HAL
open science

Détection d'intrusion dans des environnements connectés sans-fil par l'analyse des activités radio

Jonathan Roux

To cite this version:

Jonathan Roux. Détection d'intrusion dans des environnements connectés sans-fil par l'analyse des activités radio. Informatique mobile. Université Paul Sabatier - Toulouse III, 2020. Français. NNT : 2020TOU30011 . tel-02880658v2

HAL Id: tel-02880658

https://laas.hal.science/tel-02880658v2

Submitted on 22 Oct 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ FÉDÉRALE TOULOUSE MIDI-PYRÉNÉES

Délivré par :
l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)

Présentée et soutenue le *07/02/2020* **par :**

Jonathan ROUX

Détection d'intrusion dans des environnements connectés sans-fil par l'analyse des activités radio

---

JURY

Lilian BOSSUET	Professeur des Universités	Rapporteur
Samia BOUZEFRANE	Professeure des Universités	Rapporteur
Grégory BLANC	Maître de Conférences	Examinateur
Vianney LAPOTRE	Maître de Conférences	Examinateur
Ludovic MÉ	Professeur des Universités	Examinateur
Mohamed Kaâniche	Directeur de recherche	Examinateur
Éric Alata	Maître de Conférences	Examinateur
Vincent Nicomette	Professeur des Universités	Directeur de thèse

---

École doctorale et spécialité :
MITT : Domaine STIC : Sureté de logiciel et calcul de haute performance
Unité de Recherche :
Laboratoire d'analyse et d'architecture des systèmes
Directeur(s) de Thèse :
Vincent NICOMETTE
Rapporteurs :
Lilian BOSSUET et *Samia BOUZEFRANE*

Remerciements

Les travaux présentés dans ce manuscrit ont été effectués au Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS) du Centre National de la Recherche Scientifique (CNRS). Je tiens donc tout d'abord à remercier Liviu Nicu qui a assuré la direction du LAAS depuis mon arrivée.

Je remercie également Mohamed Kaâniche, le précédent directeur de l'équipe de recherche Tolérance aux Fautes et Sûreté de Fonctionnement informatique (TSF) qui m'a accueilli, ainsi qu'Hélène Waeselynck, qui l'a succédé. En plus de diriger cette équipe, Mohamed Kaâniche est très impliqué dans l'ensemble des travaux de recherche mené en son sein, et m'a donc soutenu autant administrativement que techniquement dans la réalisation de mes objectifs de recherche.

Je tiens à exprimer mon immense gratitude envers mon directeur de thèse, Vincent Nicomette, dont la bonne humeur, l'énergie et la bienveillance m'ont permis de découvrir la recherche et l'enseignement supérieur. Il a également su trouver les mots justes pour m'accompagner et m'inspirer dans les moments de doutes. Je remercie également Éric Alata, qui fut le deuxième encadrant officieux de cette thèse, et qui m'a considérablement apporté scientifiquement.

J'adresse également mes sincères remerciements aux membres du jury qui ont accepté de juger mon travail. Je leur suis très reconnaissant pour l'intérêt qu'ils ont porté à mes travaux et pour la pertinence de leurs questions :

- Grégory Blanc, Maître de conférences à Télécom SudParis.
- Lilian Bossuet, Professeur des universités à l'Université de Saint-Etienne.
- Samia Bouzefrane, Professeure des universités au CNAM Paris.
- Vianney Lapôtre, Maître de conférences à l'Université Bretagne Sud.
- Ludovic Mé, Professeur des universités à Centrale Supélec Rennes.
- Mohamed Kaâniche, Directeur de recherche au LAAS-CNRS.
- Vincent Nicomette, Professeur des universités à l'INSA de Toulouse.
- Éric Alata, Maître de conférences à l'INSA de Toulouse.

Je remercie tout particulièrement Lilian Bossuet et Samia Bouzefrane qui ont accepté de rapporter ma thèse, et dont les différents retours auront permis d'apporter un point final à ce manuscrit.

Cette thèse n'existerait pas sans l'accueil exceptionnel que j'ai reçu de l'ensemble de l'équipe TSF, et je ne peux donc pas finaliser ces travaux sans lui exprimer mes remerciements les plus sincères. Celle-ci restera pour moi l'environnement de travail et de réflexion par excellence, qui cultive l'esprit critique et l'épanouissement personnel. Les moments passés avec mes collègues doctorants et post-doctorants seront notamment gravés à jamais dans ma mémoire, je les remercie donc tous. Cependant, parmi cette équipe, certains remerciements particuliers s'imposent. Tout d'abord, une pensée toute particulière est dirigée vers mes deux principaux co-bureaux, Lola

et Kalou, avec qui j'ai tant partagé. Leur bonne humeur, leurs rires (et leurs coup de gueules) m'ont fait passer des moments extraordinaires qui m'ont définitivement changé. Cette période éprouvante n'aurait également pas été la même sans la présence de Guillaume et Matthieu, qui ont du finir par m'interdire l'accès à plein temps à leur bureau pour cause de réduction drastique de productivité. Je remercie chaleureusement Romain Cayre, qui a été une source de motivation indéfectible durant une grande partie de mes travaux, et qui n'arrêtera jamais de me remplir de fierté. Je tiens également à exprimer ma profonde gratitude envers Pierre-François, dont la collaboration et les conseils m'ont aidé à étoffer significativement le contenu de ce manuscrit. Finalement, ces remerciements ne seraient pas complets sans ceux exprimés envers Guillaume Auriol, le troisième encadrant caché (mais indispensable) derrière ces travaux. Son humanité, sa gentillesse et sa générosité sans bornes sont pour moi les deux brassards qui m'ont maintenu à flots durant ces trois années. La place me manque pour remercier encore tous ceux qui m'ont accompagnés, mais je tiens cependant à citer vos noms pour la postérité : Benlord, Joris, Aliénor, Clément, Rémi, Daniel, Malcolm, Christophe, Cyrius, Florent, Yuxiao, Jean, Bilel, Mohamed, Luca, Raul, William, Julien, Rui, Carla et Thierry.

Mes derniers remerciements sont adressés à ma famille proche : mes parents et ma soeur, qui m'ont soutenu dans mon choix de faire un doctorat. Je les remercie notamment d'avoir eu le courage d'essayer de comprendre mes travaux, pour pouvoir expliquer à leurs propres connaissances ce que je faisais de ma vie. Le plus grand de tous les merci est finalement adressé à Marie, dont la joie de vivre et la tendresse m'ont accompagné jusqu'à la toute fin. Je la remercie pour sa patience, pour son oreille attentive et pour son soutien qui furent des sources de motivation incomparables pour écrire le point final de cette thèse.

Table des matières

Introduction **1**

I Contexte et état de l'art **5**

1 Problématique **7**
1.1 L'écosystème de l'Internet des Objets 7
1.1.1 Environnements d'utilisation 9
1.1.2 Protocoles et communications 13
1.1.3 Caractéristiques des environnements et des objets 18
1.2 La sécurité dans l'Internet des Objets 21
1.2.1 Terminologie de la sécurité 21
1.2.2 Problématiques de sécurité liées aux objets connectés 25
1.2.3 Principales attaques visant les objets connectés 27
1.3 Solutions traditionnelles de sécurité réseau 31
1.3.1 Pare-feu 31
1.3.2 Réseau Privé Virtuel (VPN) 33
1.3.3 Système de Détection/Prévention d'Intrusion (IDS/IPS) . . . 33
1.4 Objectifs de la thèse 35

2 État de l'art **37**
2.1 Apprentissage automatique 37
2.1.1 Modèle et apprentissage 38
2.1.2 Différentes familles d'apprentissage 39
2.1.3 Compromis biais variance 40
2.1.4 Requêtes 41
2.2 Solutions de sécurité appliquées à l'IoT et limites 41
2.2.1 État de l'art des solutions de sécurité spécifiques 41
2.2.2 Récapitulatif des limites 43
2.3 La radio logicielle et les solutions existantes 43
2.3.1 Définition et fonctionnement de la *SDR* 44
2.3.2 Détection d'anomalies basée sur les communications radio . . 46
2.4 Conclusion 47

II Architecture de sécurité générique **49**

3 Approche générique pour la détection d'anomalies dans les environnements connectés **51**
3.1 Modèle de menaces et hypothèses 52

3.2 Vue d'ensemble de l'approche proposée 53
3.3 Sondes radio 55
3.3.1 Périphérique SDR 56
3.3.2 Contrôleur 57
3.4 Implémentation des sondes radio 58
3.4.1 Périphérique SDR 59
3.4.2 Contrôleur 60
3.5 Apprentissage du modèle des activités légitimes 61
3.5.1 Problématiques des données et choix du modèle 62
3.5.2 Auto-encodeur 64
3.6 IDS 65
3.6.1 Détection 66
3.6.2 Diagnostic 66
3.7 Conclusion 67

III Déploiement et évaluation pour les domiciles connectés 69

4 Contexte et spécificités d'implémentation 71
4.1 Introduction 71
4.2 Contexte des domiciles connectés 71
4.2.1 Caractéristiques considérées et moyens mis en œuvre 72
4.3 Implémentation et déploiement de l'approche pour des domiciles connectés 73
4.3.1 Implémentation de la phase d'apprentissage 74
4.3.2 Détection d'anomalies – IDS 79
4.4 Conclusion 80

5 Expérimentations & Résultats 81
5.1 Introduction 81
5.2 Environnement expérimental 82
5.2.1 Mise en place d'un environnement de test réaliste 82
5.2.2 Composition de l'environnement et comportements 83
5.3 Protocole expérimental 85
5.3.1 Installation de la solution 85
5.3.2 Collecte des données 86
5.3.3 Apprentissage du modèle 88
5.4 Évaluation 91
5.4.1 Protocole d'évaluation 91
5.4.2 Génération d'attaques 92
5.4.3 Injection d'attaques 94
5.4.4 Métriques d'évaluation 95
5.4.5 Résultats et discussions 97
5.4.6 Quantification du risque associée à la détection d'anomalies . 100

5.5 Conclusion 102
5.6 Synthèse de la partie III 102

IV Adaptation à un environnement professionnel et modèles de diagnostic 103

6 Détection et diagnostic des intrusions radios 105
6.1 Introduction 105
6.2 Adaptation de l'approche aux environnements professionnels 106
6.2.1 Spécificités de l'environnement 106
6.2.2 Moyens mis en œuvre 107
6.2.3 Vue d'ensemble de l'approche 108
6.3 Estimation de l'erreur de reconstruction 109
6.3.1 Modèle auto-encodeur pour la détection 109
6.3.2 Pré-traitement : suppression du bruit et passage à l'échelle . 111
6.3.3 Définition de l'erreur de reconstruction pour la détection . . . 112
6.4 Diagnostics temporel et fréquentiel 113
6.4.1 Diagnostic temporel 114
6.4.2 Diagnostic fréquentiel 115
6.5 Diagnostic spatial 117
6.5.1 État de l'art sur la localisation 117
6.5.2 Fusion des anomalies 119
6.5.3 Données de calibration et apprentissage 119
6.5.4 Modèle de diagnostic spatial 120
6.6 Conclusion 120

7 Expérimentations 121
7.1 Introduction 121
7.2 Environnement expérimental 122
7.2.1 Composition de l'environnement et déploiement de l'approche 122
7.2.2 Caractéristiques techniques des composants de l'approche . . 124
7.3 Paramétrage des modèles 124
7.3.1 Pré-traitement 124
7.3.2 Estimation de l'erreur de reconstruction 124
7.3.3 Diagnostic temporel 125
7.3.4 Diagnostic spatial 125
7.4 Protocole expérimental 125
7.4.1 Évaluation de la détection d'anomalies 126
7.4.2 Évaluation du diagnostic temporel 126
7.4.3 Évaluation du diagnostic fréquentiel 127
7.4.4 Évaluation du diagnostic spatial 127
7.5 Injection d'attaques réalistes 127
7.5.1 Attaques injectées 128

7.5.2 Anomalies non prévues 129
7.6 Résultats de l'évaluation 129
7.6.1 Résultats de la détection d'anomalies 129
7.6.2 Résultats du diagnostic temporel 130
7.6.3 Résultats du diagnostic fréquentiel 132
7.6.4 Résultats du diagnostic spatial 133
7.7 Discussions 134
7.7.1 Limites globales de l'approche 135
7.7.2 Comparaison expérimentale des environnements étudiés . . . 136
7.7.3 Vie privée 138
7.7.4 Amélioration du diagnostic spatial 139
7.8 Conclusion 140
7.9 Synthèse de la partie IV 140

Conclusion **141**
7.10 Contributions 141
7.11 Perspectives 142

A Notations **145**

Bibliographie **147**

Liste des figures

1.1 L'arbre de la sécurité de fonctionnement 22

2.1 Exemple de modulation FSK 45

3.1 Récapitulatif des étapes de l'approche 54
3.2 Monitoring des activités radios par la sonde 56
3.3 Exemple de spectrogramme 58
3.4 Exemple d'auto-encodeur 64
3.5 Un spectrogramme, sa reconstruction et la différence entre les deux. La bande noire pointillée étant une attaque de Bruijn. [Kamkar 2015]. 65

5.1 Aperçu du domicile connecté expérimental 82
5.2 Découpage des jeux de données pour les expérimentations 88
5.3 Erreur jeu de validation - 800-900MHz 98
5.4 Erreur jeu de test - 800-900MHz 98

6.1 Un aperçu des phases de l'approche 108
6.2 Détails du bloc d'estimation de l'erreur de reconstruction 110
6.3 Un spectrogramme (en haut) et sa version pré-traité (en bas) 112
6.4 Un spectrogramme, sa reconstruction et leur différence absolue. La ligne jaune pointillée horizontale correspond à une attaque "de Bruijn" 113
6.5 Détails du bloc de diagnostic temporel et fréquentiel associés aux blocs précédents 114
6.6 Une série temporelle correspondant à la reconstruction (en bleu), le seuil (en rose), l'intervalle réel d'attaque (en orange) et l'intervalle détecté (en vert) 115
6.7 Présentation globale de l'approche et détails du bloc de diagnostic spatial 118

7.1 Environnement connecté 123
7.2 Position des attaques dans l'environnement 126
7.3 Différence de variances des données liées à l'environnement professionnel (bleu) et particulier (rouge) 137

Liste des tableaux

1.1 Synthèse des protocoles présentés 19

5.1 Objets connectés installés et leurs caractéristiques 83
5.2 Listes des attaques injectées 93
5.3 Résultats en fonction du modèle utilisé 99
5.4 Résultats en fonction du modèle utilisé - mis à jour 101

7.1 Objets connectés de la salle d'expérimentation 123
7.2 Classes d'attaques injectées et anomalies non attendues (en orange) 128
7.3 Anomalies détectées 131
7.4 Rappel diagnostic temporel 131
7.5 Précision du diagnostic temporel par bande 132
7.6 Erreur médiane de fréquence 132
7.7 Distance euclidienne médiane pour chaque attaque et position 133
7.8 Distance Euclidienne médiane pour chaque classe d'attaques en chaque position 140

A.1 Paramètres utilisés au sein du manuscrit avec leurs valeurs expérimentales 145
A.2 Notations utilisées dans le manuscrit 146

Introduction

Le déploiement massif des objets connectés, formant l'Internet des Objets ou IoT (pour *Internet of Things)* bouleverse aujourd'hui les environnements réseaux traditionnels. La mise en commun des informations recueillies par d'innombrables objets disséminés dans le monde, ainsi que l'attrait économique de leur exploitation sont les principaux arguments mis en avant par les constructeurs d'objets et les défenseurs de ce concept. Cependant, les technologies liées à l'IoT et l'intégration massive des objets connectés dans des environnements privés, publics et professionnels soulève plusieurs problématiques notamment vis-à-vis de la sécurité informatique. Tout d'abord, ces objets sont souvent conçus et développés très rapidement sans réelle sécurité, et constituent donc une cible de choix pour les attaquants visant à s'introduire dans le réseau interne d'un environnement protégé ou à récupérer des informations personnelles. Ensuite, la grande mobilité et le dynamisme important des objets connectés renforcent encore l'intérêt des attaquants pour ces derniers. Là où les environnements traditionnels possèdent des infrastructures contrôlées assez statiques, possédant des politiques de sécurité bien établies pour faire face aux éventuelles menaces contre le réseau interne, les objets connectés qui s'y ajoutent sont majoritairement mobiles. Ces politiques sont souvent inefficaces pour prendre en compte les potentiels objets corrompus lors d'un déplacement puis lors d'une réintégration dans l'environnement sécurisé. Cette caractéristique de mobilité impose également l'utilisation de protocoles non-filaires, facilitant l'échange d'informations entre les objets. Or, le mode de communication sans-fil impose de nombreuses contraintes du point de vue de la sécurité, puisque l'ensemble des échanges est vulnérable à des écoutes passives de la part d'attaquants cherchant à récupérer des informations sur le contenu des messages. De plus, les objets connectés sont souvent contraints en terme de puissance de calcul et de consommation d'énergie, ce qui ne facilite pas l'implémentation de mécanismes de sécurité coûteux. En outre, il existe de nombreux protocoles sans-fil permettant aux objets de communiquer (ZigBee, Bluetooth, BLE, Z-Wave, etc.). Cette multitude de protocoles ajoute une grande hétérogénéité dans les communications sans-fils : par exemple certains d'entre eux sont propriétaires (notamment ceux utilisés en domotique) et donc difficilement auditables ; il y a également hétérogénéité dans les bandes de fréquences utilisées (comme la bande libre 2.4-2.5 GHz ou les fréquences 433 MHz et 868 MHz).

Les nombreuses problématiques précédentes remettent en question les solutions traditionnelles mises en place pour assurer la sécurité réseau de ces environnements. Aucune d'entre elles n'est en mesure de surveiller la grande diversité des communications sans-fils, tout en assurant la détection des nouvelles menaces et vulnérabilités apportées par ces objets. En effet, la grande majorité des solutions existantes ne se focalise que sur une seule technologie sans-fil, en proposant des mécanismes permettant d'inspecter et d'analyser le contenu des échanges. Même les solutions spécifiques appliquées au domaine de l'IoT ne se concentrent souvent que sur les

échanges effectués entre les objets et l'extérieur du réseau, faisant fi des communications sans-fil réalisées localement entre les objets. Bien que des solutions combinées soient envisageables pour couvrir plus exhaustivement les environnements composés d'un grand nombre d'objets, aussi appelés environnements connectés, celles-ci seraient dans l'incapacité de détecter une tentative d'intrusion exploitant des protocoles non standards ou propriétaires. Il n'est également pas trivial de proposer une solution qui soit en mesure de surveiller plusieurs bandes de fréquences en parallèle, pour détecter des menaces exploitant des protocoles différents. Aussi, il nous semble important de répondre à la question suivante : comment monitorer et surveiller les échanges sans-fil d'un environnement composé d'objets communiquant à l'aide de protocoles aux caractéristiques très hétérogènes ? Que nous pouvons également reformuler ainsi : comment développer une approche qui soit à la fois suffisamment générique pour prendre en compte cette hétérogénéité tout en étant capable de détecter de nouvelles attaques ?

Les travaux menés dans cette thèse visent à répondre à cette question et proposent un système de détection d'intrusion basé sur la surveillance des activités radio, original et adapté aux environnements connectés. Cette approche est basée sur la modélisation du comportement légitime des objets. Cette modélisation utilise la technologie de la radio logicielle pour s'affranchir des spécifications des protocoles de communication en monitorant à l'aide de sondes uniquement les puissances reçues sur de larges bandes de fréquence. Les données mesurées sont ensuite traitées par des algorithmes d'apprentissage automatique pour modéliser les comportements radios légitimes. Les déviations de ces comportements constituent les tentatives d'intrusion.

Cette thèse s'articule autour de quatre parties regroupant les différentes contributions mises en avant par nos travaux. Dans la première partie, nous présentons le contexte associé à ces travaux en nous attardant sur les concepts de l'Internet des Objets. Nous identifions ensuite les problématiques de sécurité liées à l'intégration d'objets connectés dans les environnements traditionnels. Pour mettre en avant le manque de solutions associées, nous nous attardons sur les approches traditionnelles employées dans la sécurité réseau en expliquant les raisons pour lesquelles celles-ci ne peuvent pas assurer une détection efficace des malveillances. Cet état des lieux fixe l'objectif de nos travaux : proposer une solution de sécurité réseau générique, indépendante des spécifications des protocoles, qui soit en mesure de détecter des attaques ciblant les objets connectés. Dans le second chapitre de cette partie, nous présentons donc les deux concepts qui seront utilisés par notre solution pour répondre à cet objectif : l'apprentissage automatique et la radio logicielle (ou *Software-Defined Radio* (SDR)). Nous nous focalisons également sur les solutions spécifiques à l'IoT proposées dans la littérature, et nous en déduisons l'intérêt de la radio logicielle pour monitorer l'ensemble des échanges d'un environnement.

La deuxième partie présente la première contribution de cette thèse, c'est-à-dire la proposition d'une approche générique pour la détection d'anomalies dans les environnements connectés. Nous identifions tout d'abord le modèle de menaces et les

hypothèses de sécurité, obtenus à partir de l'étude des différentes attaques mises en œuvre dans l'IoT. Ensuite, nous présentons le principe et les différentes étapes de fonctionnement de notre approche, ainsi que les deux éléments qui la composent : les sondes radios, basées sur la technologie de la radio logicielle et le système de détection d'intrusion. Une phase d'apprentissage permettant la modélisation des comportements radios dans l'environnement est également présentée. Cette contribution a fait l'objet d'une publication en conférence internationale [Roux 2017].

La troisième partie se focalise sur la deuxième contribution de nos travaux, qui consiste à déployer et évaluer notre approche dans les domiciles connectés. Nous présentons ainsi dans un premier chapitre ce contexte et ses caractéristiques, tout en décrivant l'implémentation spécifique des composants de l'approche. Dans le second chapitre de cette partie, nous proposons la mise en place d'un environnement expérimental réaliste pour évaluer notre approche. Les objets qui le composent sont issus du commerce, et les attaques injectées correspondent à des tentatives d'intrusions visant les objets connectés. Cette contribution a fait l'objet d'une publication en conférence internationale [Roux 2018].

La quatrième et dernière partie ajoute une troisième contribution à nos travaux, en étudiant l'applicabilité de notre solution générique à un environnement plus complexe que les domiciles : les environnements professionnels. Dans le premier chapitre de cette dernière partie, nous présentons les spécificités et identifions l'intérêt de fournir des informations de diagnostic aux utilisateurs de ces environnements, qui contrairement aux domiciles connectés, peuvent posséder une expertise en matière de sécurité. Ces informations sont obtenues via le diagnostic temporel, fréquentiel et spatial d'une anomalie, dont nous présentons l'implémentation. Dans le second chapitre de cette partie, nous réalisons une expérimentation au sein d'un environnement professionnel réel composé de nombreux objets et d'un grand nombre d'utilisateurs et de comportements hétérogènes. Nous présentons ainsi les différents paramétrages de ces mécanismes de diagnostic et leurs résultats vis-à-vis d'attaques réelles injectées dans l'environnement. Avant de conclure cette thèse, nous finissons ce chapitre en discutant des problématiques de déploiement et de vie privée associées à notre solution.

