareil dès lors que la connexion n'est pas sécurisée. Cependant, BLE a déjà été compromis à de multiples par une faille utilisant SCO.[Zhang20]

III.5.3 Contre-mesures

Pour contrer cela, il est tout simplement recommandé d'utiliser les mécanismes d'appairage les plus sûrs, que nous avons vu dans la partie précédente. En plus de cela, on peut faire en sorte que l'appairage s'initie lors d'une action manuelle sur l'appareil connecté, comme l'appuie sur un bouton. Enfin, les applications mobiles devraient prévenir l'utilisateur lors d'une tentative d'interception de données, afin que celui-ci prenne des mesures, comme renforcer la sécurité, changer le mot de passe ou encore trouver quel appareil tente de pirater.
On peut en dernier lieu ajouter une sécurité au niveau de la couche applicative afin de vérifier l'existence de l'identité des personnes se connectant.

III.6 Contournement de la liste blanche

Il s'agit d'une faille assez simple à comprendre. On a déjà expliqué que certains appareils ont une liste d'adresses MAC de confiance. Si un pirate en a connaissance par un quelconque moyen, alors il peut modifier sa propre adresse MAC pour en faire partie et se connecter à la victime. Il suffit ici d'avoir d'autres moyens de vérification, comme une LTK.

III.7 Violation de confidentialité

On peut également aborder les choses de la manière suivante. Les données contenues dans les paquets d'advertisement publics peuvent être collectées. Cela pose problème si ces données doivent rester confidentielles. C'est spécifiquement ce que nous étudierons lors de la mise en pratique dans la partie suivante.
Au-delà de ça, la capacité de voir quels appareils sont connectés et à quel moment représente un danger pour la confidentialité des utilisateurs. On parle de Bluetracking. Le pirate peut connaître le journal de connexions de chaque appareil à sa portée. Ce qui peut en plus l'aider à prévoir une potentielle attaque.

Conclusion

Il existe plusieurs catégories d'actes malveillants : l'écoute passive, l'attaque active par MITM, déni de service, etc. Certaines failles nécessitent du matériel particulier, comme un sniffer. L'attaquant doit aussi connaître les vulnérabilités des appareils qu'il tente de pirater, car tous les appareils n'ont pas les mêmes failles. Les constructeurs ont adapté leurs appareils de sorte à contourner ces attaques. Une des faiblesses réside dans les mécanismes de chiffrement qui sont implémentés par ces fabricants, qui ne sont pas forcément robustes. En plus du chiffrement, la sécurité doit idéalement être holistique.

Chapitre IV

Attaque BLE par sniffing

Introduction

Après avoir passé en revu une grande partie des attaques, nous allons en mettre une en pratique. Il s'agit d'une interception passive de données. Pour cela, nous disposons d'un smartphone et d'un Raspberry Pi 4 bénéficiant de BLE. Nous avons également un ordinateur avec un sniffer BlueFruit nrf51822 V2. Le Raspberry fera office de Peripheral, et le smartphone de Central.
Le Peripheral doit envoyer une fréquence cardiaque, bien évidemment simulée, sans chiffrement. Cette donnée peut être jugée sensible dans les domaines de la médecine et de la confidentialité. La fréquence cardiaque doit être lue par le Central. Dans le cadre de l'attaque, le l'ordinateur interceptera les données via le sniffer, et lira les fréquences cardiaques sans que les autres appareils ne s'en rendent compte.

Figure IV.1. Sniffer BlueFruit

IV.1 Préparation du serveur GATT

D'abord, nous allons préparer le serveur GATT. Pour cela, nous récupérons un code disponible sur Github. Il est basé sur la technologie BlueZ et est écrit en C. Voici les commandes permettant de le télécharger et le compiler :

```
git clone https://github.com/evanslai/bluez-gatt.git
```

```
cd bluez-gatt
mkdir build
cd build
cmake ../
make
```

Si cmake n'est pas installé, alors il faut écrire la commande ci-dessous avant de recompiler.

```
sudo apt-get install build-essential cmake
```

Le serveur GATT est prêt.