Première partie

Contexte et état de l'art

Chapitre 1

Problématique

Sommaire

1.1 L'écosystème de l'Internet des Objets 7
1.1.1 Environnements d'utilisation 9
1.1.2 Protocoles et communications 13
1.1.3 Caractéristiques des environnements et des objets 18
1.2 La sécurité dans l'Internet des Objets 21
1.2.1 Terminologie de la sécurité 21
1.2.2 Problématiques de sécurité liées aux objets connectés 25
1.2.3 Principales attaques visant les objets connectés 27
1.3 Solutions traditionnelles de sécurité réseau 31
1.3.1 Pare-feu 31
1.3.2 Réseau Privé Virtuel (VPN) 33
1.3.3 Système de Détection/Prévention d'Intrusion (IDS/IPS) . . . 33
1.4 Objectifs de la thèse 35

Ce chapitre présente une vue d'ensemble des concepts liés à l'Internet des Objets (IdO), en se focalisant sur leur utilisation au sein des systèmes informatiques récents et sur l'évolution des mécanismes de communication mis en place. Cette vue d'ensemble nous permet d'exposer les problématiques de sécurité associées. Ce chapitre est divisé en trois sections. Dans la première, nous présentons et définissons l'écosystème IdO, en abordant les aspects historiques, technologiques et de communication. Dans la seconde section, nous présentons la terminologie de la sécurité et son implication au sein du domaine de l'IdO pour en dégager les surfaces d'attaques. La troisième section présente les solutions de sécurité et leurs limitations pour l'IdO. Finalement, la dernière section est consacrée à la présentation des objectifs de la thèse permettant de répondre aux problématiques avancées à l'aide des sections précédentes.

1.1 L'écosystème de l'Internet des Objets

Les évolutions technologiques de ces dernières années ont permis d'imaginer une nouvelle manière de mettre en communication non seulement les personnes, mais également les objets qu'ils utilisent. La première machine à mettre en œuvre cette idée est un distributeur de soda installé dans l'université de Carnegie Mellon (CMU)

au sein du département d'informatique en 1982 [1]. En l'interrogeant à distance via son adresse IP, il était possible de savoir s'il restait des canettes dans la machine. L'Internet des Objets (IdO) ou *Internet of Things* (IoT) était né, même si le terme devait apparaître plus tard, en 2002, dans un article publié dans le journal Forbes [2] rédigé par Chana R. Schoenberger. Depuis, un grand nombre d'objets se sont dotés d'adresses pour communiquer et le terme IoT a été adopté par ces derniers sans définition exacte. Il s'agit en effet plus d'une idée que d'une réalité physique. Toutefois, cette idée s'est développée : connecter non seulement les gens, mais les objets à Internet. La motivation associée au *tout connecté* porte sur la quantité d'information valorisable permettant d'aboutir à une modélisation réaliste du monde à chaque instant. Cette modélisation combinée à l'analyse des données partagées par ces objets, amène au déclenchement automatique d'actions qui agissent sur le monde par le biais d'autres objets. Par exemple, un ensemble de capteurs supervisant l'état du trafic routier en temps réel commanderait le réveil d'un domicile pour adapter l'heure de l'alarme en fonction de l'affluence.

Souvent annoncé comme étant "la nouvelle révolution du Web" [3], ou "Web 3.0", l'IoT a tout pour plaire. Cependant, si le concept d'origine est assurément intéressant, nous sommes aujourd'hui loin d'un écosystème aussi automatisé et unifié que celui imaginé initialement. Actuellement, la majorité des développements IoT consistent à ajouter des capacités de communication à un objet qui en est initialement dépourvu. Cet objet peut alors remonter des données à un serveur ou recevoir des commandes pour interagir avec l'environnement dans lequel il est installé. Cela ne lui permet pas pour l'instant d'être autonome et d'initier des communications avec d'autres objets, de son propre chef, afin de participer à un traitement plus complexe ou généralisé.

Nous voyons donc apparaître une multitude de petits environnements, que nous pourrions associer à des réseaux locaux d'objets plutôt qu'à un vrai Internet des Objets. La seule différence avec un réseau local traditionnel repose sur le fait qu'une grande majorité de ces objets stockent leurs données sur des serveurs dans le *cloud* et donc sur Internet. Ces serveurs proposent souvent un accès complet à ces informations, au détriment de leur protection. En outre, si le traitement et le stockage des données dans le *cloud* sont aujourd'hui pleinement exploités et fonctionnels, les technologies d'interconnexion et d'échanges entre objets sont encore beaucoup trop hétérogènes pour qu'une unification prochaine soit plausible. En effet, pour permettre à des centaines de milliers d'objets différents de communiquer, il faudrait qu'une norme complète s'impose, spécifiant notamment les protocoles de communications et la forme des données, voire même l'architecture matérielle de ces objets. Cependant, les tendances montrent plutôt l'inverse. Ce manque de normalisation dans ces moyens de communications rend difficile l'uniformisation des échanges et limite fortement l'intéractivité possible. Or, l'intérêt économique associé à la possibilité d'acquérir de l'information à partir des données récoltées par des cen-

1. `https://www.cs.cmu.edu/~coke/history_long.txt`
2. `https://www.forbes.com/global/2002/0318/092.html`
3. `https://www.objetconnecte.com/iot-industrielle-1008/`

taines de milliers d'objets poussent les industriels à développer de nombreux objets connectés : selon l'Institut de l'Audiovisuel et des Télécommunications en Europe (IDATE) [4], le nombre de ces objets était estimé à 42 milliards en 2015. Tous les environnements intègrent en effet de plus en plus massivement ces différents objets pour répondre à des besoins spécifiques : les domiciles, les entreprises, les industries, les villes ou les espaces publics.

En conclusion, plutôt qu'un Internet des Objets, nous considérons plus actuelle la notion d'*environnement IoT* ou d'*environnement connecté*. Celle-ci se définissant comme étant un environnement borné composé d'objets, dit objets connectés, ayant la possibilité d'échanger de l'information avec d'autres objets de l'environnement directement ou via Internet. Les caractéristiques de ces objets, leurs rôles, ainsi que leurs moyens de communication dépendent du contexte dans lequel ceux-ci sont utilisés.

1.1.1 Environnements d'utilisation

L'intérêt principal de ces objets est de proposer des services avancés aux utilisateurs, par exemple en proposant d'automatiser les tâches redondantes pour leur faciliter la vie. Par ailleurs, cette automatisation est également intéressante dans des contextes plus industriels, dans lesquels des objets connectés peuvent par exemple tirer avantage d'informations échangées pour s'assurer du bon fonctionnement d'une chaîne d'usinage. Depuis l'apparition dans la littérature du terme IoT, les différents environnements dans lesquels sont installés des objets connectés se retrouvent nommés en fonction de leur contexte d'origine. Le préfixe *smart* est souvent utilisé pour mettre en opposition l'environnement classique, n'employant aucun objet connecté dans son fonctionnement, à celui composé d'un réseau d'objets remplissant des fonctions spécifiques dans ce contexte. La traduction de ce terme en français est "intelligent", appuyant l'idée que l'ajout de systèmes d'automatisation et de communications entre les objets rend l'environnement intelligent, donc plus en mesure de satisfaire les besoins des utilisateurs. Ainsi, les termes comme *smart-home* ou *smart-factory* apparaissent dans tous les articles, scientifiques ou journalistiques [Ahmed 2016], souvent traduits justement par domicile intelligent ou usine intelligente. Bien entendu, la notion d'intelligence avancée consiste à imaginer un environnement dans lequel l'ensemble des systèmes opérant en son sein travaillent en accord pour par exemple améliorer la consommation d'énergie. Ces objectifs dépassent largement l'aspect technologique, et prennent en compte également des dimensions sociales ou organisationnelles qui ne seront pas traitées ici, mais qui étaient déjà discutées dans un article de P. Amphoux en 1988 et 1990 sur la notion d'habitat intelligent [Amphoux 1988, Amphoux 1990]. Dans la suite de ce document, nous préférons le mot "connecté" à une traduction du mot "smart" plus descriptive de l'état de fait actuel et moins générale. Ainsi, les différents environnements seront plutôt traduits par exemple par *domicile connecté* ou *usine connectée*.

4. `https://fr.idate.org/internet-of-things-news2016/`

En nous basant sur les travaux effectués par A. Gani et al. [Ahmed 2016] sur la taxonomie des environnements connectés, nous identifions quatre espaces distincts dans lesquels les objets connectés se sont implantés pour y ajouter de la connectivité : 1) les villes connectées, 2) les bâtiments connectés, 3) les industries connectées et 4) les domiciles connectés. Chacun de ces différents environnements connectés possède ses propres spécificités, notamment vis-à-vis de l'utilisation et du fonctionnement des objets.

1.1.1.1 Villes connectées

Les villes connectées ont connu un élan de popularité dans la littérature au début des années 2000, en présentant l'intégration des *Technologies de l'Information et des Communications (TIC)* dans le contexte urbain. L'idée des *smart cities* est de doter une ville de ces technologies pour créer des espaces publics connectés notamment axés autour des citadins, mais également autour des différents éléments constitutifs de la ville, comme les espaces verts, les axes routiers ou les transports en commun. Déjà avant 2000, Singapour avait commencé à imaginer un plan, nommé IT2000, examinant la mise en place d'une "île connectée" [Choo 1997]. Notamment, l'objectif était de transformer les espaces publics, ainsi que les espaces professionnels et privés, en interconnectant les systèmes informatiques de l'île autour d'une infrastructure commune répondant aux besoins en terme de communications : téléphonie, Internet, multimédia, etc. Les exemples de projets ayant démarré autour de cet objectif dans le début des années 2000 ne manquent pas.

Dans les faits, il s'agit d'utiliser des solutions intégrées, composées de capteurs collectant des informations sur le fonctionnement de tel ou tel domaine, pour ensuite interagir avec d'autres systèmes. Cependant, contrairement à des environnements plus restreints et contrôlés, comme les bâtiments que nous verrons dans la section suivante, l'interconnexion à l'échelle d'une ville est complexe à mettre en œuvre car elle mélange des contextes très différents parfois peu maîtrisés, comme par exemple les espaces privés.

1.1.1.2 Bâtiments connectés

Déjà étudiés depuis les années 2000 dans des environnements de travail comme le Pentagone [Snoonian 2003], les *smart-buildings* ou bâtiments connectés automatisent la gestion du bâtiment. Proposés depuis de nombreuses années sur des éléments spécifiques, comme par exemple pour le système de climatisation ou de sécurité, l'évolution en *smart* dépend surtout de la capacité à concevoir un bâtiment entièrement automatisé dans la gestion de ses différents systèmes. Dans ces environnements, les objets connectés sont essentiellement des réseaux de capteurs communicants avec un système de gestion centralisé, qui s'occupe de traiter les informations relevées par ces capteurs pour solliciter des actionneurs. Par exemple, il pourrait s'agir de fermer automatiquement les portes coupe-feu lors d'un incendie, repéré par des capteurs de fumée ou de température. Les scénarios sont déjà

très bien imaginés dans la littérature, mais nécessitent cependant une capacité d'interconnexion très importante pour permettre à tous ces différents composants de communiquer. Un certain nombre de standards de communications existent déjà pour les bâtiments connectés. D. Snoonian étudiait en 2003 LonWorks et BACnet, deux solutions intégrées pour automatiser la gestion d'un bâtiment, par exemple celui de la Défense à Paris.

En dehors de ces environnements industriels, nous pouvons aussi identifier des bâtiments spécialisés dans lesquels l'IoT se fait aujourd'hui une place, par exemple les hôpitaux connectés [Fuhrer 2006]. En équipant les objets, comme le matériel médical et les médicaments, ainsi que les bracelets des patients, de *tags*, ou d'identifiants radio (*Radio Frequency Identification (RFID)* ou Radio identification en français) et l'hôpital de capteurs et de bornes permettant de lire ces identifiants, il est possible d'obtenir des informations en temps réel sur le fonctionnement de celui-ci. Par exemple, les différents patients peuvent être identifiés pour éviter des erreurs médicales, les stocks de médicaments ou d'équipements peuvent être facilement consultés, etc. D'autres contextes de ce type existent, comme par exemple les écoles ou les universités connectées [Pocero 2017]. En général, les organisations spécialisées peuvent réfléchir à une mise en place de solutions d'automatisation intégrées et unifiées dès la conception ou la construction d'un nouvel espace, encourageant donc l'apparition de bâtiments connectés dans ces contextes.

1.1.1.3 Industries connectées

La notion d'industries connectées rassemble autour de cette idée de multiples concepts : Réseaux de Capteurs Sans-fil (*Wireless Sensor Networks (WSN)*), Machine à Machine (*Machine to Machine (M2M)*), pour arriver à la notion d'IoT. Dans l'ensemble, il s'agit de fournir un mécanisme d'échanges et d'interconnexion entre les différents éléments constitutifs d'une industrie, tel que les machines et les outils, pour permettre de contrôler et de monitorer le fonctionnement complet en temps réel d'un espace industriel. Historiquement, la notion de M2M permettait justement de fournir, via la récupération d'informations de capteurs positionnés sur les machines ou directement par ces machines, une vision et un contrôle sur les procédures d'industrialisation en oeuvre [Latvakoski 2014]. Cependant, les systèmes de communication industriels reposaient sur des solutions majoritairement propriétaires fournies par les constructeurs des machines. La création d'industries connectées, liée à l'Internet des Objets, cherche aujourd'hui à transformer ces systèmes de communication pour les rendre ouverts et standardisés, facilitant ainsi l'intégration de nouveaux éléments.

Aujourd'hui, la mise en connectivité d'une industrie repose donc principalement sur le déploiement d'une architecture de communication standardisée. Plusieurs initiatives récentes ont été proposées ces dernières années pour répondre à cette problématique : INTER-IoT [Ganzha 2017], oneM2M [Swetina 2014], l'idée étant de proposer une méthode de conception d'environnement connecté unifiée et standardisée.

1.1.1.4 Domiciles connectés

Finalement, les derniers environnements intégrant l'IoT sont les domiciles connectés. Dans ces derniers, l'évolution de l'IoT pose aujourd'hui des problèmes difficiles à résoudre. Historiquement, des solutions dites de *domotique* proposaient des systèmes intégrés permettant d'automatiser de nombreuses tâches au sein d'un appartement ou d'une maison. Depuis de nombreuses années, des sociétés comme Somfy [5], ont détaillé et proposé ce type de solution. Le principe est d'intégrer au sein du domicile un ensemble d'équipements, par exemple des stores ou des équipements de sécurité, pouvant être contrôlés à distance via l'usage de télécommandes ou d'un boîtier centralisé. Avec l'apparition et la démocratisation des moyens de communications dits *sans-fil* pour le grand public, la domotique a ensuite évolué pour prendre en compte ces technologies, notamment en retirant les systèmes filaires pour les remplacer par de l'infrarouge ou des radio-fréquences. La domotique s'appuyant notamment sur une notion de confort, voulant rendre le quotidien des utilisateurs plus aisé, les systèmes de communications sans-fil représentent une technologie attirante puisqu'ils permettent le contrôle des systèmes depuis n'importe quel emplacement dans le domicile. Finalement, la démocratisation d'Internet dans les domiciles, et son intégration dans les objets, a facilité l'accès à ces derniers depuis l'extérieur, pour par exemple contrôler les systèmes installés même en cas d'absence, élargissant la notion de "domotique" à la notion de "domicile intelligent" ou *smart home*.

1.1.1.5 Évolutions des usages

L'ensemble de ces environnements présente donc de nombreux intérêts à se voir composer de multiples objets connectés, voire de solutions connectées complètes permettant d'assurer un certain niveau de confort. Lorsque la conception d'un environnement se fait en réfléchissant au préalable à l'intégration de ces objets, il est tout à fait possible de fournir et de proposer une connectivité forte, pouvant amener, en fonction de l'utilisation et des besoins exprimés, à des environnements "intelligents". Cependant, les méthodes de communications, et notamment l'évolution de leurs usages, rendent aujourd'hui infiniment plus complexe la réalisation d'un espace connecté stable, et plus globalement d'un Internet des Objets. En effet, lorsque historiquement, la majorité des échanges de données s'effectuait via des connexions filaires, un objet connecté était une machine fixe, facile à contrôler et à identifier. De plus, la mise en place d'une architecture filaire étant coûteuse et contraignante, le nombre de ces machines était plus restreint. Les environnements étaient donc stables et les différents objets d'un réseau plus clairement identifiables. Cependant, avec le développement du Wi-Fi en 1997 et l'apparition des protocoles de communications mobiles, permettant un accès à Internet depuis n'importe où, le nombre de machines connectées s'est mis à exploser. Les échanges de données sont donc devenus plus mobiles et dynamiques, tout en se multipliant exponentielle-

5. `https://www.somfy.fr/`

ment. En outre, ces dernières années, cette expansion du nombre d'objets connectés a également rencontré une multiplication des méthodes de communication sans-fil proposées, proposant chacune ses propres caractéristiques, comme nous le verrons dans la sous-section suivante. Or, l'usage de nombreuses technologies sans-fil hétérogènes rend ces environnements beaucoup plus instables, et donc, difficilement modélisables et compréhensibles. La réalisation d'une connectivité complète dans ces espaces s'est donc complexifiée.

1.1.2 Protocoles et communications

Cette section a pour objectif de donner un aperçu de cette hétérogénéité des moyens de communications présents dans l'IoT. Tout d'abord nous présentons la définition d'un protocole de communication et le principe de normalisation, puis nous détaillons les différences entre les protocoles filaires et non-filaires, ainsi que leurs avantages et inconvénients. Nous verrons finalement un aperçu de ceux mis en œuvre dans l'IoT.

1.1.2.1 Définitions et normalisation

Au début des années 1970, le développement de réseaux expérimentaux, notamment avec ARPANET [Roberts 1970], fut immédiatement suivi par l'implémentation de réseaux hétérogènes de la part des constructeurs d'ordinateurs. Chacun utilisait ses propres conventions pour interconnecter leurs équipements, présentant ainsi leur architecture réseau. Cependant, le besoin d'interconnecter les systèmes de constructeurs différents a rapidement été ressenti, et un groupe de travail (SC16) a donc été mis en place par l'Organisation Internationale de Normalisation (ISO). L'objectif de celui-ci était de proposer un ensemble de normes régissant l'interconnexion des systèmes ouverts. Ce groupe a proposé en 1979 un modèle d'architecture appelé *Modèle OSI* décrivant les différentes normes régissant la mise en place d'une architecture d'interconnexion pour les systèmes ouverts devant communiquer de l'information [Zimmermann 1980]. Ce modèle aujourd'hui bien connu fonctionne avec une structure en sept couches, permettant notamment de segmenter le problème d'interconnexion, en sous-parties simples. Le principe de ce modèle est le suivant : chaque couche N est indépendante des couches $(N-1)$ et $(N+1)$ situées respectivement en dessous et au-dessus d'elle. Chaque entité fournit à l'entité supérieure des services jusqu'à la couche la plus haute qui permet de fournir des services applicatifs de haut niveau distribués sur l'ensemble des machines du réseau. La transmission des données s'effectue par l'intermédiaire d'un medium d'interconnexion physique qui peut être un composant radio ou filaire. Ce modèle est incorporé sur chaque système du réseau, et chaque couche N communique avec la couche N des autres systèmes. Chacune des couches a donc un rôle qui lui est spécifique, défini ainsi dans la norme :

1. *Couche Physique* : Elle fournit les différentes procédures permettant l'utilisation du medium d'interconnexion physique. Elle gère la transmission des

données physiques sur le medium.

2. *Couche Liaison* : Elle fournit les éléments permettant d'établir, de maintenir et de libérer des connexions de niveau liaison entre des systèmes interconnectés. Elle gère aussi la correction d'erreur pouvant survenir sur le lien physique. Les messages échangés entre deux couches liaison s'appellent des *trames*.
3. *Couche Réseau* : Elle fournit les éléments fonctionnels permettant d'échanger de l'information entre deux machines à travers une connexion réseau. Gère notamment les éléments de routage permettant de transporter les messages de la couche transport à travers le réseau.
4. *Couche Transport* : Elle fournit les services permettant de gérer l'échange entre les entités émettrice et réceptrice de l'information, sans considération sur les éléments intermédiaires permettant d'amener ces données, qui sont gérés par la couche réseau.
5. *Couche Session* : Elle fournit les éléments permettant les échanges d'activités distribuées en respectant la synchronisation et la topologie du réseau. Elle gère les éléments de sessions établies entre deux entités.
6. *Couche Présentation* : Elle fournit les éléments permettant d'uniformiser la forme des échanges, pour permettre l'interprétation des données par la couche application. Elle gère l'affichage et le contrôle des structures de données échangées.
7. *Couche Application* : C'est la couche la plus haute qui fournit les services à l'utilisateur final notamment sous forme d'applications distribuées sur le réseau. Les autres couches permettent de supporter les services de cette couche pour l'utilisateur.

Pour communiquer et définir les spécificités et la manière dont les services de cette couche sont rendus, le modèle OSI définit également la notion de protocole standard ou normalisé. Cette notion définit les éléments d'implémentation permettant de fournir les services de la couche associée [Zimmermann 1980].

Quelques années après la définition du modèle OSI, une autre architecture très similaire est proposée pour permettre l'interconnexion de systèmes ouverts : TCP/IP [Cerf 1989]. Elle propose par la même occasion la définition d'un ensemble de protocoles standards permettant d'implémenter ce modèle dans les systèmes. La différence majeure concerne le regroupement des couches session et présentation dans la couche application, réduisant ainsi la complexité du modèle à cinq couches. Ce modèle a été rapidement adopté, et la pile protocolaire TCP/IP s'est intégrée partout, jusqu'à aujourd'hui où elle est la base du fonctionnement d'Internet. Par la suite, de nombreux standards de protocoles ont été définis, permettant de réaliser les services des différentes couches en répondant à des besoins spécifiques, par exemple en cherchant à réduire la consommation d'énergie. L'apparition de nombreux protocoles sans-fil notamment, et donc reposant sur des couches bas niveaux / matérielles (physique, liaison) spécifiques, a aujourd'hui grandement contribué à rendre les environnements composés d'objets connectés beaucoup plus complexes.

1.1.2.2 Filaires et non-filaires

Les environnements connectés se définissent principalement par leur grande connectivité, associée à la multiplication du nombre d'objets capables de communiquer. Cette multiplication provient principalement d'une expansion rapide du nombre de protocoles sans-fil proposés, répondant à certains besoins spécifiques. La caractéristique non-filaire de ces protocoles renforce la mobilité des objets, qui peuvent se positionner et s'intégrer beaucoup plus facilement dans un environnement qui emploie ces protocoles. Cette notion intervient au niveau des couches les plus basses, c'est-à-dire la couche physique et liaison, puisqu'elle repose sur un medium de communication spécifique.

La notion filaire ou non-filaire est une caractéristique qui dépend de la couche physique qui gère la transmission des données physiques sur le médium. Chacune de ces manières d'échanger de l'information a son lot d'avantages et d'inconvénients. Dans notre cas, nous définissons les caractéristiques suivantes : la fiabilité du lien, la vitesse de transmission et la complexité de l'architecture.

En ce qui concerne la fiabilité du lien, une communication filaire s'effectue via des fils ou câbles, plus résistants aux perturbations physiques de l'environnement. Dans le cas d'une communication sans-fil, les perturbations physiques sont multiples, puisque la transmission s'effectue dans l'air. La qualité des échanges est donc largement dégradée, il faut mettre en place de nombreux éléments permettant d'assurer que les échanges s'effectuent correctement, par exemple en proposant des mécanismes de redondance ou de contrôle d'erreur pour limiter les erreurs.

Concernant la vitesse de transmission, le filaire a également bien plus d'avantages, notamment liés à la fiabilité du lien définie précédemment. La fibre optique par exemple permet d'atteindre des débits extrêmement importants, en utilisant des ondes lumineuses. Aujourd'hui, les débits du sans-fil ne permettent pas d'atteindre ceux d'un lien filaire.

Finalement, l'intérêt premier du sans-fil par rapport au filaire repose sur la dernière caractéristique, c'est-à-dire la complexité de l'architecture. Dans le cas du filaire, l'architecture doit être définie au préalable, les différents raccords filaires permettant de s'intégrer au réseau doivent être réfléchis au préalable, et les entrées/sorties nécessaires pour se raccorder doivent pouvoir être présentes sur le système à connecter. Cette contrainte est un frein à la création d'une architecture de communication dynamique, constituée de multiples objets se connectant et se déconnectant constamment. Dans le cas du sans-fil cependant, le positionnement du système n'a que peu d'importance, puisque celui-ci doit juste être à portée des ondes propagées dans l'environnement. Les objets n'ont également besoin que d'un module radio leur permettant de recevoir et de transmettre de l'information, et celui-ci peut être facilement miniaturisé. L'architecture du réseau est également appelée *topologie*, qui est donc plus facilement flexible en non-filaire qu'en filaire.

Cette dernière caractéristique est celle qui a poussé le développement de protocoles sans-fil de toutes formes. En effet, si les besoins en terme de communication ne nécessitent pas une fiabilité et une vitesse critiques, par exemple si les objets

réalisent des tâches simples et limitées, alors le non-filaire est une alternative très intéressante. C'est notamment le cas pour les capteurs, dont la seule tâche communicative consiste en l'envoi de quelques messages pour remonter de l'information. Finalement, le dynamisme réalisable par le non-filaire présente également un intérêt non négligeable, surtout dans le contexte de l'IoT. En effet, dans le cas d'objets mobiles ou portables, la possibilité d'interconnexion rapide au sein d'un réseau, sans avoir à définir au préalable une infrastructure physique, est une caractéristique importante et beaucoup moins coûteuse à mettre en œuvre.

1.1.2.3 Les protocoles de l'IoT

Dans tous les environnements précédemment décrits, de nombreux protocoles se sont petit à petit imposés, puisqu'ils offrent une facilité d'intégration des nouveaux objets sans modification de l'architecture réseau. L'objectif de cette sous-section est donc de présenter succinctement les protocoles sans-fil utilisés ainsi que leur intégration dans les environnements connectés précédemment décrits, ainsi que leurs intérêts. Nous ne nous intéressons pas ici aux protocoles filaires, qui sont déjà largement étudiés dans la littérature et moins présents dans les environnements IoT.

Nous identifions deux types de protocoles sans-fil : les protocoles *centralisés* et les protocoles *décentralisés* ou *ad hoc*. Généralement, les protocoles centralisés nécessitent la présence d'un point d'accès chargé de transmettre les communications. Ce point d'accès est donc un point central du réseau, par lequel transitent les échanges. Pour résumer, chaque message émis par un élément du réseau est envoyé au point d'accès qui s'occupe ensuite de le transmettre au bon destinataire, en fonction des informations contenues dans celui-ci. À l'opposé, dans les protocoles décentralisés ou *ad hoc* chaque élément du réseau peut échanger de l'information avec un autre élément directement. Comme nous le verrons par la suite, cette distinction complexifie l'observation des échanges.

Protocoles centralisés La famille de protocoles sans fil de type centralisé la plus connue est sans aucun doute le *Wi-Fi*. Cette famille de protocoles est centralisée, puisqu'elle repose, dans sa version originale, sur l'interconnexion des différents éléments avec un point d'accès. Chaque élément du réseau est donc connecté à ce point d'accès, et la communication entre deux d'entre eux s'effectue en faisant transiter les trames par celui-ci. Cette technologie datant de plusieurs décennies est aujourd'hui très implantée dans les différents environnements décrits précédemment. Un autre avantage de cette technologie réside dans sa capacité à intégrer différentes formes de contrôle d'accès en fonction des espaces où elle est implantée. En effet, tous les environnements sont aujourd'hui composés de réseaux Wi-Fi, que ce soit les entreprises, avec des distinctions par exemple entre des réseaux internes réservés aux employés ou à des réseaux publics ouverts aux visiteurs, ou les espaces publics, qui proposent très souvent des espaces Wi-Fi gratuits ouverts à tous. L'immense majorité des domiciles utilise aujourd'hui également cette technologie. Naturellement, les objets connectés l'implémentent donc également, ce qui leur permet de s'interconnecter

rapidement à des réseaux Wi-Fi existants. Les versions plus récentes du standard Wi-Fi autorisent aujourd'hui également des fonctionnements alternatifs décentralisés mais qui sont encore assez peu utilisés. Cette famille de protocoles utilise souvent la bande de fréquence 2.4-2.5 GHz, mais est également présente avec de nouvelles spécifications sur la bande 5 GHz. Une spécification récente (802.11ad) présente même la possibilité de communiquer à l'aide d'ondes millimétriques, c'est-à-dire en émettant des données sur une bande dépassant les 30 GHz [Cordeiro 2010].

Une autre famille de protocoles bien connue est celle des réseaux de téléphonie mobile, c'est-à-dire les protocoles utilisés pour permettre l'échange de données et de communications entre des appareils mobiles comme les téléphones. Ces protocoles ont évolué selon plusieurs "générations", qui redéfinissent les spécifications de la pile protocolaire. Pour résumer, des antennes sont positionnées sur le territoire, puis chacune d'entre elles va couvrir une zone géographique avec laquelle les utilisateurs à portée pourront communiquer au travers de celles-ci. Le fonctionnement des communications est donc bien centralisé. L'objectif de la prochaine 5ème Génération est également de faire converger l'infrastructure mobile pour les objets connectés, en leur permettant de s'interconnecter à longue distance [Li 2018]. Ces protocoles communiquent sur des bandes de fréquence diverses qui dépendent des générations. Les bandes les plus utilisées dans les systèmes récents sont les fréquences 800-900 MHz, 1500 MHz, 1800 MHz, 2100 MHz et 2600 MHz.

Toujours dans les protocoles plutôt longues distances, deux protocoles assez récents proposent une vision différente des communications, plutôt axée sur la remontée d'information épisodique à distance. Ces deux protocoles s'appellent LoRaWAN et SigFox, et proposent tous les deux des communications longue portée mais bas débit souvent catégorisés comme LPWAN (*Low-Power Wide-Area Network*. L'idée de ces protocoles est d'être extrêmement peu coûteux en puissance, en limitant drastiquement le débit, et donc la quantité d'informations échangées. Pour LoRaWAN, l'ensemble des équipements communiquent à travers des passerelles à portée, sur le même principe que les communications mobiles. Le protocole IP est utilisé pour permettre à ces passerelles de communiquer entre elles via un serveur applicatif. Quant à SigFox, c'est un protocole propriétaire fonctionnant sur le même principe que LoRaWAN, avec des différences notables au niveau des couches bas niveau utilisées. Ces protocoles communiquent via la bande de fréquence de 868 MHz, qui est une bande radio libre d'utilisation.

Protocoles décentralisés ou ad hoc Les environnements IoT reposant sur une forte connectivité et un important dynamisme entre les différents éléments qui les composent, les protocoles ad hoc présentent des caractéristiques intéressantes favorisant leurs récents développements. Ces derniers peuvent en effet directement échanger de l'information sans passer par une passerelle qui nécessiterait une infrastructure spécifique. Avec les objets connectés, un grand nombre de protocoles ont été proposés, le plus connu étant sans aucun doute Bluetooth et sa variante *Low-Energy* ou basse consommation, *Bluetooth Low Energy (BLE)*. Présent de-

puis déjà quelques années sur les téléphones mobiles qui l'utilisaient principalement pour l'échange de données multimédia à courte portée, il s'est développé dans sa version basse consommation pour les objets connectés. Des centaines d'objets l'implémentent donc, permettant aux téléphones équipés de cette technologie de directement communiquer avec eux. Son fonctionnement au niveau de la couche physique est assez particulier, puisqu'il partage la bande 2.4-2.5 GHz avec le Wi-Fi et d'autres protocoles. Il utilise une modulation en sauts de fréquence, ce qui permet d'éviter les interférences [6]. Aujourd'hui, la topologie du Bluetooth et du BLE est souvent en point-à-point.

Zigbee est une autre pile protocolaire décentralisée qui propose quant à elle une topologie en réseau maillé, basée sur la norme 802.15.4 spécifiant la couche physique et liaison. L'objectif est similaire à BLE : proposer un protocole basse consommation pour les communications courte portée. Ce protocole est très utilisé dans la domotique moderne, puisqu'il permet d'interconnecter assez facilement un réseau de capteurs dans un domicile. Il fonctionne sur des bandes différentes en fonction de son pays d'utilisation, 868 MHz en Europe, 915 MHz sur le continent américain et l'Australie, il est également utilisable dans le monde sur la bande 2.4-2.5 GHz.

Les exemples présentés ici ne sont qu'un minuscule aperçu de la quantité impressionnante de protocoles proposés pour interconnecter de manière sans-fil des objets, mais permettent d'apprécier la grande complexité et diversité des technologies existantes. En parallèle de tous les protocoles présentés ici, nombre d'entre eux sont propriétaires, leur spécification étant inconnue du grand public, à l'instar de SigFox. Il s'agit souvent, comme lors des débuts de l'interconnexion des systèmes ouverts [Zimmermann 1980], d'entreprises privées qui développent leur propre protocole de communication pour les objets qu'ils vendent. C'est par exemple le cas de Nest Labs, qui utilise un protocole propriétaire appelé Thread [7]. Les spécifications de ces protocoles n'étant pas disponibles au grand public, il est très difficile d'en comprendre le fonctionnement, aboutissant à de nombreuses problématiques d'interconnexion et de surveillance des communications. Pour évaluer succinctement la diversité des protocoles sans-fil existants dans le monde, le site Sigidwiki [8] recense l'ensemble des signaux identifiés et non identifiés ainsi que des descriptions plus ou moins précises de leur fonctionnement. En ne comptant que les signaux identifiés, il en existe déjà 379 différents.

Le tableau 1.1 récapitule les protocoles ayant été présentés au cours de cette section, ainsi que les fréquences sur lesquelles ils opèrent.

1.1.3 Caractéristiques des environnements et des objets

Ayant un aperçu un peu plus clair du fonctionnement de l'IoT et des spécificités, notamment des protocoles de communication, des objets et des environne-

6. https://fr.wikipedia.org/wiki/étalement_de_spectre_par_saut_de_fréquence
7. https://www.threadgroup.org/What-is-Thread/Overview
8. https://www.sigidwiki.com/wiki/Signal_Identification_Guide

Table 1.1 – Synthèse des protocoles présentés

Protocole	Fréquence
Wi-Fi	2.4-2.5 GHz, 5 GHz
Zigbee	868 MHz, 915 MHz, 2.4-2.5 GHz
SigFox	868 MHZ, 902 MHz
Bluetooth	2.4-2.5 GHz
LoRaWAN	433 MHz, 868 MHz, 915 MHz
Protocoles cellulaires	800-900 MHz, 1.5 GHz, 1.8 GHz, 2.1 GHz, etc.
Protocoles divers UHF	433-434 MHz, 868-869 MHz
Bluetooth Low Energy (BLE)	2.4-2.5 GHz

ments connectés, nous sommes à présent en mesure d'en définir les caractéristiques. Celles-ci nous seront utiles pour identifier les besoins et les problématiques de ces environnements.

La première caractéristique est **l'hétérogénéité** des technologies utilisées au sein d'un même environnement connecté. Comme décrit précédemment, les protocoles implémentés au niveau des couches, notamment basses, sont nombreux et fonctionnent de manière très spécifique, ne serait-ce que dans leur utilisation du medium de communication. Ainsi, un certain nombre de problématiques peuvent se poser par la suite dans l'interconnexion des différents objets ou dans la surveillance des communications. Ensuite, même si cela n'a pas été détaillé précédemment, les architectures matérielles des objets utilisés dans un environnement sont également fortement impactées par cette hétérogénéité. Chaque objet possède souvent sa propre architecture, avec ses spécificités et ses composants. L'architecture la plus utilisée aujourd'hui dans les objets connectés et les appareils mobiles est sans aucun doute ARM [9]. Cependant, celle-ci évolue rapidement, et les processeurs qui l'implémentent sont nombreux. Nous voyons donc apparaître au sein des environnements des objets ayant des architectures très hétérogènes. Concernant les composants, ils réalisent leurs objets à partir de composants sur étagère ou *Commercial Off-The-Shelf (COTS)*, c'est-à-dire à partir de produits fabriqués en série plutôt que spécifiques à un projet. Un des avantages de ce concept est la limitation des développements spécifiques et donc de l'hétérogénéité. Cependant, un grand nombre de *COTS* différents sont développés puis proposés à la vente, et les objets intègrent donc ces composants en fonction de leurs propres besoins.

La seconde caractéristique qui a déjà été légèrement discutée est la **mobilité** des objets connectés, rendant les environnements connectés **dynamiques**. La mobilité définie l'aspect portable, qui est une caractéristique intrinsèque des objets connectés. En effet, l'intérêt de l'IoT étant d'interconnecter n'importe quels objets, ces derniers peuvent être des appareils mobiles (*smart-phones*, *smart-watches*, etc.). Les environnements doivent donc être en mesure d'adapter leurs architectures réseaux rapidement pour prendre en compte la disparition ou l'apparition d'objets

9. `https://www.arm.com/`

désirant s'interconnecter, d'où le **dynamisme** de ces environnements.

La caractéristique suivante est liée aux fonctions réalisées par les objets connectés. Ces derniers réalisent pour la plupart une tâche unique et identifiable, et doivent limiter leur consommation d'énergie pour être en mesure de fonctionner pendant longtemps sur batterie. Par exemple, une fourchette connectée [10] a une tâche précise, celle de fournir des informations sur la vitesse de mise en bouche, et il ne paraît pas intéressant d'avoir à constamment recharger celle-ci entre chaque repas pour réaliser cette fonction. Le **minimalisme** dans la conception, la fabrication et l'usage est donc une caractéristique essentielle au fonctionnement de ces objets connectés. Bien entendu, dans des environnements plus industriels et parfois dans les domiciles, des machines branchées au secteur peuvent être connectées, mais l'objectif de limitation des frais d'énergie est toujours mis en avant.

Deux caractéristiques supplémentaires peuvent être directement déduites de l'explosion du nombre d'objets connectés produits depuis quelques années. Tout d'abord, ce **nombre conséquent d'objets** est une caractéristique à part entière, puisque contrairement à des environnements traditionnels avec un nombre de systèmes limité, des problématiques de scalabilité, c'est-à-dire d'adaptation à des changements d'ordre de grandeur, peuvent se poser. Ensuite, cette expansion rapide rend le marché de l'IoT extrêmement concurrentiel. Les fabricants essayent donc d'être les premiers à produire et vendre l'ajout de connectivité sur un objet. Cela a une conséquence regrettable : l'hétérogénéité des technologies est accentuée, puisque la conception d'un objet est limitée à ses fonctionnalités et à son ergonomie, en délaissant les aspects d'interconnexion et d'intégration dans un environnement plus global. En outre, une grande majorité des fabricants de ces objets n'étaient pas, ou ne sont pas des experts en conception de produits informatiques ou connectés. Ce manque d'expertise et de compétences des fabricants rend les objets *peu fiables* sur beaucoup d'aspects, notamment en terme de sécurité comme nous le verrons. Nous définissons donc cette cinquième caractéristique comme étant la **conception fragile** des objets connectés.

Finalement, la sixième et dernière caractéristique concerne la fonctionnalité de base de quasiment tous les objets connectés. L'objectif principal de ces derniers est souvent présenté comme étant un moyen de rendre des services à ses utilisateurs via l'interconnexion ou l'automatisation de certaines actions. Cependant, pour réaliser cet objectif, les objets connectés collectent des informations, puis les échangent pour agir sur leur environnement. Cette collecte exploite souvent des données révélant des informations concernant les utilisateurs ou cet environnement, rendant les objets connectés **sensibles en terme de vie privée**. Outre l'aspect protection de la vie privée souvent peu respecté par les fabricants, cette caractéristique fait de ces objets des cibles privilégiées pour des personnes malveillantes.

10. `http://www.slowcontrol.com/`

1.2 La sécurité dans l'Internet des Objets

Un aspect très souvent exposé et discuté dans les objets connectés, et plus généralement dans l'Internet des Objets, est celui de la sécurité. Beaucoup d'articles scientifiques, mais également de presse, insistent sur le manque de sécurité de ces nombreux objets. Au vu des caractéristiques présentées précédemment ainsi que des réflexions autour des environnements connectés, nous jugeons également qu'un certain nombre de problématiques de sécurité peuvent se poser vis-à-vis de l'intégration d'objets dans un environnement. Cette section présente donc dans un premier temps les éléments de définition nécessaires pour comprendre ce qu'est la sécurité informatique. Ensuite, nous décrivons les surfaces d'attaques des environnements connectés et des objets qui le composent, et ce en fonction des caractéristiques présentées précédemment. Enfin, les problématiques de sécurité associées à l'Internet des Objets pouvant être déduites de nos connaissances sont exprimées dans une dernière sous-section.

1.2.1 Terminologie de la sécurité

La sûreté de fonctionnement définit les termes de sécurité-innocuité (safety) et sécurité-immunité (security), dans laquelle nos travaux s'inscrivent. Nous présentons ici le vocabulaire qui sera utilisé par la suite, en commençant par définir les principaux concepts de la sûreté de fonctionnement, issus de [Laprie 1996] et mis à jour dans [Avizienis 2004], puis en se focalisant sur les concepts liés à la sécurité-immunité.

1.2.1.1 Sûreté de fonctionnement

La sûreté de fonctionnement d'un système informatique est définie comme « la propriété qui permet aux utilisateurs d'un système de placer une confiance justifiée dans le service qu'il leur délivre ». Le service délivré correspond au comportement du système perçu par ses utilisateurs. La sûreté de fonctionnement comporte trois axes principaux : les attributs qui la décrivent, les entraves qui empêchent sa réalisation et les moyens d'atteindre celle-ci (figure 1.1).

La sûreté de fonctionnement associée à la sécurité-immunité définissent les attributs et les propriétés complémentaires suivantes :

— La capacité d'un système à être prêt à l'utilisation se définit comme la **Disponibilité**.

— La continuité du service se définit comme la **Fiabilité**.

— La non-occurrence de conséquences catastrophiques pour l'environnement se définit comme la **Sécurité-innocuité**.

— La non-occurrence de divulgations non autorisées de l'information se définit comme la **Confidentialité**.

— La non-occurrence d'altérations inappropriées de l'information se définit comme l'**Intégrité**.

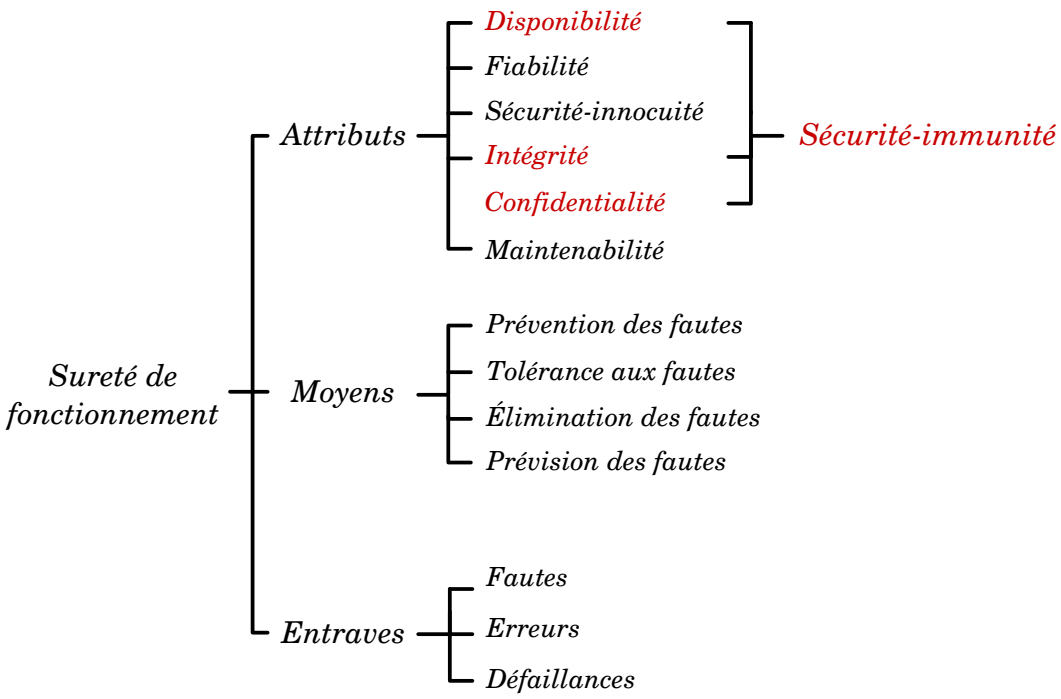

FIGURE 1.1 – L'arbre de la sécurité de fonctionnement

— L'aptitude d'un système à être réparé ou à subir des évolutions se définit comme la **Maintenabilité**.