IV.2 Préparation du client GATT

Du côté du smartphone, de nombreuses applications permettent à l'appareil de se comporter comme un Central BLE. Sur Google PlayStore, nous avons opté pour «BLE Scanner».

IV.3 Paramétrage de l'appareil pirate

IV.3.1 Drivers

D'abord, il faut installer les drivers nécessaires au bon fonctionnement du sniffer. Nous avons un modèle « Black Board », donc les drivers de SiliconLabs sont nécessaires, quel que soit l'OS. Pour une « Blue Board », il faut les drivers de FTDI Chip.

IV.3.2 Installation de l'API AdaFruit

On installe l'API fournie par AdaFruit qui permet d'exploiter les capacités de la clé. Cette API développée en Python permet de récupérer les données envoyées sur le Serial Port du sniffer et de les structurer sous un fichier pcap. Ainsi, on pourra ouvrir les paquets interceptés avec Wireshark.

```
git clone
https://github.com/adafruit/Adafruit_BLESniffer_Python.git
cd Adafruit_BLESniffer_Python
```

Il faut bien sûr avoir Wireshark sur la machine. Sous Linux, la commande adaptée est la suivante :

```
sudo apt-get install wireshark
```

Tandis que sur Mac ou Windows il faut aller sur la page de téléchargement de Wireshark.

IV.3.3 Détermination du Serial Port

On détermine maintenant le Serial Port associé au sniffer. Sous Linux, il est de la forme /dev/ttyXXXX. On le trouve à l'aide de la commande suivante :

```
dmesg | grep tty
```

Sous Windows, il est de la forme COMX. On ouvre le Device Manager et on cherche dans la catégorie « COM & LPT ». Normalement il s'agit d'un port COM.
Sous Mac, il ressemble à /dev/tty.X. la commande ci-dessous est nécessaire pour le trouver :

```
ls /dev/tty.*
```

IV.3.4 Python et pySerial

Nous avons essayé de lancer l'API à l'aide de Python 3.6.9, mais cela n'a pas fonctionné alors qu'avec Python 2.7 oui. Nous recommandons donc d'utiliser cette dernière version. De plus, les packages six et pySerial sont indispensables. Il faut donc veiller à leur installation.

IV.4 Interception des données (sniffing)

On place de préférence la machine pirate entre le client et le serveur GATT.

IV.4.1 Lancement du serveur GATT (Raspberry)

Sur le Raspberry Pi, on regarde l'adresse MAC Bluetooth à l'aide de la commande

```
hciconfig
```

On la garde en mémoire. Maintenant, on se retrouve dans le dossier bluez-gatt/build. On lance le serveur avec l'argument -r pour activer le service « Heart Rate ».

```
chmod +x btgatt-server
./btgatt-server -r
```

IV.4.2 Lancement de l'API (ordinateur pirate)

Pour lancer l'API, il faut d'abord être dans son dossier. On use alors de la commande :

```
python2 sniffer.py <serial-port>
```

Il faut ajouter sudo avant si on est sous Linux, et appeler simplement python si la version est adéquate.
Il faut noter le chemin vers les logs et les paquets capturés, qui est affiché dans les premières lignes. L'API se connecte dans un premier temps au sniffer. Puis, elle scanne tous les appareils BLE environnants. Après les avoir identifié avec leur adresse MAC Bluetooth et leur RSSI, elle les numérote et les liste. Plus la machine est proche, plus le RSSI est grand. On doit alors choisir le bon appareil. Comme on est à la fois pirate et utilisateur, la tâche nous est simplifiée, on sait quelle adresse MAC choisir. Le serveur GATT est désormais sur écoute.

```
^Cquentin@quentin-hp:~/Téléchargements/Adafruit_BLESniffer_
Python-master$ sudo python2 sniffer.py /dev/ttyUSB0
[sudo] Mot de passe de quentin :
Capturing data to logs/capture.pcap
Connecting to sniffer on /dev/ttyUSB0
Scanning for BLE devices (5s) ...
Found 5 BLE devices:

  [1] "" (40:93:BB:93:D6:19, RSSI = -47)
  [2] "" (70:DF:9D:4B:F2:14, RSSI = -87)
  [3] "" (41:10:61:3E:A7:1A, RSSI = -89)
  [4] "" (7B:02:23:B7:93:DF, RSSI = -67)
  [5] "" (DC:A6:32:75:94:5D, RSSI = -53)

Select a device to sniff, or '0' to scan again
> 5
Attempting to follow device DC:A6:32:75:94:5D
.....................................................
```