La non-sûreté de fonctionnement correspond quant à elle à une perte de confiance, qui ne peut plus ou ne pourra plus être placée dans le service délivré. Les causes ou résultats de celle-ci sont les circonstances indésirables définies comme étant les *entraves* :

— Une **défaillance** survient lorsque le service délivré dévie de l'accomplissement de la fonction du système.

— Une **erreur** est la partie de l'état du système qui est susceptible d'entraîner une défaillance.

— Une **faute** est la cause adjugée ou supposée d'une erreur.

Pour mettre en place la sûreté de fonctionnement dans un système, nous disposons de plusieurs *moyens*. Ces moyens sont des méthodes et techniques permettant de fournir au système l'aptitude à délivrer un service conforme à l'accomplissement de sa fonction, et de donner confiance dans cette aptitude. Le développement d'un système sûr passe par l'utilisation combinée d'un ensemble de méthodes :

— **La prévention de fautes** empêche l'occurrence ou l'introduction de fautes.

— **La tolérance aux fautes** permet de fournir un service à même de remplir la fonction du système en dépit des fautes.

— **L'élimination des fautes** réduit la présence (nombre, sévérité) des fautes.

— **La prévision des fautes** estime la présence, la création et les conséquences des fautes.

Les travaux décrits et présentés dans ce manuscrit s'inscrivent dans le cadre de la sécurité-immunité, qui se définit comme l'association des attributs de disponibilité de confidentialité et d'intégrité. Dans la suite, nous détaillons les concepts liés à la sécurité-immunité.

1.2.1.2 Sécurité-immunité

La sécurité-immunité a pour but de protéger un système contre les fautes définies comme intentionnellement nuisibles, c'est-à-dire créées ou commises délibérément pour nuire, appelées aussi *malveillances*. Dans cette section, nous allons appliquer les attributs génériques définis précédemment au contexte de la sécurité-immunité. Dans le reste de ce manuscrit, les termes de sécurité ou de sécurité informatique seront utilisés comme alias à la sécurité-immunité, sauf exception explicitement précisée.

Attributs Dans le contexte de la sécurité-immunité, les attributs de la sûreté de fonctionnement peuvent être spécialisés de la manière suivante :

— **Disponibilité** : prévention de rétentions d'information non autorisées.

— **Confidentialité** : prévention de divulgations d'information non autorisées.

— **Intégrité** : prévention de modifications d'information non autorisées.

Malveillances Le projet MAFTIA [Powell 2001] s'intéresse au concept de faute due à l'homme pour la sécurité-immunité. Dans nos travaux, nous nous concentrons principalement sur ces fautes, et notamment à deux classes de fautes intentionnellement nuisibles, appelées également *malveillances* : les logiques malignes et les intrusions. Les logiques malignes sont des fautes internes intentionnelles, qui sont conçues pour provoquer des dégâts (bombes logiques) ou pour faciliter les futures intrusions par l'ajout de vulnérabilités (portes dérobées). Celles-ci peuvent être présentes dès la première utilisation du système, donc ajoutées par le concepteur du système, ou durant son exploitation par l'installation d'un cheval de Troie ou par un virus. Concernant les intrusions, celles-ci sont associées à deux causes :

— Un acte malveillant ou une *attaque* essayant d'exploiter une faiblesse du système. Une attaque étant une faute d'interaction, dont le but est de violer un ou plusieurs des attributs de sécurité. Elle peut être aussi définie comme une tentative d'intrusion.

— Une faiblesse ou une *vulnérabilité* placée dans les exigences, la spécification, la conception ou la configuration du système, ou dans la manière dont il est utilisé. Une vulnérabilité étant une faute accidentelle, ou une faute intentionnellement malveillante ou non malveillante.

L'intrusion se définit donc comme étant une faute externe nuisible qui résulte d'une attaque ayant réussi à exploiter une vulnérabilité.

Fautes dues à l'homme sans volonté de nuisance Le projet MAFTIA décrit également deux autres types de fautes dues à l'homme n'étant pas des malveillances, car sans volonté de nuisance, qui peuvent cependant être génératrices de vulnérabilités ou d'intrusions dans les systèmes :

— Les fautes de conception sont ajoutées à la conception du système, ce sont des fautes de développement accidentelles ou intentionnelles sans volonté de nuire. Par exemple, l'implémentation d'un standard cryptographique ou d'un protocole communication peut être vulnérable [Ronen 2018].

— Les fautes d'interactions sont des fautes externes, dues à la mauvaise utilisation du système. Par exemple, l'utilisateur d'un objet n'utilise pas les fonctionnalités de sécurité proposées par le fabricant pour des questions d'ergonomie.

Les fautes dues à l'homme sans volonté de nuisance ajoutent principalement des vulnérabilités au système et permettent donc, par transitivité, la réalisation d'attaques voire d'intrusions qui pourront par la suite amener des logiques malignes en ajoutant de nouvelles vulnérabilités. Les fautes d'interaction humaines, notamment dans l'IoT, peuvent être induites par une conception non sécurisée d'un objet, par exemple en proposant un mode de fonctionnement sécurisé ainsi qu'un mode dégradé (c'est par exemple le cas dans des implémentations du BLE ou du Bluetooth). Ces fautes, bien que très sensibles pour la sécurité des systèmes, ne font pas l'objet de ces travaux, car elles dépendent souvent de la formation des utilisateurs. Pour ce qui est des vulnérabilités ajoutées par les fautes de conception, elles seront potentiellement exploitées de manière malveillante par des attaques. Il est donc indispensable de lutter contre les malveillances pouvant aller à l'encontre des trois attributs de la sécurité-immunité.

1.2.1.3 Moyens de lutte contre les malveillances

Nous pouvons encore une fois spécialiser les moyens permettant d'avoir un système sûr de fonctionnement pour les appliquer à la sécurité-immunité.

Prévention des fautes La prévention des fautes peut se dériver en trois méthodes pour les attaques, les vulnérabilités et les intrusions. La prévention des attaques consiste à dissuader les utilisateurs malveillants d'attaquer le système. Cela est possible via la loi et la pression sociale par exemple. La prévention des vulnérabilités lutte contre l'introduction de vulnérabilités dans la conception du système via l'application de méthodes semi-formelles ou formelles, mais aussi l'éducation de l'utilisateur (choix de mot de passe robuste, utilisation correcte d'un certificat ou d'une paire de clés par exemple). La prévention d'intrusions est mise en place via les techniques d'authentification, d'autorisation et des pare-feu et par transitivité aux préventions des attaques et des vulnérabilités.

Tolérance aux fautes Dans le cas de malveillances, on s'intéresse principalement à la tolérance aux intrusions. Il s'agit d'un ensemble de méthodes que met en

place un système pour être capable de détecter une intrusion, de se réparer et se reconfigurer tout en continuant de garantir une disponibilité de service et/ou l'intégrité des données pendant une attaque [Deswarte 1991]. Nous pouvons aussi citer les mécanismes de redondance ou de diversification, qui garantissent qu'un certain niveau de sécurité peut être assuré malgré la compromission de certains composants du système. Nos travaux se concentrent sur cet aspect, en détectant les intrusions via l'analyse des activités radios d'un environnement connecté à surveiller.

Élimination des fautes Seule l'élimination de vulnérabilités est réellement pertinente. En effet, on ne peut interdire formellement à un utilisateur d'attaquer un système et à l'identique, si une vulnérabilité existe, une intrusion peut de toute façon exister. Le développement d'un système est complexe, il est donc très difficile d'éliminer complètement les fautes d'un système. Les concepteurs peuvent donc tenter de réduire leur nombre en appliquant, entre autres, de la vérification, du diagnostic et de la correction. La vérification consistant à déterminer si le système satisfait des propriétés, le diagnostic étant la recherche des fautes ayant empêché la satisfaction des propriétés, et la correction correspondant aux modifications nécessaires.

Prévision des fautes Enfin, la prévision de fautes propose un ensemble de méthodes permettant d'identifier les vulnérabilités, attaques, intrusions potentielles d'un système et de mesurer l'impact des erreurs sur les attributs de la sécurité-immunité sur le système, tout en reportant ces identifications.

1.2.1.4 Surfaces d'attaques et modèle de menaces

Pour permettre de mettre en place les moyens associés à un système sûr de fonctionnement, il est nécessaire de comprendre deux éléments constitutifs du système que nous cherchons à protéger :

— **Modèle de menaces** : définit les menaces potentielles, de telle manière que les vulnérabilités d'un système sont identifiées pour proposer des moyens permettant de prévenir, tolérer, éliminer ou prévoir des malveillances.

— **Surface(s) d'attaques** : définit les vecteurs d'attaques, c'est-à-dire les points d'accès extérieurs au système pouvant contenir des vulnérabilités qu'un attaquant pourrait exploiter pour réaliser une intrusion.

1.2.2 Problématiques de sécurité liées aux objets connectés

Les différentes caractéristiques de l'IoT évoquées précédemment, ainsi que les éléments de définition concernant la sécurité-immunité nous permettent à présent d'identifier les différentes problématiques de sécurité liées aux objets et environnements connectés.

Tout d'abord, la conception fragile des objets est une des caractéristiques générales à l'IoT qui pose le plus de problèmes du point de vue de la sécurité. Les fabricants n'ayant pas les compétences informatiques nécessaires pour réaliser des

systèmes sûrs de fonctionnement, les objets intégrés aujourd'hui dans les environnements connectés sont construits à partir de *COTS* et d'implémentations non robustes qui peuvent poser des problèmes de sécurité. De plus, les méthodes de conception et de développement sécurisé, généralement maîtrisées dans le monde de l'informatique traditionnel laissent à désirer dans les systèmes connectés. Ainsi, en prenant des exemples comme l'implémentation d'une pile protocolaire Bluetooth, nous voyons souvent apparaître des vulnérabilités basiques qui ne respectent pas les attributs de la sécurité-immunité. C'est par exemple le cas avec le ver Mirai [Antonakakis 2017], qui exploitait des serveurs Telnet avec authentification faible pour s'introduire dans les systèmes et en modifier le comportement pour se répandre..

Directement liée à cette première caractéristique, l'hétérogénéité renforce les problématiques de sécurité pouvant se poser dans les objets connectés. En effet, à cause d'implémentations non-sûres, la forte proportion d'objets utilisant des technologies différentes peut amener des vulnérabilités distinctes sur les objets qui composent un environnement. La prévision de ces fautes via des moyens de détection est très difficile à mettre en oeuvre, puisque ces moyens doivent être suffisamment génériques pour prendre en compte cette forte hétérogénéité. En outre, les systèmes sont très vulnérables et les surfaces d'attaques se multiplient. À la fois sur le matériel, mais également sur les interfaces de communications utilisées, qui sont les points d'accès les plus visibles pour un attaquant, notamment dans le sans-fil.

Le sans-fil largement exploité dans les objets connectés est donc également une caractéristique pouvant poser un certain nombre de problèmes de sécurité. En effet, tel que détaillé auparavant, les protocoles non-filaires utilisent des ondes émises dans l'air pour transmettre de l'information. Dans ce cas, il peut être aisé pour un attaquant de surveiller les communications pour obtenir des informations. Même si les protocoles implémentent pour la plupart des standards cryptographiques permettant d'assurer la confidentialité, la conception fragile des objets évoquée précédemment ne garantit pas toujours cette propriété. Concernant les différentes familles de protocoles, dans le cas d'un fonctionnement centralisé, l'usage d'une passerelle permet d'ajouter certains moyens de luttes : les communications devant transiter par un point d'interconnexion, celui-ci peut utiliser des mécanismes de tolérance ou de prévision contre les malveillances. Au contraire, lors d'un fonctionnement décentralisé, les objets interagissent directement entre eux, et doivent donc intégrer directement ces moyens de protection, ce qui n'est pas toujours possible en raison du minimalisme des objets.

La caractéristique du minimalisme pose un certain nombre de contraintes en terme de moyens pouvant être mis en œuvre pour lutter contre les malveillances. En effet, une forte proportion d'objets ne possèdent pas les caractéristiques minimales en terme de puissance ou d'architectures pour permettre d'y intégrer des moyens de luttes comme la tolérance ou de la prévision. Il faut donc souvent faire reposer la sécurité d'un environnement composé de plusieurs objets sur une solution externe à ces derniers, qui n'a pas la possibilité de contrôler ou d'agir sur ces objets.

Le dynamisme des environnements, lié à la grande mobilité des objets qui les composent, pose également de nombreux problèmes de sécurité. En effet, les objets connectés ont aujourd'hui tendance à être déplacés et manipulés dans plusieurs environnements aux contraintes différentes. Cette pratique est appelée *Bring Your Own Devices (BYOD)*, pour "prenez vos appareils personnels". Dans le cas d'un domicile, le risque et les conséquences d'une intrusion étant souvent considérés comme faibles, une grande partie de la sécurité repose sur le point d'entrée des communications ou sur les objets eux-mêmes. Dans un environnement professionnel, ces risques et conséquences sont bien plus élevés, et une intrusion peut plus facilement engendrer de lourdes conséquences. Cependant, si un usager de ces deux environnements applique la pratique *BYOD*, des intrusions peuvent être menées sur les objets de son domicile, voire dans les transports, dans le but de les corrompre. Ces objets corrompus pourront ensuite être intégrés dans l'environnement professionnel, qui sera fragilisé.

Enfin, la dernière problématique qui nous impose aujourd'hui une réflexion sécurité autour des objets connectés est celle de l'intérêt porté par les attaquants sur ces derniers. Une des caractéristiques mise en avant dans la section 1.1.3 est la nature sensible des informations personnelles manipulées par ces objets. En effet, ces informations sont très intéressantes aux yeux des attaquants, qui peuvent les utiliser à d'autres fins, par exemple pour connaître les habitudes d'un habitant avant un cambriolage, ou pour récupérer des informations d'authentification. En couplant cela avec les différentes problématiques préalablement présentées, ces objets deviennent des cibles de choix pour les attaquants, qui peuvent à moindre coût mettre en défaut les propriétés de disponibilité, d'intégrité et de confidentialité.

Ces problèmes de sécurité peuvent se résumer au sein d'une seule problématique : comment s'assurer que les propriétés de confidentialité, intégrité et disponibilité nécessaires à la sécurité-immunité soient respectées dans des environnements connectés malgré la présence d'objets vulnérables, potentiellement infectés avant leur intégration, mobiles, utilisant des moyens de communications sans-fil hétérogènes vulnérables vis-à-vis de la confidentialité, n'ayant pas toujours les caractéristiques physiques pour mettre en oeuvre des moyens de protection et qui sont des cibles de choix pour les attaquants ?

Pour commencer à répondre à cette problématique, nous devons tout d'abord identifier et comprendre les spécificités des attaques visant ces objets vulnérables. La section suivante de ce chapitre présente donc un panorama des attaques ciblant les environnements connectés, en se focalisant sur celles visant les protocoles de communication.

1.2.3 Principales attaques visant les objets connectés

1.2.3.1 Classifications proposées

Plusieurs classifications ont été proposées dans la littérature pour classer ces attaques selon leurs spécificités. Nous pouvons en trouver deux types : les classifica-

tions classiques, assez proches de celles employées dans l'informatique traditionnelle, et les classifications plus récentes qui se focalisent sur les spécificités des objets connectés. Dans le cas des premières, le *Open Web Application Security Project* (OWASP) *Internet of Things* [OWASP 2017], par exemple, classifie les attaques selon le type de vulnérabilités exploitables. Ce projet définit également les différentes surfaces d'attaques de l'IoT, telles que l'interface Web, le réseau, etc., ainsi qu'un certain nombre d'informations sur les problématiques de sécurité liées à ce domaine. Pour les secondes, qui sont plus adaptées de notre point de vue aux spécificités de l'IoT, E. Ronen et al. [Ronen 2016] propose une classification basée exclusivement sur l'objectif d'impact des intrusions sur les fonctionnalités des objets :

— Ignorer la fonctionnalité : l'attaquant ne considère l'objet que comme un objet connecté à un réseau, et cherche à l'utiliser pour réaliser des attaques, par exemple un réseau de bot (botnet) pour une attaque distribuée de déni de service (*Distributed Denial of Service* (DDOS)) [11].

— Réduire la fonctionnalité : l'attaquant cherche à empêcher ou supprimer la fonctionnalité de l'objet, par exemple en coupant le refroidissement d'un réfrigérateur.

— Manipuler la fonctionnalité : l'attaquant utilise le ou les fonctionnalités de l'objet dans un objectif malveillant, en modifiant ou inversant le comportement attendu d'un objet. Par exemple, utiliser l'ouverture des volets automatiques à distance pour réaliser une intrusion physique.

— Étendre la fonctionnalité : l'attaquant étend la fonctionnalité de l'objet pour réussir à réaliser un objectif complètement différent via cette fonctionnalité étendue. Ronen et al. démontrent ce type d'attaque via l'utilisation des changements de luminosité d'une ampoule pour faire fuiter de l'information via un canal caché.

L'avantage de cette classification repose sur l'interprétation directe des conséquences des intrusions telles que perçues par le ou les utilisateurs finaux (e.g. la porte s'ouvre, la température augmente, etc.)

1.2.3.2 Attaques ciblant les protocoles sans-fil

Nos travaux visant à détecter les attaques impactant les communications réseaux de ces objets, cette section propose de donner un aperçu de celles répertoriées dans la littérature. Pour les présenter, nous avons choisi de nous focaliser sur la surface d'attaques des protocoles sans-fil, et de classer ces attaques par protocole de communications utilisé.

***Bluetooth Low Energy* (BLE)** Les protocoles récents tel que BLE sont sans-aucun doute les plus touchés par des vulnérabilités, puisque leur utilisation suscite

11. Un réseau de bots contrôlé à distance par un attaquant pour réaliser des déni-de-service sur des infrastructures conséquentes, par exemple en envoyant des milliers de requêtes de connexion pour surcharger un serveur. [Antonakakis 2017]

un intérêt accru de la part des chercheurs en sécurité et des attaquants. En effet, S. Jasek [Jasek 2016] décrit plusieurs attaques sur le BLE (*Bluetooth Low Energy*) réalisées via un outil nommé Gattacker, lui permettant d'intercepter, de rejouer, ou d'injecter des messages entre deux objets, en réalisant une attaque de l'homme du milieu (*Man-in-the-Middle* (MitM)). Cette dernière permet de se placer en tant qu'attaquant entre deux entités en communication, pour pouvoir contrôler leurs échanges. De son coté, D. Cauquil [Cauquil 2016] présente une solution similaire appelée *Btlejuice* qui permet de réaliser des attaques MitM sur des objets communiquant en BLE. Les premiers travaux focalisés sur les vulnérabilités présentes dans la spécification du Bluetooth datent de 2013, réalisés par M. Ryan [Ryan 2013]. Dans cet article, il décrit le manque de sécurité associé au mécanisme d'appairage entre deux périphériques, permettant à un attaquant de récupérer la clé de chiffrement utilisée grâce à une attaque de type force brute, impactant la confidentialité des échanges. Armis Lab. [Ben 2017] a publié une série de fautes de conception touchant plusieurs systèmes opératoires récents, notamment Android, IOS et Linux (librairie BlueZ [12]), permettant la prise de contrôle à distance des objets les utilisant. Finalement, un framework de sécurité offensive appelé Mirage [13] a été développé au sein du LAAS-CNRS par R. Cayre [Cayre 2019], fournissant les modules de base pour la réalisation de tests de pénétration dans les environnements BLE.

Zigbee Zigbee est une pile protocolaire proposée pour concurrencer le Bluetooth basée sur la spécification 802.15.4, qui propose une topologie maillée permettant à tous les éléments du réseau de communiquer directement entre eux, chaque élément ayant la possibilité d'agir comme un routeur pour échanger des messages entre une source et une destination. Une série d'objets connectés bien connue du grand public est la célèbre marque Philips Hue, proposant des luminaires connectés pour les domiciles fonctionnant en Zigbee, où chaque ampoule communique avec une passerelle Zigbee connectée à Internet. E. Ronen et al. [Ronen 2018] ont démontré qu'il était possible de prendre le contrôle des ampoules en réinitialisant la connexion entre la passerelle et ces dernières. Ils présentent également une preuve de concept de ver pouvant se propager à la suite d'une insertion de code malveillant dans les ampoules, permettant de contaminer et de prendre le contrôle de tout un réseau d'ampoules. T. Zillner [Zillner 2015] présente également un outil appelé SecBee permettant de tester la sécurité des objets utilisant ce protocole. Grâce à cet outil, ils ont pu montrer qu'un grand nombre d'implémentations de cette pile protocolaire ne fournissaient pas les garanties de sécurité nécessaires, notamment lors des échanges de clés pour chiffrer les communications, qui sont pourtant définies dans la spécification. Un dernier framework qui est sans aucun doute le plus abouti pour ce protocole est *KillerBee* [14]. Celui-ci fournit les outils de base pour tester la sécurité de réseaux Zigbee, et il est également particulièrement bien maintenu comparé à d'autres framework similaires proposés.

12. http ://www.bluez.org/
13. `https://redmine.laas.fr/projects/mirage`
14. `https://github.com/riverloopsec/killerbee`

Wi-Fi et protocoles applicatifs de la pile TCP/IP Le protocole Wi-Fi a subi son lot de vulnérabilités, souvent liées aux protocoles d'échanges de clé permettant le chiffrement des communications au sein d'un réseau composé d'objets Wi-Fi. Au niveau des spécifications, la dernière série de vulnérabilités en date est celle de KRACK [Cimpanu 2017], permettant de réaliser une attaque MitM sur des réseaux utilisant le protocole WPA2, aujourd'hui encore largement employé dans les environnements connectés. Une autre attaque très connue exploitant des vulnérabilités de la couche applicative Telnet reposant sur la pile protocolaire TCP/IP est le ver Mirai [Antonakakis 2017], aujourd'hui largement étudié dans la littérature. Ce ver permet de prendre le contrôle d'objets (à l'origine des routeurs, caméra IP, etc.) à distance, dans le but de réaliser une attaque DDOS sur des infrastructures réseaux. Il a notamment impacté le service *DynDNS*[15] le 21 octobre 2016, ce qui a par transitivité affecté des sites comme *Twitter*. Mirai reposait sur l'intrusion via des ports de débogage Telnet qui utilisait des identifiants par défaut pour réaliser l'authentification, puis se répliquait sur les objets à portée. N. Dhanjani présente dans son livre *Abusing the Internet of Things* [Dhanjani 2015] les problématiques de sécurité des objets connectés utilisant le Wi-Fi et des protocoles applicatifs de la pile TCP/IP, tel qu'UPnP. L'une de ses faiblesses est que l'authentification est reportée sur celle réalisée par Wi-Fi. Pour résumer, lorsqu'un objet est authentifié auprès du point d'accès Wi-Fi de l'environnement, celui-ci a toutes les autorisations pour communiquer, commander, et interroger les objets utilisant UPnP. N. Dhanjani le montre sur un babyphone Belkin WeMo disponible dans le commerce.

Autres protocoles sans-fil Dans le domaine domotique et industriel, les objets connectés utilisent parfois des protocoles de communications propriétaires, définis par les fabricants. Ces protocoles ont des fonctionnements distincts, contribuant à la forte hétérogénéité des environnements connectés.

Concernant ces protocoles propriétaires, une série de vulnérabilités a été découverte par M. Newlin [Newlin 2016], touchant les périphériques sans-fil tels que les souris et les claviers, appelée *MouseJack*. Les protocoles utilisés pour faire communiquer ces périphériques avec des ordinateurs sont souvent propriétaires et utilisent la bande 2.4 GHz, un exemple étant l'*Enhanced Shockburst* (ESB) [Nordic 2007], qui a un fonctionnement similaire au BLE. M. Newlin a notamment montré qu'un grand nombre d'objets était sensible à des attaques permettant de violer la confidentialité des échanges, ou d'injecter des messages tels que des frappes clavier ou des déplacements de souris. Toujours dans le registre des protocoles propriétaires, A. Francillon et al. [Francillon 2011] ont utilisé des relais malveillants permettant l'ouverture à longue distance de voitures utilisant la technologie d'ouverture sans clé *Passive Keyless Entry and Start* (PKES), qui communiquent sur les fréquences 315 MHz et 433 MHz. S. Kamkar [Kamkar 2015] présente une série d'attaques par rejeu sur les portes de garages et les véhicules, et fournit un dispositif appelé OpenSesame[16]

15. `https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/`
16. `https://github.com/samyk/opensesame`

qui utilise des séquences De Bruijn [17] pour les ouvrir à distance. Sur le protocole Z-Wave, qui est un protocole propriétaire moins connu de l'IoT, B. Fouladi et al. [Fouladi 2013] présentent un outil appelé Z-Force permettant d'intercepter les communications de ce protocole. Il montre également la présence d'une vulnérabilité sur l'implémentation du standard cryptographique AES pour des serrures connectées, qui permet l'ouverture à distance de celles-ci. Finalement, sur le protocole DVB-T, un protocole satellitaire utilisé par les télévisions connectées fonctionnant entre les fréquences 475 MHz et 850 MHz, Y. Bachy et al. [Bachy 2019] ont découvert des vulnérabilités permettant d'injecter des messages malveillants ou de prendre le contrôle à distance d'une télévision.

Au travers des attaques présentées ci-dessus, nous voyons toute la difficulté à imaginer une solution de sécurité capable de réaliser de la détection d'intrusion dans le contexte d'environnements connectés. En effet, cette solution doit être en mesure de détecter et/ou de protéger les systèmes vis-à-vis de l'ensemble des attaques potentielles touchant des protocoles hétérogènes. Il faut donc que celle-ci soit suffisamment générique pour permettre de monitorer ces communications sans-fil très différentes, afin d'identifier des tentatives d'intrusion pouvant survenir sur ces surfaces d'attaques.

1.3 Solutions traditionnelles de sécurité réseau

Il nous faut à présent identifier les architectures existantes qui soient en mesure de réaliser de la détection/protection vis-a-vis des attaques de l'IoT. Pour cela, cette section présente un état de l'art de ces architectures, en identifiant les limites des approches traditionnellement utilisées dans les systèmes informatiques.

Dans le cas des solutions de sécurité réseau, c'est-à-dire cherchant à protéger ces systèmes des tentatives d'intrusions par le biais des communications filaires ou non-filaire, les trois composants principaux sont : le pare-feu ou *firewall* en anglais, le réseau privé virtuel ou *VPN* en anglais, et les systèmes de détection ou de prévention d'intrusion ou *IDS/IPS* en anglais. La problématique ici est de savoir s'il est possible d'utiliser ces solutions traditionnelles dans des environnements connectés, malgré les caractéristiques spécifiques de l'IoT.

1.3.1 Pare-feu

Le pare-feu, ou *firewall* en anglais, est un composant sécurisé, matériel ou logiciel, permettant d'implanter une politique de sécurité réseau. Celle-ci correspond à un ensemble de règles définissant les communications autorisées ou non. Un pare-feu est utilisé pour cloisonner des espaces dans le réseau, par exemple en séparant le réseau interne contenant des données sensibles du réseau externe, par exemple Internet. Il agit donc comme un filtre, qui va autoriser ou non une communication en fonction de son type, de son contenu (s'il n'est pas chiffré), de son destinataire

17. `https://fr.wikipedia.org/wiki/Suite_de_de_Bruijn`

ou de sa source. L'exemple le plus connu de pare-feu est celui utilisé dans les distributions Linux : iptables, qui permet de filtrer les messages reçus et envoyés sur une machine. Nous identifions 2 types de pare-feu :

— Pare-feu réseau : pare-feu positionné sur le réseau, au centre ou a minima au carrefour des communications de l'environnement qu'il cherche à protéger. L'exemple le plus courant est celui du pare-feu positionné sur le routeur ou sur le point d'accès du domicile, qui voit donc passer l'ensemble des communications locales ainsi que celles entrantes et sortantes.

— Pare-feu personnel : pare-feu personnel positionné sur le système à protéger. Il est donc en mesure de voir tout le trafic produit par le système, ainsi que toutes les communications reçues. Le filtre peut donc rejeter des tentatives d'intrusions ou empêcher la propagation d'une bombe logique à d'autres systèmes du réseau.

Cependant, dans le cas d'environnements connectés, ces types de pare-feu sont très difficiles à mettre en œuvre dans l'état. En effet, l'hétérogénéité des protocoles utilisés et leur forte évolution rendent complexe l'utilisation d'un pare-feu réseau, puisqu'il n'est pas envisageable d'en proposer un qui soit en mesure d'analyser toutes les communications, tout en restant maintenable. En outre, dans le cas des systèmes interconnectés traditionnels, les protocoles, filaires ou non-filaires, fonctionnent souvent dans un mode centralisé (c'est le cas du Wi-Fi par exemple). Ainsi, les communications passent toutes par un point d'interconnexion équipé d'un pare-feu, qui est à la capacité de les filtrer. Dans le cas des tentatives d'intrusion venant de l'extérieur, un pare-feu serait tout à fait en mesure de protéger les objets connectés accessibles depuis Internet. Par exemple, dans le cas de Mirai, de simples règles interdisant les requêtes de connexion Telnet depuis l'extérieur auraient suffi à limiter sa propagation. Or, les objets connectés utilisent souvent des protocoles ad hoc, et donc décentralisés. Un seul pare-feu réseau traditionnel serait donc dans l'incapacité de voir l'ensemble des échanges et ne pourrait pas appliquer ses règles de filtrage. Une solution potentielle serait d'utiliser plusieurs pare-feu réseaux, chacun dédié à un protocole. Cependant, ce type de solution est à la fois coûteux, contraignant et peu maintenable vis-à-vis des évolutions et du dynamise des environnements connectés.

Le second type de pare-feu est tout aussi inapplicable dans le domaine de l'IoT. En effet, une autre caractéristique importante des objets connectés est celle du minimalisme, imposant un certain nombre de contraintes à ces objets. Un pare-feu personnel intégré à chaque objet est donc difficilement concevable, puisqu'il impose des capacités de calcul et d'analyse suffisantes pour monitorer, analyser et filtrer les communications en entrée et en sortie. En outre, cela imposerait également aux constructeurs de concevoir des objets implémentant dès leur commercialisation ce composant, ce qui est aujourd'hui peu réaliste.

1.3.2 Réseau Privé Virtuel (VPN)

Le réseau privé virtuel ou *VPN* en anglais, est un système qui permet d'isoler et de sécuriser les communications entre deux réseaux distants. L'idée est de construire un tunnel entre deux réseaux, dans lequel tout le trafic entrant et sortant de l'un des deux réseaux est échangé par le biais de ce tunnel. En ajoutant de l'authentification forte pour n'accepter des échanges qu'avec des systèmes ou utilisateurs autorisés, les échanges sont sécurisés entre deux systèmes.

Ce mécanisme est notamment utilisé dans l'IoT pour protéger les accès distants aux objets connectés localement au sein d'un environnement connecté. En effet, un utilisateur établissant un VPN depuis un réseau distant vers son réseau local composé d'objets sera considéré comme étant local, et pourra donc contrôler les objets malgré la présence d'un pare-feu ou d'un IDS. Ici aussi, le VPN peut être placé au niveau du réseau, tel que présenté précédemment, ou directement sur un système pour pouvoir s'y connecter de manière sécurisée. Ce deuxième type de VPN est peu applicable aux objets connectés, pour les mêmes raisons que celles énoncées pour les pare-feu personnels. Quant aux VPN réseaux, ils permettent en effet de s'authentifier et de sécuriser les échanges vers un réseau local composé d'objets, et donc est un bon moyen de se protéger des attaques venant de l'extérieur, notamment d'Internet. Cependant, il est inconcevable de mettre en place des VPN pour des objets qui communiquent de manière décentralisée, puisqu'il en faudrait un par objet, voire un par protocole utilisé par l'objet si celui-ci en utilise plusieurs. Dans ce cas, des solutions comme la cryptographie sont plus avantageuses pour établir une communication chiffrée et authentifiée entre deux objets.

1.3.3 Système de Détection/Prévention d'Intrusion (IDS/IPS)

Les derniers composants utilisés dans le cadre de la sécurité réseau sont les systèmes de détection ou de prévention d'intrusion ou *IDS/IPS* en anglais. Leur objectif est de détecter ou de prévenir des intrusions en analysant les échanges entre ou au sein de systèmes. Pour cela, ces systèmes doivent être en mesure de : capturer ces échanges, d'identifier une attaque dans ces captures et de lever une alerte et/ou d'empêcher la tentative d'intrusion après identification. Nous pouvons également identifié deux types d'IDS/IPS principaux :

— IDS/IPS réseaux : qui surveillent les échanges effectués sur le réseau, aussi appelés *NIDS* (ou *NIPS*), qui fonctionnent à l'aide d'une sonde spécifique positionné sur le réseau.

— IDS/IPS hôtes : qui surveillent la sécurité aux niveaux des hôtes, donc directement au sein des systèmes, aussi appelés *HIDS* (ou *HIPS*), qui fonctionnent à l'aide d'une ou plusieurs sondes implantée(s) dans les hôtes.

Le type d'IDS/IPS qui nous intéresse dans notre cas est le NIDS, puisque c'est celui qui va permettre de surveiller les communications entre objets connectés dans un environnement. Les HIDS sont souvent utilisés dans les systèmes pour vérifier que des actions effectuées au sein du système n'impactent pas les propriétés de

sécurité, par exemple en vérifiant les fichiers de logs, les appels systèmes effectués ou en vérifiant l'accès à certaines ressources de ce système. Un exemple d'HIDS a été développé au sein du LAAS-CNRS par A. Damien et al. pour les systèmes avioniques [Damien 2018]. Des HIDS pourraient être implémentés dans les objets connectés pour assurer les propriétés de sécurité en interne. Cependant, les mêmes contraintes que précédemment s'appliquent, et seule une collaboration avec les fabricants peut permettre la mise en place de ce type de composant. Dans le cas du NIDS, les solutions existantes couvrent souvent les protocoles traditionnels tels que le Wi-Fi, ou des protocoles applicatifs. Or, les nombreux protocoles non-filaires de l'IoT ne sont souvent pas couverts, ou seulement partiellement. Une solution potentielle serait d'équiper un NIDS d'un grand nombre de récepteurs adaptés à tous les protocoles de l'IoT. Or, ces protocoles ont tendance à évoluer et de nouveaux apparaissent et disparaissent dans les environnements, nécessitant des reconfigurations régulières de l'IDS pour pouvoir capturer et monitorer tous les échanges, même ceux ad hoc. De plus, les protocoles propriétaires dont les spécifications sont inconnues au préalable, qui sont caractéristiques de l'IoT, doivent également être observables, pour empêcher des attaques semblables à celles présentées par A. Francillon et al. [Francillon 2011] et S. Kamkar [Kamkar 2015].

Pour répondre aux problématiques de l'IoT, un composant spécifique semblable à un NIDS serait donc une solution intéressante, pourvu que celle-ci soit suffisamment générique pour limiter le besoin de reconfiguration et pour pouvoir monitorer tous les protocoles, notamment ceux propriétaires.

Pour l'identification des attaques, deux stratégies différentes sont utilisées :

— IDS/IPS à signatures : une attaque est identifiée comme telle si les éléments de la capture (contenu du paquet, métadonnées) correspondent à la signature d'une attaque connue ou déjà identifiée. Ce mode de fonctionnement est similaire à celui des antivirus, qui utilisent une base de signatures d'attaques pour pouvoir identifier une menace potentielle.

— IDS/IPS comportemental : une attaque est identifiée comme telle si les éléments de la capture correspondent à un comportement anormal. Dans le cas de ces IDS/IPS, un modèle de comportement normal est établi au préalable, correspondant à l'ensemble des communications considérées comme légitimes, donc sans malveillances. La capture d'une communication est donc comparée à ce modèle, et si celle-ci est sensiblement différente au modèle légitime, cette communication est identifiée comme étant illégitime.

Chaque type d'identification possède ses avantages et ses inconvénients. Les IDS/IPS à signatures sont plus précis, puisqu'ils possèdent une connaissance exhaustive des attaques identifiées comme telles. Cependant, une attaque n'ayant jamais été rencontrée ou n'étant pas listée dans la base de signatures (c'est le cas des exploitations de vulnérabilités *zero-day*, correspondant à des vulnérabilités non connues) ne sera pas détectée comme étant une attaque. Au contraire, les IDS/IPS comportementaux sont en mesure de détecter des attaques non connues au préalable, puisqu'ils pourront identifier une communication malveillante de ce

type vis-à-vis de ses différences par rapport au modèle comportemental établi. Ces types d'IDS/IPS sont néanmoins moins précis, puisque si un comportement légitime n'a pas été intégré au modèle comportemental (puisque non capturé lors de sa construction par exemple), celui-ci sera identifié à tord comme une attaque. Dans le cas des objets et des environnements connectés, la diversité des objets et la relative nouveauté des protocoles utilisés rendent l'approche comportementale plus intéressante, puisque le risque de vulnérabilités *zero-day* est plus élevée dans le cas de technologies récentes.

Finalement, ces différentes approches traditionnelles couvrent donc un certain nombre de malveillances pouvant être perpétrées dans les environnements connectés, notamment celles venant de l'extérieur. En effet, la mise en place de solutions de sécurité réseaux se base sur les points d'entrée et de sortie du réseau, comme un pare-feu, peut suffire à détecter et protéger les systèmes au sein du réseau local d'attaques venant d'Internet. Cependant, dans le cas des communications s'effectuant au sein du réseau local, ces solutions sont très incomplètes, et ne permettent pas de protéger ou de détecter efficacement des attaques exploitant les protocoles de communications sans-fil décentralisés. Ainsi, nous pensons que seule une solution générique, ayant la capacité de monitorer tous les protocoles en même temps, même ceux n'ayant pas de spécifications connues, serait en mesure d'assurer un mécanisme de tolérance aux fautes viable dans ces environnements connectés.

1.4 Objectifs de la thèse

Dans ce chapitre, nous avons présenté et discuté du concept d'Internet des Objets, notamment via la présentation des environnements connectés et des objets connectés, ainsi que de leur manière de communiquer par rapport aux systèmes traditionnels. Nous avons ensuite présenté les différentes caractéristiques de ces objets, ainsi que les problèmes de sécurité associés à celles-ci qui nous a amené à nous questionner sur la possibilité d'une solution permettant d'assurer les propriétés de la sécurité-immunité : la confidentialité, l'intégrité et la disponibilité. Les moyens pouvant être mis en oeuvre ont également été présentés : la prévention, la tolérance, l'élimination ou la prévision.

Nous pouvons déduire à partir des problèmes de sécurité qu'il serait impossible d'appliquer de la prévention, de l'élimination ou de la prévision de manière globale à l'IoT sans réaliser un travail préalable de formation aux bonnes pratiques de la sécurité pour les fabricants de ces objets, proposés par des organismes comme le NIST [Ross 2016] ou par l'*US Department of Homeland Security* [U.S. Department of Homeland Security 2016]. Cependant, des mécanismes de tolérance sont possibles, en particulier la détection d'intrusions pouvant survenir dans un environnement connecté. L'une des principales surfaces d'attaques des objets étant leurs systèmes de communications, puisque visibles et accessibles à distance par les attaquants, nous avons choisi de nous focaliser sur les menaces associées.

L'objectif de cette thèse est donc d'étudier la faisabilité, puis de proposer une

solution de sécurité générique, permettant d'assurer la détection et la protection d'un environnement composé d'objets connectés. La solution que nous proposons est un système de détection d'intrusions (IDS) comportemental, basé sur deux éléments principaux :

— La radio logicielle (*Software-Defined-Radio* (SDR)), pour écouter les communications au niveau des ondes radios. Celle-ci nous permet de nous affranchir des spécificités protocolaires et de nous adapter à la forte hétérogénéité des communications.

— L'apprentissage automatique, dont l'objectif est de définir un modèle des communications légitimes de manière automatisée.

Le prochain chapitre présente donc l'état de l'art de la sécurité dans l'IoT et de ces technologies, ainsi que leurs utilisations.

Chapitre 2

État de l'art

Sommaire

2.1 Apprentissage automatique 37
2.1.1 Modèle et apprentissage 38
2.1.2 Différentes familles d'apprentissage 39
2.1.3 Compromis biais variance 40
2.1.4 Requêtes 41
2.2 Solutions de sécurité appliquées à l'IoT et limites 41
2.2.1 État de l'art des solutions de sécurité spécifiques 41
2.2.2 Récapitulatif des limites 43
2.3 La radio logicielle et les solutions existantes 43
2.3.1 Définition et fonctionnement de la *SDR* 44
2.3.2 Détection d'anomalies basée sur les communications radio . . 46
2.4 Conclusion 47

Pour proposer une solution de détection adaptée aux problématiques de sécurité des environnements et des objets connectés, il est nécessaire d'étudier les travaux existants de la littérature ainsi que les technologies récentes pouvant nous aider dans la réalisation de cette solution. Aujourd'hui, une grande majorité d'équipements de sécurité utilise l'apprentissage automatique lors de la définition des modèles de sécurité ou de l'établissement d'une politique de sécurité (règles d'un pare-feu, etc.). Notre solution se basant également sur cette technologie, nous allons tout d'abord en présenter la terminologie et les éléments de base nécessaires à sa compréhension. Ensuite, nous présenterons un état de l'art des solutions spécifiques appliquées à l'IoT, utilisant notamment des éléments d'apprentissage automatique. Les limites de celles-ci nous permettront d'aborder un autre élément essentiel de notre approche, la radio logicielle ou *Software-Defined Radio* (SDR), qui permettra de répondre notamment aux problématiques d'hétérogénéité des protocoles. Finalement, avant de conclure, nous étudierons la présence dans la littérature d'approches utilisant justement la radio logicielle pour réaliser de la détection afin d'identifier l'intérêt de cette technologie.

2.1 Apprentissage automatique

Pour établir un modèle de sécurité du type comportemental, en particulier nécessaire pour les IDS comportementaux, deux approches existent : une première

basée sur des experts et une seconde basée sur l'apprentissage automatique. La première solution consiste à faire générer ce modèle par des experts qui ont une connaissance complète des enjeux, des interactions et des comportements légitimes ou illégitimes à détecter. Cependant, la définition d'un modèle de ce type comporte un grand nombre de limites dans le cas d'un domaine comme celui de l'IoT. Tout d'abord, l'expertise et la connaissance des différents éléments précédents dans le cas d'environnements connectés sont des compétences extrêmement difficiles à trouver, vis-à-vis de la relative jeunesse de l'IoT, et ces compétences sont souvent coûteuses, en temps comme en argent. En outre, l'humain étant faillible, celui-ci n'est pas toujours en mesure d'adapter rapidement ses connaissances pour être en mesure de rendre son modèle robuste aux évolutions de l'environnement dans lequel il est déployé, ce qui est une des caractéristiques principales des environnements connectés. Inversement, dans cette situation, la seconde solution qui repose sur l'apprentissage automatique est souvent à privilégier. Cette méthode consiste à utiliser des algorithmes, qui permettent à partir de données dites d'apprentissage, de paramétrer un modèle générique pour l'adapter à un problème donné. Le principal désavantage de cette deuxième solution repose sur la nécessité d'acquérir des données qui soient représentatives du problème, et qui doivent être suffisamment nombreuses pour obtenir un modèle précis, c'est-à-dire qui généralise correctement. De plus, l'apprentissage, c'est-à-dire le paramétrage automatique du modèle, se base sur des algorithmes à forte complexité, qui sont donc coûteux temporellement. Cependant, contrairement à la première solution, l'utilisation de l'apprentissage automatique pour la définition d'un modèle à partir de données rend celui-ci plus adaptable, potentiellement améliorable via l'acquisition de nouvelles données, tout en étant complètement automatique, donc moins coûteux. Ces raisons expliquent pourquoi aujourd'hui l'apprentissage automatique est très couramment utilisé dans les solutions de sécurité.

Cette section s'attarde donc sur les différents éléments constitutifs de l'apprentissage automatique, en expliquant tout d'abord le concept de modèle et le vocabulaire spécifique associé à celui-ci. Ensuite, nous nous focalisons sur l'apprentissage, en commençant par sa définition, son fonctionnement, et les différentes familles d'apprentissage existantes dans la littérature, puis en détaillant un principe important de ce domaine qui est le compromis biais-variance. Finalement, l'utilisation du modèle appris est expliqué via l'introduction des requêtes.

2.1.1 Modèle et apprentissage

L'objectif de l'apprentissage automatique est de transformer des données en *savoir*, sous la forme d'un modèle, c'est-à-dire une description d'un système utilisant le langage et des concepts mathématiques. Pour cela, différentes "familles" de modèles (appelées *classes de modèles*) peuvent être utilisées. Un algorithme d'apprentissage automatique a donc comme entrées une classe de modèles et des données et a pour sortie un modèle qui est une instanciation de cette classe avec des paramètres, dont ceux appris grâce aux données.

Plus précisément, on peut décomposer les paramètres constituants un modèle en trois catégories :

— les paramètres de classe qui dépendent de la classe de modèle utilisée,

— les paramètres fournis par le développeur, appelés aussi hyperparamètres,

— les paramètres appris automatiquement lors de la phase d'apprentissage, simplement nommé par la suite "paramètres".

De nombreuses classes de modèles existent, adaptées à différents problèmes. Par exemple, des classes permettent de modéliser des comportements probabilistes tandis que d'autres peuvent modéliser des évolutions temporelles.

L'apprentissage automatique vise à apprendre, à partir de données, un modèle qui soit capable de généraliser, c'est-à-dire d'être à même d'agir ou de prédire de nouvelles données. Selon la tâche à apprendre et le format des données, différentes familles d'apprentissage peuvent être distinguées.