Figure IV.2. AdaFruit Sniffer API

IV.4.3 Connexion du client (smartphone)

Sur l'application, on se connecte au Raspeberry. On le retrouve lors du scan à l'aide du RSSI, du nom et/ou de l'adresse MAC. Une fois connecté, on ouvre la characteristic « Heart Rate ». Pour afficher la fréquence cardiaque, on doit simplement autoriser les notifications, en appuyant sur « N » par exemple.

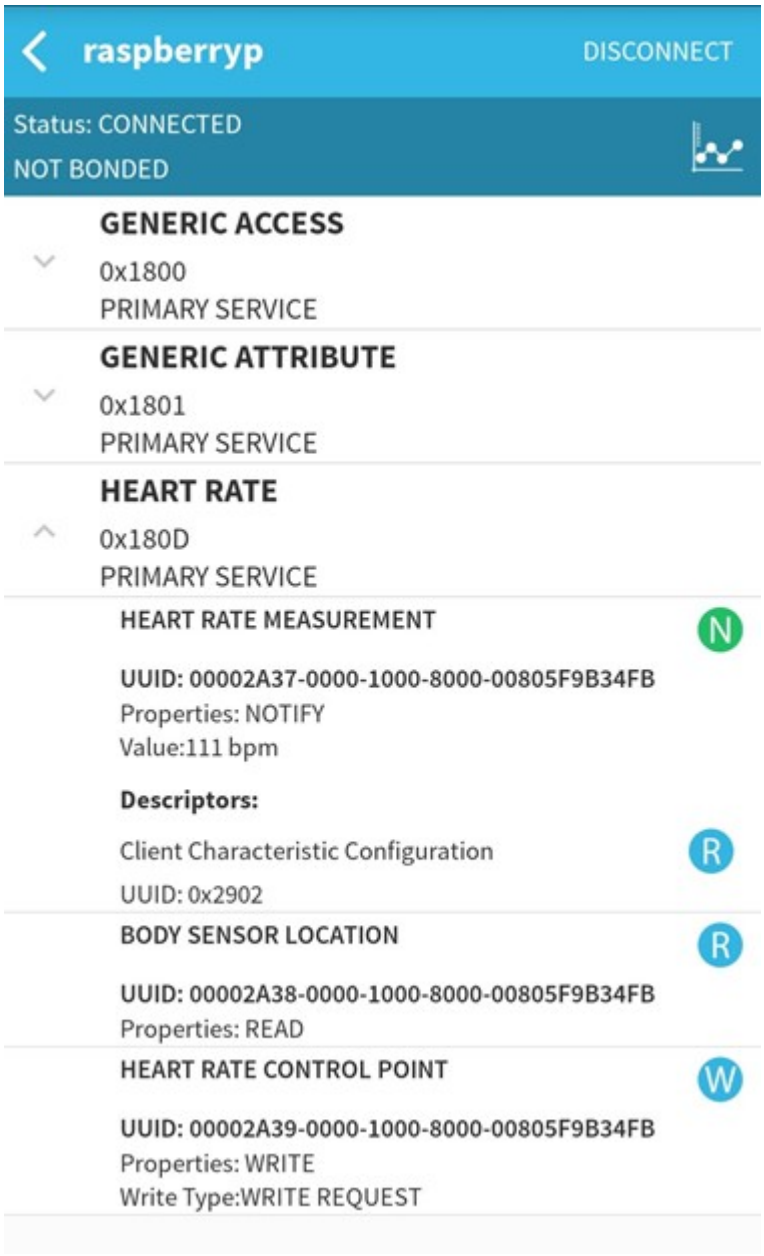

Figure IV.3. BLE Central

IV.4.4 Écoute passive du serveur (ordinateur pirate)

Malgré les échanges sans erreur entre le client et le serveur GATT, l'API capture tous les paquets qui transitent. On peut arrêter l'écoute à tout moment à l'aide de l'appel système Ctrl+C. On ouvre le fichier Wireshark nommé « capture ». Rappelez-vous, son chemin était indiqué au lancement de l'API. D'ailleurs, en cas de problème, on peut se référer aux logs dans le même dossier.