2.1.2 Différentes familles d'apprentissage

Nous pouvons identifier quatre différentes familles d'apprentissage, définies pour des objectifs distincts, qui sont tirées du livre *Artificial Intelligence : a modern approach* par S. Russel et P. Norvig [Russell 2016]. Ces familles sont les suivantes :

1. Apprentissage supervisé ;
2. Apprentissage non-supervisé ;
3. Apprentissage semi-supervisé ;
4. Apprentissage par renforcement.

La première famille est celle de l'apprentissage *supervisé*. L'objectif de ce type d'apprentissage est de faire correspondre une entrée à une sortie sur la base d'exemples de couples déjà labellisés, c'est-à-dire fournissant la sortie attendue par rapport à une entrée. Dans ce cas, l'algorithme d'apprentissage doit être en mesure d'apprendre un modèle qui puisse généraliser à des entrées non apprises, en réussissant à leur faire correspondre les sorties adaptées. Deux sous-catégories de problèmes existent au sein de cette famille : la *classification* et la *régression*. La première cherche, comme son nom l'indique, à apprendre un modèle capable de classer les entrées dans des catégories, ou classes. Par exemple, dans les IDS, la classification peut être utilisée pour classer des paquets réseaux en tant qu'anomalie ou non. Des modèles connus sont par exemple les Machines à Vecteurs de Support (ou *SVM*) et les arbres de décision. La seconde quant à elle, s'intéresse à des sorties qui ne sont pas discrètes mais continues. Par exemple, un modèle obtenu via régression doit, à partir des caractéristiques d'une maison, être capable d'en définir le prix.

Par opposition à la famille précédente, l'apprentissage non-supervisé est basé sur des données qui ne sont pas labellisées. L'objectif est de trouver un modèle capable d'établir des relations entre ces données, par exemple en les regroupant selon leurs similarités. Le *clustering* est un problème d'apprentissage non-supervisé,

où l'objectif est d'apprendre un modèle capable de créer des *clusters* ou groupes à partir des données fournies. *k-means* est un exemple d'algorithme d'apprentissage non-supervisé.

La troisième famille est la plus large puisqu'elle comprend l'ensemble des problèmes qui ne sont ni supervisés, ni non-supervisés. Par exemple, un apprentissage réalisé à partir d'une seule classe d'exemples ou à partir de données seulement partiellement labellisées. Dans le cas d'un IDS, les données peuvent n'être que des exemples de données légitimes, puisqu'il peut être complexe de générer des activités illégitimes en fonction des environnements. Ainsi, l'apprentissage doit définir un modèle capable de classer si une nouvelle entrée correspond à une activité légitime ou non. Les modèles de cette famille sont nombreux, et les exemples les plus courants sont *One-Class SVM* ou l'*Auto-encodeur*.

Enfin, l'apprentissage *par renforcement* fonctionne différemment. Au lieu d'avoir un ensemble figé de données, l'algorithme d'apprentissage va agir sur son environnement et récupérer de nouvelles données. L'objectif est alors de "renforcer" (améliorer) un modèle au fur et à mesure, et ce de manière automatique, tout en choisissant les actions à réaliser. Cette méthode est souvent employée dans le cas de la création d'une intelligence artificielle (IA) dans les jeux, puisque l'apprentissage cherche à définir le modèle des actions ou de l'enchaînement d'actions optimal dans la réalisation d'un objectif, par exemple gagner aux échecs. Pour cela, l'apprentissage effectue des actions sur le système, étudie les conséquences de ces actions, et met à jour le modèle en cherchant à s'approcher de la solution optimale, puis recommence. On peut par exemple faire jouer un modèle contre lui-même pour lui apprendre à mieux jouer. L'exemple d'application le plus connu de cette famille est sans aucun doute AlphaGo [Borowiec 2016], qui a appris à jouer au jeu de Go contre lui-même, jusqu'à battre le champion du monde.

2.1.3 Compromis biais variance

Comme présenté dans la sous-section précédente, les classes de modèles sont nombreuses, et lorsque le besoin d'utiliser l'apprentissage automatique est identifié, il faut être en mesure de choisir la classe de modèle adaptée au problème. En effet, une classe de modèles unique ne peut pas être adaptée à tous les problèmes à cause du *compromis biais variance* qui est le compromis à trouver entre la quantité de données et la complexité du système que l'on cherche à modéliser.

Certaines classes de modèles, comme les réseaux de neurones, sont plus complexes puisque l'apprentissage du modèle est soumis à l'apprentissage d'un grand nombre de paramètres. Dans ce cas, une quantité plus importante de données est nécessaire pour déterminer ces paramètres. S'il n'y a pas assez de données, un phénomène dit de *sur-apprentissage* peut apparaître : l'apprentissage va trop généraliser et apprendre un modèle inutilement complexe (statistiquement, on dit que les estimateurs des paramètres ont une grande variance). Au contraire, un modèle simple avec peu de paramètres (par exemple SVM), ne sera pas adapté à tous les problèmes, notamment ceux trop complexes : il y a une erreur de modélisation ir-

réductible qu'on appelle en statistique le biais. Dans ce cas, peu de données sont nécessaires pour apprendre un modèle correct, et le phénomène de sur-apprentissage est moins courant : un modèle de ce type est en effet incapable de modéliser un système complexe.

2.1.4 Requêtes

Les derniers éléments importants de l'apprentissage automatique sont les *requêtes*. Une requête est une question à poser au modèle une fois celui-ci appris. Des exemples de questions peuvent être : *cette communication est-elle légitime ?* ou *quel est l'enchaînement d'actions optimal à réaliser pour accomplir un objectif précis ?*.

Une fois le modèle appris, celui-ci peut être utilisé de plusieurs manières, et le choix du modèle dépend également du type de requêtes à lui soumettre. Généralement, un modèle simple sera associé à des requêtes à faible complexité, et inversement.

2.2 Solutions de sécurité appliquées à l'IoT et limites

Pour répondre au manque de solutions de sécurité pour les environnements connectés, un certain nombre de travaux de la littérature propose des améliorations des architectures existantes ou des solutions complètes dont l'objectif est de détecter et de réduire les malveillances.

2.2.1 État de l'art des solutions de sécurité spécifiques

Y. Sung [Sung 2016] a investigué la possibilité de détecter de potentiels intrus physiques dans un environnement étudiant en utilisant des sondes qui monitorent l'activité radio à l'aide des *Received Signal Strength Indicators* (RSSI) du Bluetooth Low Energy. Ces mesures permettent d'obtenir la puissance en réception d'un signal perçu par une antenne. Elle donne donc des indications sur l'intensité du signal reçu. Ces travaux se limitent à détecter des intrus qui possèdent, sur eux, des équipements BLE et se focalisent donc seulement sur les intrusions physiques. Par conséquent, ils ne sont donc pas en mesure de détecter des attaques qui surviennent sur les communications sans-fil. Cependant, ils montrent des résultats prometteurs sur la détection de comportements anormaux au sein d'un environnement connecté.

M. Miettinen et al. [Miettinen 2017] ont développé une solution, appelée IoT Sentinel, dont l'objectif est de 1) identifier précisément les objets connectés d'un environnement ainsi que leurs caractéristiques en utilisant des techniques de fingerprinting sur les échanges Wi-Fi ; 2) identifier les vulnérabilités de chaque objet en utilisant la base de données de vulnérabilités appelée *Common Vulnerabilities and Exposures* (CVE) ; 3) isoler les objets vulnérables à l'aide d'un pare-feu qui limite leurs interactions possibles (interdiction de se connecter à Internet, autorisation de communiquer avec le serveur du fabricant, etc.). Deux problèmes se posent avec ce type de solution préventive. Premièrement, IoT Sentinel ne se focalise que sur les

communications Wi-Fi effectuées sur le réseau local et les autres communications sans-fil ne sont donc pas prises en compte. Ainsi, si une attaque comme celle de E. Ronen et al [Ronen 2018] est réalisée, IoT Sentinel ne la prend pas en considération. En outre, aucune vulnérabilité zero-day n'est couverte par cette solution, puisque celle-ci se base sur les CVE. Deuxièmement, IoT Sentinel contraint les objets vulnérables en les empêchant de communiquer avec l'extérieur. Or, la majorité des objets a besoin de communiquer avec des serveurs externes. La conception des objets étant souvent fragile, une autre conséquence est le manque de maintenance des fabricants, qui ne mettent pas à jour ni ne corrigent les vulnérabilités. En conséquence, un objet vulnérable de ce type devient définitivement isolé et donc inutile.

S. Raza et al. [Raza 2013] ont développé un NIDS pour les réseaux IPv6 et 6LoWPAN (un protocole d'adaptation situé entre la couche réseau IPv6 et la couche liaison 802.15.4) appelé SVELTE. Leur approche consiste à construire un IDS hybride, à la fois positionné sur les objets et sur le routeur 6LoWPAN du réseau. Pour mettre en place cette solution, il faut donc modifier les objets et établir une connexion sécurisée entre eux pour détecter des attaques sur le routage, telles que les *sinkhole attacks*[1] ou *selective forwarding attacks*[2]. Une problématique importante de leur solution est la modification nécessaire des objets connectés pour assurer la détection. Cette hypothèse dépend en effet grandement des constructeurs de ces objets, qui sont déjà nombreux au sein d'un seul et même environnement. Comme pour les autres solutions présentées jusqu'à présent, leur approche ne se focalise que sur un seul protocole, 6LoWPAN, ce qui limite grandement la capacité de détection dans un environnement connecté actuel.

Y. Meidan et al. [Meidan 2017] propose une approche basée sur de l'apprentissage automatique pour détecter des objets non autorisés à partir d'une liste blanche d'objets établie au préalable. Pour cela, le comportement des objets de la liste blanche est appris à l'aide d'un classifieur à partir de leurs échanges IP, chaque classe correspondant à un objet de la liste. Ensuite, si un nouvel objet apparaît, celui-ci est détecté comme étant non autorisé puisque son flux IP ne correspond pas à ceux de la liste blanche, puisque le classifieur ne trouve aucune classe associée à ce flux. Encore une fois, seules les communications TCP/IP sont traitées par cette solution. Ainsi, les objets communiquant directement à l'aide de protocoles ad hoc ne peuvent pas être identifiés. Une autre limite concerne le non respect de la caractéristique de dynamisme de ces objets, qui ont tendance à apparaître et disparaître des environnements. Il faut donc fournir un moyen aux utilisateurs d'autoriser et de réapprendre le classifieur pour prendre en compte de nouveaux objets.

S. Siby et al. [Siby 2017] développe une solution appelée IoTScanner qui permet le monitoring de protocoles hétérogènes. Chaque protocole est en effet monitoré à l'aide d'un matériel spécifique adapté à la réception de ce protocole (e.g. un périphérique Bluetooth pour monitorer le trafic Bluetooth, etc.). Cependant, bien que ce type de solution est intéressant du point de vue de la caractéristique d'hétérogé-

1. redirection du trafic vers un nœud malveillant contrôlé par un attaquant
2. un attaquant contrôlant un noœud ne transmet qu'un sous-ensemble des paquets

néité, il ne peut pas être aisément maintenu : chaque fois qu'un nouveau protocole est ajouté à ceux de l'environnement, IoTScanner doit être enrichi avec un périphérique dédié associé à ce protocole. De plus, les protocoles propriétaires ou non standardisés ne peuvent pas être pris en compte par cette approche, puisque par définition, ils ne fournissent pas les éléments permettant de monitorer le contenu des messages échangés par ce biais. Il faudrait dans ce cas développer un périphérique ou dongle spécifique après rétro-ingénierie, ce qui est à la fois coûteux en temps et en argent.

2.2.2 Récapitulatif des limites

Les solutions présentées dans cette section souffrent d'une faiblesse commune importante : elles ne se focalisent pour la plupart que sur un seul protocole, et seulement sur ceux pour lesquels les connaissances sont suffisantes pour analyser le contenu des échanges effectués. En outre, ces solutions ne prennent que rarement en compte les protocoles récents de l'IoT, qui font pourtant partie intégrante des environnements connectés, et pour lesquels nous avons montré que de nombreuses vulnérabilités existent et sont exploitées. Pour les solutions du type de IoTScanner qui proposent de monitorer plusieurs protocoles, aucune réponse n'est donnée pour tenir compte des protocoles propriétaires, et leur adaptation à des évolutions protocolaires ou aux changements de l'environnement n'est pas traitée, puisqu'elle requiert des reconfigurations non négligeables.

Dans ce contexte, il est difficile de maintenir sur le long terme ces solutions de sécurité, notamment si celles-ci sont implémentées dans les passerelles, donc limitées dans leur capacité à détecter ou se protéger des attaques survenant sur des protocoles non supportés, typiquement ad hoc. Cependant, une technologie récente venant du monde des télécommunications pourrait potentiellement répondre à ces problématiques, la technologie radio logicielle, souvent nommée *Software-Defined Radio* ou *SDR* dans la littérature.

2.3 La radio logicielle et les solutions existantes

Pour pallier le manque de généricité vis-à-vis du monitoring des protocoles hétérogènes de l'IoT, une technologie intéressante est la radio logicielle ou *Software-Defined Radio* (SDR). Celle-ci permet en effet de gérer logiciellement les éléments constitutifs des chaînes de réception et de transmission utilisées dans les systèmes de communications sans-fil. Nous présentons dans cette section la définition de la SDR, ainsi que son fonctionnement, en mettant en avant les avantages de son utilisation dans un contexte hétérogène comme celui de l'IoT. Ensuite, nous présentons des solutions utilisant ce mécanisme pour réaliser de la détection d'anomalies, dans le but d'analyser son utilisation et son intérêt mais également pour positionner nos travaux.

2.3.1 Définition et fonctionnement de la *SDR*

Selon M. Beach et al [Tuttlebee 2003] la *Software-Defined Radio* ou SDR est une sous-partie d'un modèle multidimensionnel appelée *Software-Based Radio (SBR)* qui cherche à transformer la manière dont les émetteurs et récepteurs traditionnels sont implémentés. L'objectif de la SBR est de proposer des périphériques sans-fil qui puissent être adaptables, reconfigurables et multi-fonctionnels vis-à-vis de leurs modes (émission, réception), des bandes de fréquences radios utilisées, des interfaces et des formes d'ondes émises et reçues. Pour cela, l'idée est d'utiliser des mécanismes logiciels, plutôt que matériels, pour renforcer ces caractéristiques au niveau du périphérique radio. Un modèle en couches est proposé pour la construction d'un périphérique SBR, concernant : 1) l'implémentation de la couche radio, c'est-à-dire la manière dont est implémenté ce type de périphérique en terme de fonctionnalités radios (la SDR), 2) l'implémentation des mécanismes réseaux, englobant le fonctionnement complet du système communicant, 3) l'implémentation d'une couche de service, concernant la mise en place de services permettant de faire évoluer l'infrastructure du système via des mécanismes logiciels et finalement 4) l'implémentation d'une couche utilisateur, autorisant l'utilisateur à accéder à des fonctionnalités et un paramétrage radio différent en fonction de ses applications. Dans le cadre de nos travaux, nous nous intéressons plutôt à la première couche, la radio logicielle ou SDR, c'est-à-dire à l'implémentation d'un périphérique radio de manière logicielle. En effet, l'intérêt est ici de comprendre comment un périphérique de ce type peut être utilisé pour permettre le monitoring de protocoles hétérogènes, ayant des caractéristiques très différentes dans la manière d'exploiter le médium de communication.

Au niveau de la couche physique, un certain nombre de paramètres doivent être définis pour permettre l'échange de données sur un médium de communication. Pour rappel, dans le cas de communications non-filaires, la couche physique définit comment transmettre dans l'air des données binaires à partir des formats de données des couches supérieures (trames, paquets, etc.). Les paramètres nécessaires aux communications sont nombreux, et peuvent être très différents en fonction de la pile protocolaire utilisée (BLE, Zigbee, etc.). Les deux paramètres qui nous intéressent, et qui différencient le plus souvent les couches physiques de ces piles protocolaires sont : 1) la modulation et 2) la fréquence utilisée pour transmettre l'information. La seconde concerne simplement la ou les fréquences sur lesquelles le protocole échange des messages entre un émetteur et un récepteur.

La modulation consiste à modifier la forme du signal à transmettre en une forme qui soit adaptée au canal de transmission, à l'aide d'une onde sinusoïdale appelée *porteuse*. En faisant varier les paramètres d'amplitude, de phase et/ou de fréquence de cette porteuse, la modulation applique une transformation du signal d'origine, générant ainsi le signal modulé. Par exemple, la modulation *Frequency Shift Keying* (FSK) [Masahisa 1968] est une modulation en fréquence, puisque la porteuse permet de modifier la fréquence du signal modulé en fonction des bits transmis, tel que présenté sur la figure 2.1. L'inverse de cette transformation est la

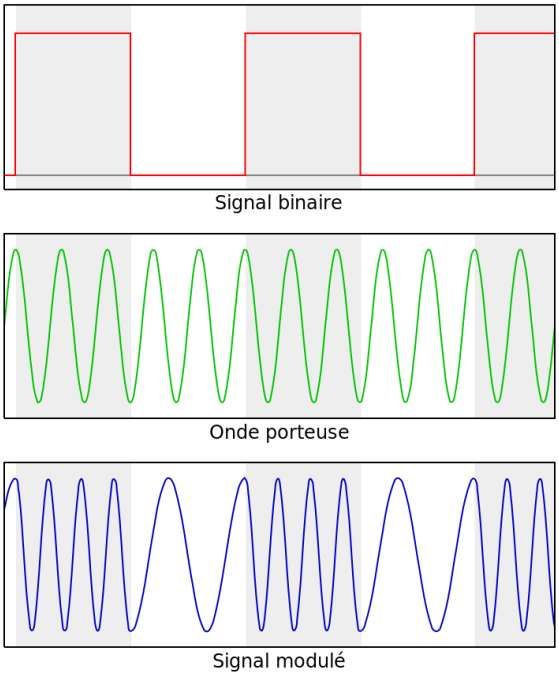

FIGURE 2.1 – Exemple de modulation FSK

démodulation, qui consiste à détransformer un signal reçu qui a été modulé lors de l'émission.

Alors que le périphérique radio traditionnel implémente matériellement ces paramètres, ce qui le rend très statique, la SDR propose de les manipuler logiciellement, améliorant l'adaptabilité des périphériques radios. Pour cela, les signaux perçus par l'antenne d'un périphérique SDR sont transformés en signaux IQ, représentant le signal brut avant toutes les transformations traditionnellement faites par le matériel, notamment la démodulation. Ainsi, à partir de ces signaux IQ, les paramètres du signal peuvent être retrouvés, et la modulation et la démodulation peuvent être adaptées logiciellement pour correspondre à ces paramètres. Bien entendu, cette adaptabilité forte est compensée par une plus forte latence, puisque ces transformations sont réalisées logiciellement plutôt qu'à l'aide de matériels dédiés.

Un certain nombre d'implémentations de périphériques SDR existent aujourd'hui, proposant des caractéristiques différentes vis-à-vis de la couverture en fréquence ou du mode de communication possible - simplex (seulement réception ou transmission), half-duplex (soit l'un soit l'autre) ou full-duplex (les deux). Les cartes électroniques les plus connues proposant de la SDR sont les USRP (*Universal Software Radio Peripheral*) qui ont l'avantage d'être performants mais également coûteux et imposants. Plus récemment, des solutions comme le HackRF [Gadgets 2017] ou le LimeSDR [3] proposent des alternatives à bas coût implémentant des outils lo-

3. `https://limemicro.com/products/boards/limesdr/`

giciels pour les utiliser facilement, comme par exemple des renifleurs (*sniffers*) pour rapidement obtenir les signaux IQ, ou des outils de rejeu, permettant d'enregistrer des signaux IQ et de les rejouer. Le HackRF propose également un système appelé *sweep* qui fournit des informations sur la puissance reçue par l'antenne à des fréquences différentes en balayant de larges bandes de fréquence.

L'utilisation d'un périphérique SDR est potentiellement une solution intéressante pour répondre aux problématiques d'hétérogénéité des protocoles. En effet, il permettrait d'adapter logiciellement un périphérique radio dans le but de monitorer des piles de protocoles fonctionnant avec des couches physiques très différentes. Cela se fait notamment via le *sweep*, qui s'affranchit des spécificités liées à la modulation.

2.3.2 Détection d'anomalies basée sur les communications radio

L'utilisation de la radio logicielle dans la détection d'anomalies est un sujet de recherche relativement récent, et il convient de réaliser un état de l'art des solutions associées proposées dans la littérature.

O'Shea et al. [O'Shea 2016] proposent un détecteur d'anomalies se basant sur un modèle d'apprentissage automatique, appelé réseaux de neurones récurrents, utilisant les communications radios. Grâce à ce modèle, la solution est en mesure de détecter des anomalies radios dans des bandes de fréquences relativement faibles de 20 MHz. La radio logicielle est ici utilisée pour monitorer les communications physiques entre tous les systèmes d'un environnement, en récupérant la puissance reçue par une sonde sur chaque fréquence de la bande de fréquence monitorée. La puissance reçue ainsi que la fréquence sur laquelle celle-ci est perçue permettent ensuite d'apprendre un modèle des communications radios légitimes puis d'identifier des anomalies si de nouvelles communications ne sont pas reconnues par le modèle. Une limitation importante de leur système concerne l'expérimentation, qui n'évalue la solution que sur des anomalies synthétiques, donc pas nécessairement réalistes vis-à-vis d'un environnement connecté. En outre, la faible bande de fréquence monitorée (20 MHz) ne permet pas d'identifier des anomalies associées à des communications utilisant des fréquences éloignées via une seule sonde. Or, une des caractéristiques de l'IoT est la grande hétérogénéité de la couche physique des protocoles employés par les objets connectés, fonctionnant sur des fréquences tels que 868 MHz ou 433 MHz.

S. Rajendran et al. [Rajendran 2018] présentent quant à eux SAIFE, un détecteur d'anomalies interprétable, c'est-à-dire donnant des informations sur les caractéristiques de l'anomalie, utilisant un autre modèle : l'*Adversarial Autoencoder* et des données provenant des communications physiques effectuées dans un environnement. L'objectif de cette solution est de proposer des outils permettant de monitorer les bandes de fréquences surchargées, en se focalisant dans cet article sur les bandes de fréquences de LoRa et Sigfox (800-900 MHz). De la même manière que la solution précédente, les communications radios sont monitorées via des périphériques SDR, puis utilisées pour apprendre le modèle des ondes électromagnétiques perçues traditionnellement. Grâce à des mécanismes de diagnostic, ils sont en mesure de proposer des informations sur une anomalie détectée, comme la fréquence ou la date à

laquelle celle-ci a été identifiée. Une évaluation basée sur des données synthétiques montre le bon fonctionnement de leur approche, avec des résultats satisfaisants. Cependant, l'évaluation réalisée sur des données réelles obtenues via le jeu de données Electrosense [4] ne permet pas de vérifier ses performances de détection, puisque les données ne sont pas labellisées. Finalement, leurs travaux se focalisent exclusivement sur des environnements larges tels que les villes connectées, et donc sur les protocoles longue portée seulement.