Une fois sur Wireshark, on cherche les paquets contenant le « Heart Rate Measurement », la mesure de la fréquence cardiaque. On va les retrouver dans les paquets dont le protocole associé à ATT pour Attribute, qui partent du Slave vers la Master. Pour une meilleure

lisibilité, on peut supprimer les paquets vides avec le filtre « btle.length != 0 ». On cherche les paquets qui nous intéressent avec le raccourcis clavier Ctrl+F. Les paquets qui nous intéressent contiennent la chaîne de caractères « Heart Rate Measurement », dans « Détail du paquet ».

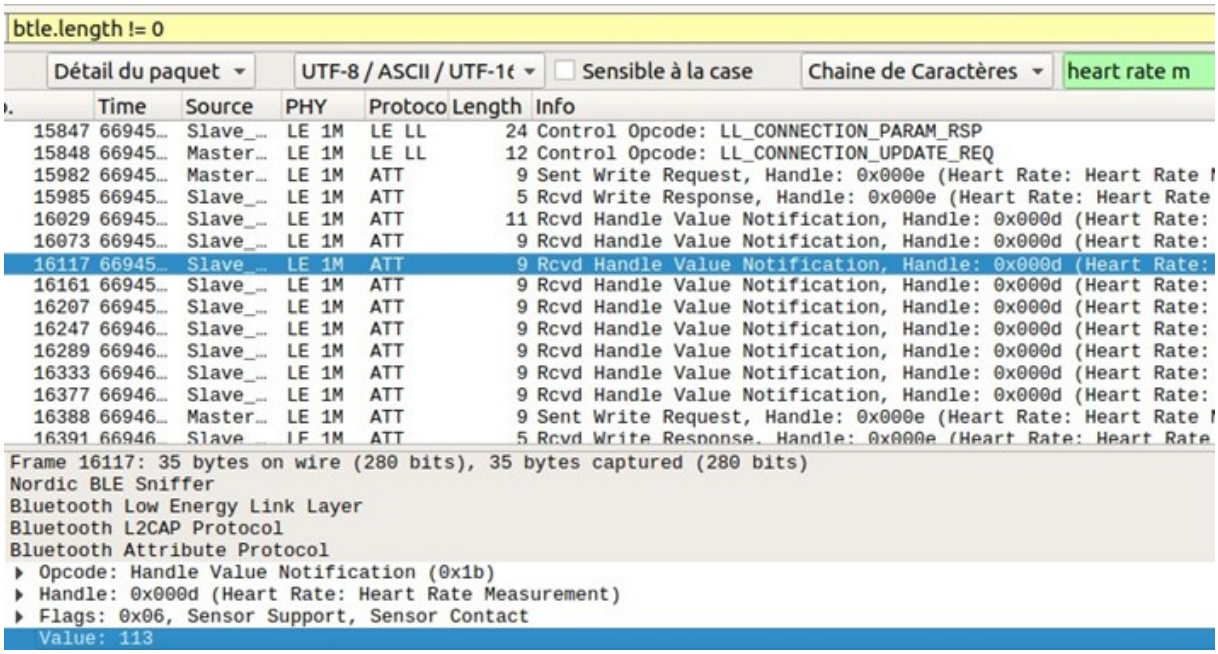

Figure IV.4. Wireshark affichant la fréquence cardiaque

Ainsi, toutes les mesures de fréquence cardiaque sont accessibles. Il ne peut y avoir de confidentialité sans chiffrement.