2.4 Conclusion

À notre connaissance, aucune approche de la littérature ne propose un détecteur d'anomalies radio générique aux protocoles sans-fil de l'IoT. En outre, les travaux effectués dans cette thèse ont été menés en parallèle des autres travaux présentés dans cet état de l'art. Nous retrouvons ainsi certains points communs dans la manière d'aborder la question de la détection dans les environnements connectés, notamment du point de vue de la SDR et de l'apprentissage automatique. Dans le prochain chapitre, nous présentons donc la première contribution de ce manuscrit, qui consiste en un système de détection d'intrusion générique pour les environnements connectés.

4. `https://electrosense.org/open-api-spec.html`

Deuxième partie

Architecture de sécurité générique

Chapitre 3

Approche générique pour la détection d'anomalies dans les environnements connectés

Sommaire

3.1 Modèle de menaces et hypothèses 52
3.2 Vue d'ensemble de l'approche proposée 53
3.3 Sondes radio 55
3.3.1 Périphérique SDR 56
3.3.2 Contrôleur 57
3.4 Implémentation des sondes radio 58
3.4.1 Périphérique SDR 59
3.4.2 Contrôleur 60
3.5 Apprentissage du modèle des activités légitimes 61
3.5.1 Problématiques des données et choix du modèle 62
3.5.2 Auto-encodeur 64
3.6 IDS 65
3.6.1 Détection 66
3.6.2 Diagnostic 66
3.7 Conclusion 67

Actuellement, les solutions de sécurité permettant de détecter ou de prévenir des malveillances se focalisent exclusivement sur des protocoles précis. Or, les environnements connectés récents se composent d'un ensemble hétérogène de protocoles de communications non interopérables, ne permettant pas à une solution unifiée de couvrir toutes les menaces. Il serait envisageable d'utiliser un ensemble de solutions adaptées à chacun de ces protocoles, mais cela ne permettrait pas de pallier les évolutions ou l'apparition de nouveaux protocoles sans nécessiter des modifications importantes, sans compter le coût important d'une telle architecture.

Ces solutions font également fi de potentielles avancées dans la standardisation de protocoles nouveaux, consacrées à résoudre les problèmes inhérents aux objets qui composeront les environnements de demain. En outre, certains de ces objets utilisent aujourd'hui des protocoles propriétaires ou non standards qui rendent difficiles l'écoute et l'analyse de leurs communications. Un élément important des architectures de sécurité est la capacité à fournir une protection qui puisse être *robuste*

vis-à-vis des potentielles évolutions pouvant survenir. Pour répondre à ce besoin, ce chapitre décrit une architecture de détection générique pouvant être mise en place dans le cadre d'environnements connectés, permettant la remontée d'anomalies. Cette architecture se base principalement sur l'utilisation de la radio logicielle (SDR) en se focalisant sur la couche la plus basse des communications : la couche physique. Ainsi, seules les ondes électromagnétiques reçues sont analysées pour détecter une potentielle anomalie.

Ce chapitre est donc découpé en 7 sections présentant tout d'abord le modèle de menaces et les hypothèses liées à l'environnement. Le principe de cette architecture et son fonctionnement général sont ensuite introduits dans une deuxième section, avant de détailler les différents éléments qui la composent. Dans une troisième et quatrième section, les *sondes radios* ainsi que leur implémentation sont expliquées, suivi par l'apprentissage et son utilisation dans l'approche dans une cinquième section. Finalement, le dernier composant, l'*IDS*, est ensuite présenté dans une sixième section sous ses deux formes : la détection et le diagnostic, avant de conclure ce chapitre.

3.1 Modèle de menaces et hypothèses

Dans un premier temps, il est nécessaire de définir les capacités d'un attaquant ciblant un environnement connecté, ainsi que ses objectifs. Dans la suite de ce chapitre, nous considérons un attaquant comme toute personne non authentifiée susceptible d'agir sur les différents objets en interaction dans l'environnement. Plus précisément, il doit être en mesure de communiquer avec ces derniers, soit par l'intermédiaire d'un objet déjà présent qui a été préalablement corrompu, soit via un objet qu'il a lui même introduit à portée de ses cibles. Les attaques étant perpétrées depuis le réseau Internet ne sont pas considérées, ce qui est une hypothèse raisonnable sachant que des méthodes de protection réseau éprouvées peuvent aisément être mises en place sur ce canal de communication. Dans nos travaux, nous considérons comme principale surface d'attaque de ces environnements les communications sans-fil, sans exclure les protocoles non-standardisés ou propriétaires qui sont insuffisamment pris en compte dans les solutions actuelles. Un attaquant soumis à ces hypothèses peut chercher à réaliser les objectifs suivants :

— Collecter des informations confidentielles [Newlin 2016] ;
— Contrôler les objets connectés pour modifier leur fonctionnement nominal [Ronen 2016] ;
— Contrôler les objets connectés pour les utiliser comme rebond [Ronen 2018] ;
— Modifier les objets connectés pour préparer une intrusion physique [Ho 2016].

Par la suite, nous considérons que pour réaliser ces précédents objectifs, un attaquant doit interagir avec les objets connectés, ou générer des communications qui perturbent celles déjà en présence et ce potentiellement de manière concurrente avec des communications légitimes. Nous supposons également que ces interactions

vont générer des activités radios mesurables et anormales qui peuvent être détectées par un système de monitoring sur le champ radio. Celles-ci peuvent être considérées comme suspectes selon plusieurs situations :

— la puissance perçue est trop importante ou trop faible ;

— la communication est effectuée sur des fréquences habituellement non utilisées ;

— l'activité a lieu à un moment inhabituel ;

— le comportement des communications, i.e. l'enchaînement des activités radio, est non usuel.

La prochaine section détaille une architecture de sécurité générique permettant de surveiller les communications radio pour détecter ce type d'activités suspectes au sein d'un environnement connecté.

3.2 Vue d'ensemble de l'approche proposée

Pour fournir un moyen de détecter les différentes activités suspectes détaillées dans la section 3.1, une solution serait de développer, par l'intermédiaire de matériel dédié, une sonde spécifique à chaque protocole en présence au sein de l'environnement considéré, dans le but de monitorer le trafic correspondant. Cependant, sur le long terme, cette solution n'est pas envisageable car elle nécessite la connaissance préalable des spécifications de tous les protocoles utilisés dans l'environnement, ce qui n'est souvent pas possible et extrêmement coûteux à mettre en œuvre. De plus, les environnements IoT étant très dynamiques, les protocoles utilisés ont tendance à évoluer et à changer au cours du temps, notamment via l'introduction de nouveaux objets connectés. L'architecture proposée doit donc respecter un ensemble de caractéristiques pour pouvoir s'adapter à ces types d'environnement :

1. **Maintenabilité** : pouvoir s'adapter facilement aux potentielles évolutions de l'environnement ;
2. **Indépendance des protocoles** : pouvoir prendre en compte n'importe quel protocole ou technologie, même sans connaissance de sa spécification ou du standard utilisé ;
3. **Non-invasive** : pouvoir s'intégrer à l'environnement sans modification des éléments qui le composent ni perturbation des communications existantes ;
4. **Portabilité** : pouvoir s'intégrer rapidement à n'importe quel environnement connecté.

Pour répondre à ces caractéristiques, nous avons donc imaginé une architecture de sécurité de type IDS, c'est-à-dire capable d'identifier et de détecter des intrusions dans un environnement connecté. Comme défini dans la section 1.3.3, les deux types d'IDS souvent utilisés sont les HIDS, et les NIDS. Les premiers s'installent directement sur le composant, un *Hôte*, pour en surveiller les activités, tandis que les

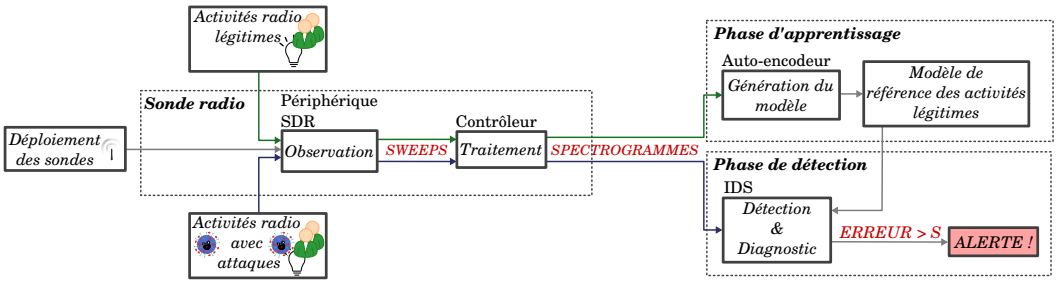

FIGURE 3.1 – Récapitulatif des étapes de l'approche

seconds, les NIDS, vont s'installer sur un composant spécifique pour surveiller les échanges effectués sur les canaux de communications entre différents éléments. Ces derniers utilisent généralement les informations contenues dans les paquets échangés pour détecter des intrusions. Dans le cas de notre solution, nous identifions une nouvelle catégorie d'IDS intitulé *RIDS* pour *Radio Intrusion Detection System*, qui n'utilise quant à lui que des observations de l'espace radio pour détecter des intrusions. Notre solution proposée a pour objectif de monitorer les activités radios sur des larges spectres de fréquences pour couvrir les communications sans fil usuelles. Ces bandes de fréquences doivent être définies au préalable pour couvrir l'ensemble des communications d'un environnement, et la caractéristique 2 impose à l'approche de ne pas connaître les spécificités des protocoles. Pour cela, elle ne doit donc se focaliser que sur les signaux physiques, en s'affranchissant de la capacité à démoduler ces derniers. Ainsi, le contenu des communications ne peut pas être retrouvé, et seules la puissance, la durée et la fréquence de ces dernières peuvent être utilisées comme information pour déterminer leur légitimité. Selon le modèle de menaces préalablement établi, ces informations sont supposées suffisantes pour détecter une anomalie au sein d'un environnement composé d'objets connectés.

Cependant, pour distinguer les activités radios légitimes de celles qui sont malveillantes, l'approche doit être en mesure d'établir un modèle des communications légitimement effectuées par les utilisateurs dans l'environnement. Ainsi, deux phases sont nécessaires pour permettre à notre solution de détecter des malveillances. La première doit recueillir les activités radios générées par l'utilisation des objets connectés en absence d'activités malveillantes et établir un modèle de référence de celles-ci. La seconde phase quant à elle doit détecter les comportements illégitimes ou les anomalies à partir de ce modèle. Pour résumer, cette dernière va tout d'abord observer et capturer en temps réel les communications de l'environnement, puis comparer celles-ci avec le modèle de référence, et enfin générer et lever des alertes lorsqu'une différence trop importante est observée. Il est nécessaire d'établir le modèle de référence préalablement à cette seconde phase.

En conclusion, comme présenté dans la figure 3.1, notre solution repose sur deux composants pour mettre en place ces trois étapes, ainsi que d'un composant hors-ligne non présent sur la figure permettant de générer le modèle de référence :

— des *Sondes Radio* qui observent et capturent les signaux physiques dans l'es-

pace radio ;

— le générateur de modèle des activités légitimes ;

— l'*IDS* qui traite ces observations et implémente la détection.

Les trois sections suivantes s'attardent sur les particularités de ces trois composants.

3.3 Sondes radio

Les *Sondes Radio* sont installées dans l'environnement qui doit être couvert par la solution sans modifier la composition de celui-ci. Les caractéristiques 1 et 2 (maintenabilité et indépendance des protocoles) imposent à l'approche de s'affranchir des considérations techniques hétérogènes présentes dans les protocoles de l'IoT. C'est pourquoi une *Sonde Radio* ne se focalise que sur les signaux physiques, sans chercher à démoduler ces derniers. Ainsi, le standard d'un protocole n'a pas besoin d'être connu pour pouvoir identifier des communications, et l'approche reste très adaptative. En effet, même si des objets utilisant de nouveaux protocoles sont rajoutés dans l'environnement, une sonde n'a besoin dans le pire des cas que d'une modification mineure si ceux-ci communiquent sur des bandes de fréquences non monitorées. Finalement, les caractéristiques 3 et 4 reposent principalement sur la manière dont les activités radios sont monitorées. C'est pourquoi une sonde radio ne doit ni perturber les communications sans fil des objets en présence, ni modifier la manière de communiquer de ces derniers. Ainsi, la solution peut aisément s'installer dans n'importe quel environnement sans en modifier ses usages. Les sondes vont principalement procéder à l'acquisition des données nécessaires au fonctionnement global de l'approche. Elles doivent être adaptées aux hypothèses faites sur l'environnement considéré et doit donc remplir un certain nombre de conditions :

— Capacité à monitorer les activités radios sur plusieurs bandes de fréquences ;

— peu coûteuses ;

— facilement reconfigurables par un expert de sécurité en fonction de l'environnement ;

— portables ;

— capacité de communiquer de manière sécurisée sans perturber l'environnement radio ;

— ne nécessiter aucune modification des objets connectés.

Comme illustré par la figure 3.2, deux tâches distinctes peuvent être identifiées : la phase d'*observation*, qui consiste à capturer les activités radio sur les bandes de fréquences considérées sous forme de *sweep*, et la phase de *traitement* qui consiste à mettre en forme les données obtenues en *spectrogrammes* pour l'IDS. Préalablement à ces deux phases, un étape de déploiement est réalisée par un expert en fonction de l'environnement. Celle-ci consiste à positionner les sondes aux endroits adaptés pour permettre d'observer au mieux les communications en présence.

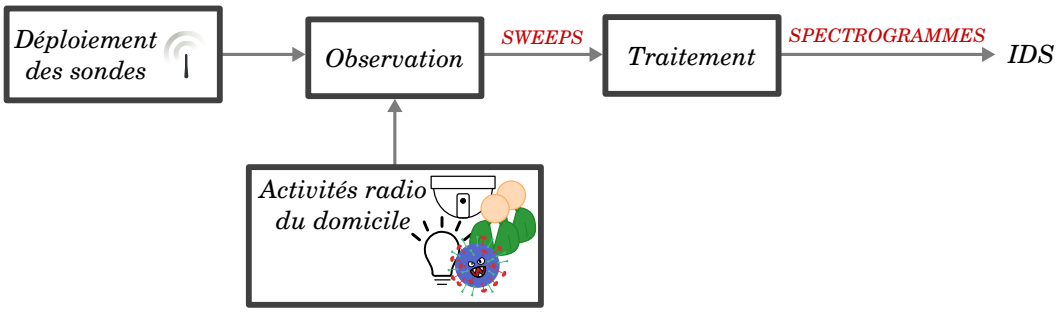

FIGURE 3.2 – Monitoring des activités radios par la sonde

Pour réaliser ces deux phases, une sonde est composée de deux éléments principaux lui permettant de répondre à l'ensemble des conditions précédentes : un périphérique de radio logicielle (SDR) et un contrôleur. Nous définissons p l'identifiant d'une sonde radio. Dans la suite de ce document, toutes les notations et paramètres utilisés sont récapitulés dans l'annexe A.

3.3.1 Périphérique SDR

Dans notre cas, le périphérique SDR nous permet de monitorer en parallèle et en temps-réel de larges bandes de fréquence, contrairement à des composants matériels spécifiques à une technologie communiquant sur une fréquence ou une bande de fréquences précise. Comme précisé dans la section 2.3 de ce document, un périphérique SDR a également la possibilité d'utiliser des mécanismes logiciels lui permettant de démoduler et de retrouver les informations contenues dans les communications (paquets, commandes, etc.) via les signaux IQ. Cependant, ces mécanismes reposent principalement sur la connaissance au préalable des paramètres liés à la technologie utilisée, comme par exemple le type de modulation. Cependant, pour être en mesure de respecter les conditions précédentes, nos sondes ne doivent faire aucune hypothèse sur la couche physique des communications, et donc sur ses paramètres. Ainsi, nous avons choisi de ne pas utiliser ces mécanismes logiciels et de directement traiter les sorties du périphérique SDR.

Pour réaliser l'étape d'observation, nous utilisons un mécanisme particulier pouvant être mis en place matériellement et logiciellement sur un grand nombre de périphériques SDR. Ce mécanisme, appelé *balayage* ou *sweep*, permet à l'aide d'un seul périphérique de monitorer des bandes de fréquences de plusieurs GHz en quelques secondes. Dans ce cas, les données reçues ne sont plus des signaux IQ, mais seulement la puissance reçue sur chaque fréquence de la bande sur laquelle le périphérique effectue son *sweep*. La puissance spectrale de chaque fréquence est calculée via plusieurs Transformées de Fourier FFT (Fast-Fourier-Transform) appliquée sur toute la bande de fréquence (d'où le terme *sweep*). Pour réaliser ces *sweep*, le périphérique change sa fréquence centrale d'écoute autant de fois que nécessaire pour fournir la puissance spectrale sur l'ensemble de la bande en plusieurs points, en fonction de la résolution fréquentielle recherchée. Ce mécanisme est donc relativement lent, ce qui

impacte la résolution temporelle globale lors du monitoring. Cependant, cela permet de mesurer la puissance reçue sur chaque fréquence dans des bandes très larges, jusqu'à 6 GHz sur le HackRF One [1] par exemple. Un autre avantage qu'apporte les *sweep* à la solution repose sur la possibilité d'observer des bandes de fréquences disjointes qui peuvent être facilement reconfigurées. Par exemple, il est possible d'obtenir la puissance sur trois bandes distinctes de 100 MHz : 400-500 MHz, 800-900 MHz et 2.4-2.5 GHz. Compte tenu de nos hypothèses, ce mécanisme semble être suffisant pour identifier des anomalies dans les communications radios effectuées dans un environnement. En outre, bien qu'un ensemble de périphériques SDR très performants soit coûteux (USRP par exemple), cette technologie se démocratise rapidement et un certain nombre de périphériques peu coûteux sont disponibles dans le commerce, comme le HackRF One qui implémente par ailleurs directement le mécanisme de *sweep*.

Dans la suite, les fréquences considérées correspondront à un ensemble F de M intervalles de fréquences avec $F_i = (f_i^s, f_i^e)$, entre f_i^s et f_i^e kHz, ($M = |F|$). Sachant que la mesure de puissance associée à ces fréquences implique l'utilisation de FFT, notons w le nombre de points ou *nombre d'échantillons* de chaque FFT fixé à $w = 8192$. Pour un horodatage donné, la mesure de puissance associée à F_i est appelée un *sweep* et le vecteur ainsi obtenu est noté S_i. Le nombre de valeurs pour chaque FFT calculée par le périphérique SDR est constant, mais peut être filtré en fonction d'un paramètre noté bw appelé *bin width* (exprimé en MHz). L'inverse de ce paramètre correspond à la résolution fréquentielle recherchée, soit $1/bw$ le nombre de valeurs obtenues par MHz. Le vecteur S_i contient donc $K_i = \frac{f_i^e - f_i^s}{bw}$ valeurs pour un sweep. Le nombre de valeurs par seconde est noté V et dépend principalement de bw et de w ainsi que du type de périphérique SDR utilisé, notamment vis-à-vis de la vitesse à laquelle celui-ci calcule les différentes FFT lors d'un sweep. V ne dépend pas de la largeur de bande F_i considérée. Notons également T le temps en secondes entre deux *sweeps*, avec $1/T$ la résolution temporelle du balayage effectué, défini par $T = (\sum_i K_i)/V$, qui ne dépend pas de bw, mais de la somme des largeurs des bandes de fréquence considérées. Pour toutes les bandes $\{b\}$ considérées, les *sweeps* sont obtenus en temps réel et ce de manière continue, et sont par la suite traités par le contrôleur.

3.3.2 Contrôleur

Le second composant d'une sonde radio consiste en un contrôleur qui doit être capable de 1) traiter les *sweeps* S_i obtenus pour les mettre sous un format facile à analyser, à stocker et à partager avec l'IDS et 2) communiquer ces données traitées à l'IDS. Pour simplifier, le contrôleur va agréger les *sweeps* S_i obtenus en sortie du périphérique SDR en plusieurs *spectrogrammes* ou *waterfalls* qui seront ensuite envoyés à l'IDS. Un exemple de spectrogramme ainsi obtenu est représenté par la figure 3.3. La phase de communication entre les contrôleurs et l'IDS est réalisée via un réseau filaire dédié et sécurisé qui est établi au préalable. Celui-ci assure une

1. https ://greatscottgadgets.com/hackrf

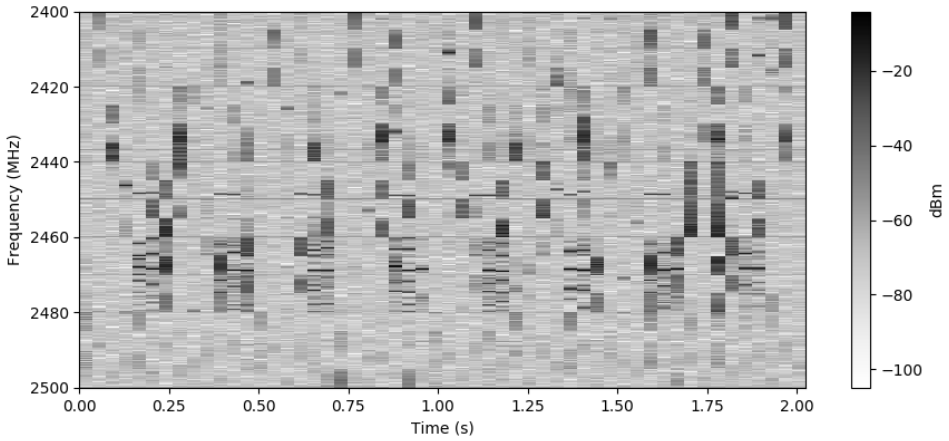

FIGURE 3.3 – Exemple de spectrogramme

mise en place rapide de l'approche dans n'importe quel environnement et garantit la non-perturbation des communications sans fil déjà en présence dans celui-ci.