Conclusion

En définitive, une personne mal intentionnée peut intercepter des données pour quelques dizaines d'euros seulement. La démarche à suivre est extrêmement simple si les données sont transmises en clair. Il suffit d'un outil comme crackle si elles sont chiffrées avec une méthode comme Just Works. On peut imaginer qu'avec plus de temps et de connaissances que nous, il est possible d'avoir une action bien plus considérable, comme faire du replay. Dans notre cas, un algorithme de chiffrement, au niveau de la link layer par exemple, aurait déjà suffi à éviter une telle écoute. Chaque vulnérabilité a une solution adaptée.

Conclusion

Les risques liés à BLE ont augmenté de manière proportionnelle à l'essor du protocole. Malgré les mécanismes de sécurité standardisés, de nombreuses failles restent exploitables. Elles peuvent être responsables d'interceptions passives ou actives, falsification de données ou encore de déni de service. Il est du ressort des fabricants du matériel et des développeurs implémentant les systèmes de prévenir les risques. Comme nous l'avons vu, il existe de nombreux moyens de s'assurer de la sécurité d'un terminal BLE et d'un échange, en partant de la robustesse du chiffrement jusqu'à la vérification des entrées en passant par un appairage sûr.

En quelques mois il a été possible, non sans difficulté, de compromettre la confidentialité de données BLE. De cette manière, on a pu comprendre comment ce protocole précis pouvait être sécurisé. Cela est d'autant plus important que le BLE est utilisé dans divers domaines importants, y compris en médecine. Avec un peu plus de temps, nous aurions pu nous intéresser à un autre réseau, tel que LoRa. Sa place est également non négligeable dans le monde de la télécommunication et de l'IoT, mais son fonctionnement est très différent.

Références

[Benchoff16] Biren BENCHOFF. *« The Terrible Security Of Bluetooth Locks »*. 8 août 2016. URL : https://hackaday.com/2016/08/08/the-terrible-security-of-bluetooth-locks/.

[Bluetooth] Bluetooth SIG. *« Bluetooth Radio Versions »*. URL : https://www.bluetooth.com/learn-about-bluetooth/radio-versions/.

[Crackle14] *« crackle, crack Bluetooth Smart (BLE) encryption »*. 24 avril 2014. URL : http://lacklustre.net/projects/crackle.

[Grant21] Reilly GRANT et Ovidio RUIZ-HENRÍQUEZ. *« Web Bluetooth, Draft Community Group Report »*. 4 mai 2021. URL : http://webbluetoothcg.github.io/web-bluetooth.

[IMA17] Projets IMA. *« Evaluation des attaques sur protocole Bluetooth »*. 2017. URL : https://projets-ima.plil.fr/mediawiki/index.php/Evaluation_des_attaques_sur_protocole_Bluetooth.

[Jasek] Sławomir JASEK. *« GATTACKING BLUETOOTH SMART DEVICES »*. URL : https://raw.githubusercontent.com/securing/docs/master/whitepaper.pdf.

[Jasek17] Sławomir JASEK. *« Blue picking –hacking Bluetooth Smart Locks »*. 3 mars 2017. URL : https://conference.hitb.org/hitbsecconf2017ams/materials/D2T3%20-%20Slawomir%20Jasek%20-%20Blue%20Picking%20-%20Hacking%20Bluetooth%20Smart%20Locks.pdf.

[Jeon18] Perm Soonsawad KANG EUN JEON James She et Pai Chet NG. *« BLE Beacons for Internet of Things Applications : Survey Challenges and Opportunities »*. 2018. URL : https://www.researchgate.net/publication/322201975_BLE_Beacons_for_Internet_of_Things_Applications_Survey_Challenges_and_Opportunities.

[Jung15] Kwangsu Cho HYUNHEE JUNG Dongryeol Shin et Choonsung NAM. *« BLE-OTP Authorization Mechanism for iBeacon Network Security »*. août 2015. URL : https://www.researchgate.net/publication/283191016_BLE-OTP_Authorization_Mechanism_for_iBeacon_Network_Security.

[Lounis20] Karim LOUNIS et Mohammad ZULKERNINE. *« Bluetooth Low Energy Makes "Just Works" Not Work »*. 2 avril 2020. URL : https://hal.archives-ouvertes.fr/hal-02528877/document.

[Nordic] Nordic SEMICONDUCTOR. *« nRF51822 »*. URL : https://infocenter.nordicsemi.com/pdf/nRF51822_PS_v3.4.pdf.

[Posch20] Maya POSCH. *« Hands-On : Wireless Login With The New Mooltipass Mini BLE Secure Password Keeper »*. 23 juillet 2020. URL : https://hackaday.com/2020/07/23/hands-on-wireless-login-with-the-new-mooltipass-mini-ble-secure-password-keeper/.

[Ren16] Kai REN. *« Bluetooth Pairing Part 3 – Low Energy Legacy Pairing Passkey Entry »*. 25 août 2016. URL : https://www.bluetooth.com/blog/bluetooth-pairing-passkey-entry/.

[Rosa13] Tomas ROSA. *« Bypassing Passkey Authentication in Bluetooth Low Energy »*. 2013. URL : https://eprint.iacr.org/2013/309.pdf.

[Rose18] Anthony ROSE et al. *« Securing bluetooth low energy locks fromunauthorizedaccess and surveillance »*. 20 juin 2018. URL : https://hal.inria.fr/hal-01819142/document.

[Woolley15] Martin WOOLLEY. *« Bluetooth Technology Protecting Your Privacy »*. 2 avril 2015. URL : https://www.bluetooth.com/blog/bluetooth-technology-protecting-your-privacy/.

[Woolley16] Martin WOOLLEY. *« Advertising Works Part 2 »*. 9 février 2016. URL : https://www.bluetooth.com/blog/advertising-works-part-2/.

[Zhang19] Yue ZHANG, Jian WENG et Xinwen FU. *« B Bluetooth Low Energy (BLE) Security and Privacy »*. Octobre 2019. URL : https://www.researchgate.net/publication/336360887_B_Bluetooth_Low_Energy_BLE_Security_and_Privacy.

[Zhang20] Yue ZHANG et al. *« Breaking Secure Pairing of Bluetooth Low Energy Using Downgrade Attacks »*. 14 août 2020. URL : https://www.usenix.org/system/files/sec20-zhang-yue.pdf.

Images

- Page 3, https://lembergsolutions.com/sites/default/files/Articles/How%20Secure%20is%20the%20BLE%20Connection/BLE_communication.svg
- Page 3, https://blog.groupe-sii.com/wp-content/uploads/2015/11/image009.png

- Page 5, https://microchipdeveloper.com/local–files/wireless:ble-gatt-data-organization/gatt-heart-rate-service.png
- Page 12, image personnelle
- Captures d'écran de la démonstration : images personnelles

Glossaire

ACK	ACKnowledgement
API	Application Programming Interface
ATT	ATTribute
BLE	Bluetooth Low Energy
Bluetooth SIG	Bluetooth Special Interest Group
CRC	Contrôle de Redondance Cyclique
GATT	Generic ATTribute
IoT	Internet of Things
IRK	Identity Resolving Key
LTK	Long Term Key
MAC	Media Access Control
MITM	Man In The Middle
NFC	Near Field Communication
OOB	Out of Band
OTP	One Time Password
PIN	Personal Identification Number
RSSI	Received Signal Strength Indication
STK	Short Term Key
TK	Temporary Key
UUID	Universally Unique IDentifier

Advertising	Envoi de paquets pour notifier la présence du terminal
Appairage	Association de deux terminaux Bluetooth
Bluesnipping	Technique utilisée par un pirate pour augmenter sa portée Bluetooth
Bluetracking	Traçage par les informations Bluetooth
Central	Terminal qui initie la connexion avec un ou plusieurs Peripherals
Characteristic	Élément contenant une variable et possiblement des descripteurs
Client GATT	Terminal qui reçoit les characteristics d'un serveur ou plusieurs serveurs
Descripteur	Variable donnant des informations sur une characteristic
Esclave	Terminal envoyant ses characteristics au maître
Liaison	Bonding en anglais, cela sert à planifier les futures connexions après l'appairage
Maître	Terminal initiant la connexion pour obtenir des characteristics
Peripheral	Peut accepter la connexion d'un Central
Service	Ensemble de characteristics
Serveur GATT	Terminal qui envoie ses characteristics à un client