Sachant que les sweeps sont générés en continu par le périphérique SDR, les N sweeps consécutifs sont agrégés en un *spectrogramme* ou *waterfall* noté $S_{p,b}$ (avec p l'indice de la sonde et b la bande observée). Le j-ème spectrogramme est noté $S^j_{p,b}$. Par la suite, l'indice j sera omis. Chaque spectrogramme est ensuite horodaté à une date t correspondant à la date du dernier sweep : (t, S). Ainsi, un spectrogramme est une matrice de taille $\sum_i K_i \times N$. La figure 3.3 présente un exemple de spectrogramme obtenu durant 2 secondes (avec T =0.0375 secondes) mesuré par un HackRF One configuré sur la fréquence 2.4-2.5 GHz, correspondant à environ 54 sweeps. L'ordonnée k correspond au k-ème bin (une fréquence en MHz) et l'abscisse l correspond au l-ème *sweep*, correspondant à la date en secondes auquel ce balayage a été observé. En résumé, le pixel à la position (l, k) correspond à la puissance $*S_i[l, k]$, observé au temps $t + l * T$, associé à la fréquence $f^s_i + k * bw$ KHz. L'échelle à droite fait correspondre la couleur des pixels à la puissance observée : noir correspond à une puissance élevée tandis que blanc correspond à une puissance faible.

3.4 Implémentation des sondes radio

Pour implémenter une sonde radio utilisée dans l'approche, il est nécessaire de fournir des éléments matériels qui soient en mesure d'implémenter une écoute continue d'un large spectre de fréquences, tout en traitant en temps réel ces observations dans le but d'une analyse fine par l'IDS. Ces sondes doivent donc : 1) être alimentées de façon continue, 2) rester actives sans interruption de service et 3) fournir les éléments logiciels et matériels pour traiter puis communiquer les données avec les autres composants. Le premier élément qui les compose est donc le périphérique SDR, relié au contrôleur.

3.4.1 Périphérique SDR

Pour implémenter l'approche, il est nécessaire d'avoir un périphérique SDR qui réponde aux différentes problématiques énoncées précédemment. Les composants qui définissent le fonctionnement de ce périphérique sont l'antenne utilisée et la méthode de communication entre ce périphérique et le contrôleur.

Concernant l'antenne utilisée, sachant que l'approche doit permettre d'observer des bandes de fréquences larges de plusieurs gigahertz, il est nécessaire d'équiper la sonde avec une antenne qui soit capable de percevoir des signaux sur des fréquences très différentes. Un certain nombre d'antennes, notamment pour la technologie SDR, existent dans le commerce et permettent d'obtenir des résultats satisfaisants lors de la démodulation de certains protocoles. Dans notre cas, sachant qu'il n'est pas indispensable d'obtenir précisément le contenu des communications, mais seulement d'observer la puissance reçue par la sonde à une position donnée, une antenne spécialisée et coûteuse n'est pas forcément essentielle. Cependant, le compromis repose sur la qualité de la réception : une antenne adaptée à une fréquence particulière recevra les signaux sur cette fréquence avec un rapport signal sur bruit (SNR) plus élevé (c'est-à-dire que la puissance du signal est plus élevée que le bruit). Dans le cas d'une implémentation dans un environnement connecté, c'est une question primordiale à se poser vis-à-vis des objets en présence. Par exemple, si ceux-ci utilisent principalement des protocoles fonctionnant dans la bande de fréquence 2.4-2.5 GHz, une antenne adaptée à cette bande peut être suffisante.

Quant au périphérique SDR à utiliser, il doit principalement permettre d'observer des bandes de fréquences très larges, et ce en un minimum de reconfiguration et pour un coût relativement faible. Les périphériques les plus connus sont les USRP, mais ils sont relativement coûteux, ce qui limite leur utilisation dans certains contextes. Cependant, un certain nombre de périphériques SDR grand public apparaissent dans le commerce depuis quelques années, et certains implémentent des fonctionnalités qui sont particulièrement intéressantes dans notre approche. Ces derniers sont souvent beaucoup moins chers que des USRPs, mais permettent de réaliser facilement de la démodulation de communications ou du rejeu. Un des périphériques les plus connus de cette gamme est le HackRF One, et c'est celui que nous avons choisi d'utiliser pour l'implémentation de la solution. En effet, l'aspect démodulation n'étant pas intéressant dans notre approche, une qualité de réception moyenne avec une bande de fréquence observable de 20 MHz est suffisante. En outre, il implémente un outil logiciel appelé *hackrf_sweep* permettant de balayer les fréquences de 1 MHz à 6 GHz en quelques secondes, ce qui est particulièrement intéressant du point de vue de notre approche. Bien entendu, ce type d'outil logiciel est facilement implémentable sur n'importe quel périphérique SDR en modifiant la fréquence centrale pour obtenir la puissance reçue sur des bandes de fréquences très larges. La section 2.3 récapitule les différences entre ces différents périphériques et les caractéristiques de chacun. Le HackRF One impose cependant une communication de type USB avec le contrôleur, qui est moins rapide qu'une communication Ethernet principalement utilisée par les USRP. Encore une fois, un compromis est

à trouver entre précision, vitesse et coût de la solution. Ce périphérique SDR peut être trouvé dans le commerce pour environ 300 €.

3.4.2 Contrôleur

Le contrôleur doit dans un premier temps récupérer les sorties du périphérique SDR et les traiter pour générer les spectrogrammes, puis les communiquer à l'IDS. L'implémentation de ce composant doit donc correspondre à ces deux besoins précédents. Il est donc nécessaire d'utiliser un composant matériel et logiciel qui permette : 1) d'avoir une unité de calcul permettant de traiter en temps réel les observations provenant du périphérique SDR avec une latence faible et 2) fournir les entrées/sorties nécessaires pour recueillir les informations venant du périphérique SDR et émettre les spectrogrammes générés à l'IDS. Ces deux besoins sont toujours associés avec la contrainte de coût liée à l'environnement dans lequel la solution intervient.

Concernant l'unité de calcul nécessaire, les périphériques SDR grand public et notamment le HackRF One choisi dans notre implémentation sont capables de balayer 8 GHz de bande en une seconde. Sachant cela et pour un réglage par défaut de ce périphérique, la résolution temporelle atteinte $1/T$ est d'un sweep par seconde pour parcourir ces 8 GHz. Le nombre de valeurs par seconde V dépend de la résolution fréquentielle choisie $1/bw$ et du nombre de points des FFT calculées w. Par exemple, en fixant $w = 8192$, nous obtenons $V = \frac{8000}{b}w$ valeurs par seconde, avec le temps entre deux *sweep* $T = 1s$. Finalement, pour un balayage allant de $f_s = 0\ MHz$ à $f_e = 8\ GHz$ et $bw = 0.04\ MHz$ (qui est la plus petite valeur possible dans le cas du HackRF One), nous avons donc $V = K = \frac{8000-0}{0.04} = 200000$ valeurs de puissance par seconde et par sweep. Sachant que chaque valeur est encodée en *float64* soit 8 octets, on obtient un débit théorique maximum de 1600000 octets/s, soit 1.6 Mo/s. Ce débit étant largement inférieur à la puissance de calcul de n'importe quelle architecture récente, il est donc acceptable de dire que le contrôleur n'a pas besoin d'une architecture spécialisée et très performante pour être capable de gérer et de traiter ce débit. La majorité du traitement consiste à concaténer à la volée les mesures de puissance obtenues en sortie du périphérique SDR. La complexité de ce traitement étant linéaire, nous considérons que traiter ces données est réalisable en un temps négligeable par n'importe quel processeur 32 bits récent cadencé à plus de 500 MHz.

Concernant les entrées/sorties nécessaires, elles dépendent principalement du type de périphérique SDR utilisé et de la manière dont il communique avec son contrôleur ainsi que du type d'architecture qui implémente l'IDS. Dans le cas du HackRF, la communication se fait via USB, il nous faut donc au moins une entrée USB pour pouvoir récupérer les observations. Le second point concerne l'interface de communication entre le contrôleur et l'IDS. Dans notre cas, nous avons choisi de réaliser cette communication par un lien Ethernet, ce qui permet d'atteindre un débit encore une fois largement suffisant vis-à-vis des données, tout en limitant les problèmes matériels pouvant survenir sur ce lien lors d'un usage en continu. Finale-

ment, nous nous sommes dirigés dans notre implémentation vers un nano-ordinateur Raspberry Pi 3 B+ qui possède toutes les caractéristiques précédentes pour un prix aux alentours de 30 €. En outre, celui-ci permet également d'implémenter rapidement des programmes de traitement en Python3 permettant d'accélérer la mise en oeuvre dans le cadre d'une expérimentation. Le coût estimé pour l'implémentation d'une sonde prototype est donc d'environ 430 €, ce qui est raisonnable dans le cadre d'un domicile connecté, sachant que ce coût peut aisément être diminué en cas d'industrialisation de la solution.

Un point important vis-à-vis des contrôleurs utilisés sur un ensemble de sondes couvrant l'environnement est la synchronisation entre celles-ci. En effet, pour être en mesure d'apprendre les comportements radios et de détecter les anomalies, il est nécessaire que toutes les sondes soient synchronisées temporellement. Ainsi, le modèle de référence se base sur la vision de l'espace radio vue par les différentes sondes à un temps t qui est le même pour toutes les sondes. Pour cela, le HackRF One permet l'utilisation d'une horloge externe qui peut permettre de les synchroniser. Dans notre implémentation, nous avons choisi de nous baser sur le protocole *Network-Time-Protocol* (NTP) qui permet au sein d'un réseau de synchroniser les horloges des différents contrôleurs sur une heure de référence. Les sondes communiquant à l'IDS via un réseau dédié, il est facile d'implémenter et de déployer ce type de solution.

3.5 Apprentissage du modèle des activités légitimes

Vis-à-vis des anomalies pouvant être détectées et connaissant les hypothèses établies précédemment, l'IDS doit être en mesure de définir le comportement légitime des communications observées dans un premier temps, puis analyser les déviations dans les futures communications pour établir une détection. L'IoT étant un domaine vaste où la majorité des attaques sont récentes, il est difficile d'établir un modèle pérenne basé sur des signatures d'attaques. Une méthode potentielle aurait été de générer des attaques au sein de l'environnement pour établir cette base de signatures. Cependant, pour les raisons évoquées précédemment, il est très difficile d'être exhaustif dans cette génération. De plus, en fonction du contexte, les utilisateurs de ce type de solution ne sont pas forcément experts dans le domaine, et il est donc très difficile de les faire mettre en oeuvre des attaques qui puissent être réalistes et exhaustives. Pour toutes ces raisons, notre choix s'est porté sur le développement d'un IDS comportemental, sans s'intéresser à la détection à partir d'une base de signatures connues, mais sur la base d'un modèle des comportements légitimes.

L'aspect comportemental impose donc la mise en place d'un modèle décrivant l'ensemble des comportements légitimes pouvant être observés dans un contexte spécifique. La phase de construction de ce modèle peut être réalisée de deux manières : soit à l'aide d'un expert ayant connaissance ou analysant les comportements des individus et des objets, soit de manière automatique. Notre solution se basant exclusivement sur l'observation des échanges radios, la complexité intrinsèque de

ce type de communications et le peu d'informations récupérables rendent difficile la première alternative. De plus, un grand nombre de travaux récents utilisent des techniques d'apprentissage automatique (*Machine Learning*) qui permettent d'établir automatiquement, à partir de données de références, des modèles adaptés à différentes problématiques. Nous avons donc choisi de nous focaliser sur une solution utilisant l'apprentissage automatique qui donne de très bons résultats dans la littérature. La construction de ce modèle des activités radios légitimes doit par la suite permettre de détecter des anomalies via l'identification de déviations dans le comportement des communications. Cette phase de construction est définie comme étant la *phase d'apprentissage* dans la suite de ce document.

Dans la suite de cette section, nous abordons principalement les problématiques liées aux données recueillies, qui imposent certaines restrictions sur le modèle utilisé. Nous commençons donc par identifier ces problématiques, nous permettant de sélectionner un modèle adapté. Nous expliquons ensuite le fonctionnement de celui-ci, avant d'aborder la manière dont sont traitées les données pour répondre à nos besoins.

3.5.1 Problématiques des données et choix du modèle

Deux principales caractéristiques concernant les données collectées sont capitales et définissent le type de technique qui doit être implémenté dans notre phase d'apprentissage. Premièrement, les sondes vont continuellement collecter les activités radios durant la phase d'apprentissage, générant ainsi un nombre conséquent de spectrogrammes qui devront être traités pour réduire la taille des données collectées et faciliter leur apprentissage. Ces derniers sont collectés pendant un temps suffisamment long pour couvrir l'ensemble des comportements pouvant apparaître dans les communications des utilisateurs et des objets. Le positionnement des sondes a été fait préalablement à cette phase et doit rester le même entre l'apprentissage et la détection, notamment pour conserver une cohérence entre les observations.

Deuxièmement, lors de la phase d'apprentissage, tous les spectrogrammes sont étiquetés comme appartenant à une seule classe de comportement : légitime. Le problème est donc de détecter une anomalie dans l'observation courante en comparaison avec un modèle de référence. Dans l'apprentissage automatique, ce type de problème est défini comme étant un problème de classification (voir section 2.1.2), les classes étant : présence d'une anomalie ou non. Un certain nombre d'algorithmes de classification existent tels que les Réseaux de neurones (*Neural Networks*), la méthode des k plus proches voisins (k-NN), les Machines à vecteurs de support (SVM), etc. La sélection du meilleur type d'algorithme dépend majoritairement du type et de la forme des données à partir desquelles faire l'apprentissage. Par exemple, Y. Sung [Sung 2016] détaille l'utilisation de réseaux bayésiens (*Bayesian networks*) pour la détection d'intrus dans un environnement composé de capteurs Bluetooth, en mesurant les indicateurs de puissance des signaux reçus (*Received Signal Strength Indication* or *RSSI*). E. Hodo et al. [Hodo 2016], quant à eux, présentent un IDS basé sur les réseaux de neurones artificiels (ANN) capables de détecter des attaques

de déni de service dans des environnements IoT, avec 99.4% de précision sur des attaques simulées.

Notre problème étant principalement associé à une seule classe de comportement pouvant être labellisée, seul un algorithme d'apprentissage automatique de type classification à partir d'une classe (*One-Class Classification* ou *OCC*) peut être utilisé. En outre, notamment vis-à-vis de la taille des données collectées et malgré la nécessité de réduire ces dernières pour obtenir des temps de traitement réalistes, un algorithme de type réseaux de neurones semble le plus adapté à mettre en place. En effet, celui-ci est principalement utilisé dans le cas de problématiques de classification et permet d'obtenir un modèle rapidement à partir d'un nombre conséquent de données, contrairement à des modèles comme SVM qui converge très lentement.

Notre approche inclut un modèle qui représente les activités légitimes dans l'environnement connecté à partir des spectrogrammes collectés par les sondes. Ce modèle peut détecter des anomalies qui diffèrent significativement du comportement appris. Parmi les nombreux modèles qui ont été développés, nous avons choisi le réseau de neurones de type auto-encodeur, qui fournit des résultats très intéressants dans la littérature sur des cas d'applications variés. Pour être plus précis, un certain nombre de travaux [Jiang 2017] se basent sur l'utilisation d'un réseau de neurones de type auto-encodeur dans le cadre de la détection automatique d'erreurs de conduite à partir d'images de vidéo-surveillance.

Sachant que les sondes produisent un flux ininterrompu de mesures, il nous faut les découper en morceaux pour entraîner l'auto-encodeur. Pour détecter s'il existe une anomalie dans une coupe d'un spectrogramme, nous utilisons le modèle appris pour produire la reconstruction de cette coupe. Si le spectrogramme original est "normal" et ne contient aucune attaque, l'erreur de reconstruction sera faible puisque l'auto-encodeur est construit autour de la minimisation de cette erreur. Cependant, si une anomalie est présente, cette erreur devrait augmenter.

Le facteur principal qui nous guide dans le choix de l'auto-encodeur repose sur le fait qu'il fournit une erreur de reconstruction par attribut de l'entrée. Cette propriété est utile dans le cas d'un diagnostic pour permettre de déterminer la fréquence d'une anomalie (voir section 6.4.2). La seconde raison correspond à la possibilité pour ce modèle d'apprendre à partir d'une quantité importante de données, ce qui est le cas dans notre solution. Comme nous ne faisons aucune hypothèse sur les caractéristiques des activités radios, nous ne pouvons pas facilement réduire les dimensions du problème. Par exemple, nous ne réduisons ni notre écoute à des bandes fines ni la résolution fréquentielle et temporelle. Un grand nombre de modèles de détection d'anomalies, comme les réseaux bayésiens, les modèles de Markov ou les SVM ne s'adaptent pas à tant de données d'apprentissage. Ainsi, au vu de la complexité du problème et de cette quantité de données, le compromis biais-variance expliqué dans la section 2.1.3 nous amène à choisir l'auto-encodeur comme modèle.

La sous-section suivante détaille son fonctionnement et ses avantages dans notre solution.

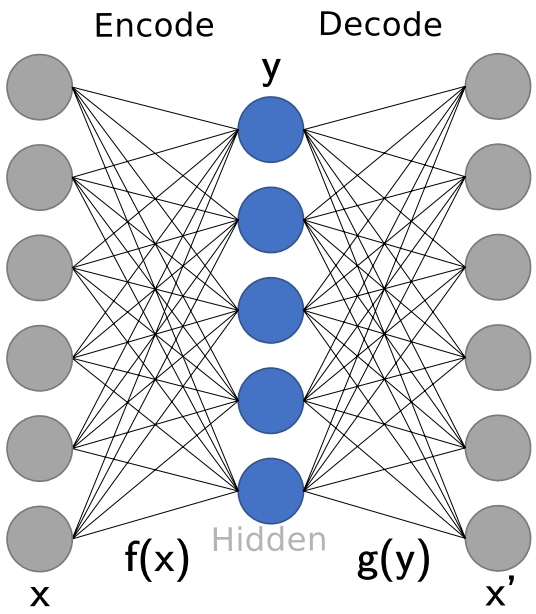

FIGURE 3.4 – Exemple d'auto-encodeur

3.5.2 Auto-encodeur

Un auto-encodeur [Ng 2011] est un modèle d'apprentissage semi-supervisé appartenant à la famille des réseaux de neurones profonds qui va essayer de reconstruire en sortie des données identiques aux données d'entrées. Un exemple simple d'auto-encodeur est représenté sur la figure 3.4. Il va chercher à apprendre une représentation (un encodage) des données d'entrées. Il est structuré comme un réseau de neurones non-récurrent avec le même nombre d'entrées et de sorties, ainsi qu'un ensemble de couches cachées incluant un goulot d'étranglement. Celui-ci apprend une représentation compressée des entrées (l'encodage) permettant de les reconstruire le plus précisément possible en sorties.

Un auto-encodeur est composée de deux parties. La première encode les entrées jusqu'au goulot d'étranglement y, comme une fonction $y = f(x)$. La seconde est composée des autres couches qui vont étendre les dimensions de y, comme une fonction $g(y)$. L'objectif principal de l'auto-encodeur est de déterminer ces deux fonctions tel que $x' = g(f(x)) \approx x$ où x correspond aux entrées. Le processus d'apprentissage correspond à la minimisation d'une fonction de coût $L(x, g(f(x)))$ où L peut par exemple être une erreur quadratique moyenne.

L'auto-encodeur est donc un algorithme d'apprentissage de reconstruction des données d'entrée. Or, dans notre solution, il s'agit de classifier si une observation est une anomalie ou une communication légitime. Cependant, si on considère que l'apprentissage ne s'effectue que sur des communications légitimes, l'auto-encodeur est un moyen efficace de définir la classe de ces communications, puisqu'il construit un modèle capable de reconstruire efficacement ces dernières. Dans ce cas, si une observation contient une activité radio qui n'a jamais été observée lors de l'appren-

tissage, celle-ci sera mal reconstruite et l'erreur entre les entrées et les sorties sera élevée. Un détecteur pourra donc se baser sur la mesure de cette erreur pour déterminer s'il s'agit d'une activité légitime ou d'une anomalie. Un exemple de ce type de résultat est présenté sur la figure 3.5.

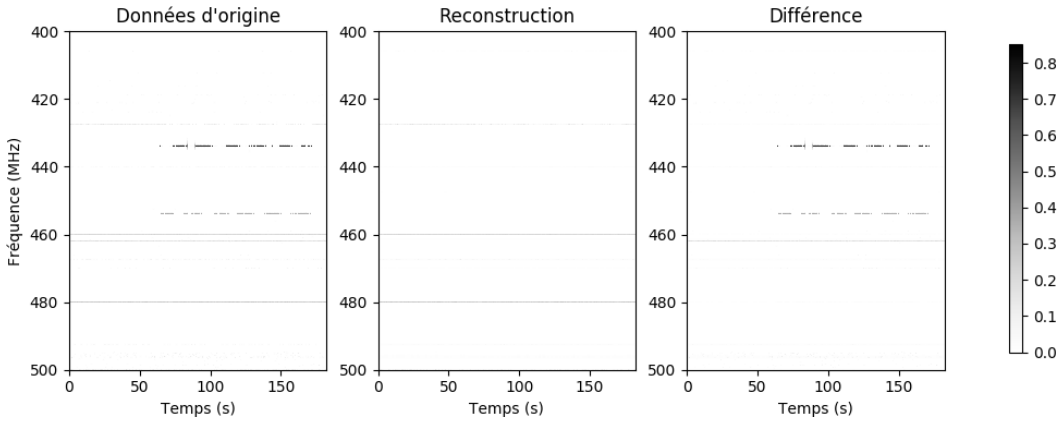

FIGURE 3.5 – Un spectrogramme, sa reconstruction et la différence entre les deux. La bande noire pointillée étant une attaque de Bruijn. [Kamkar 2015].

3.6 IDS

Lorsque le modèle de référence des activités radios a été établi, l'IDS peut être mis en place dans l'environnement en le connectant aux différentes sondes. Ce composant implémente le modèle précédent de manière statique, et utilise celui-ci de manière à pouvoir détecter les déviations des comportements radios. Cette étape correspond à la *phase de détection*. Les spectrogrammes générés par les sondes ne sont cette fois-ci pas stockés par un composant hors-ligne mais directement traités à la volée par l'IDS. Dans cette phase, les sondes génèrent en temps réel des spectrogrammes des activités observées et les transmettent à l'IDS. Celui-ci peut ensuite les utiliser directement ou choisir de les pré-traiter en fonction de l'implémentation du détecteur.

Contrairement à la phase d'apprentissage, l'IDS n'a pas besoin ici d'apprendre un modèle à partir d'un grand nombre de données, ce qui impose une architecture puissante et coûteuse. Lorsque le modèle est établi par le composant hors-ligne précédent, l'IDS n'a plus qu'à implémenter une version statique de celui-ci, et de mettre en entrée les données reçues au fur et à mesure. Ainsi, ce composant ne nécessite pas d'architecture particulière, et pourrait être directement intégré à un élément de type point d'accès, de la même manière qu'un pare-feu par exemple.

Nous distinguons ensuite ici deux manières de lever une alerte et de prévenir l'utilisateur : la première consiste principalement à être en mesure de fournir la présence ou non d'une anomalie, ce qui permet de prévenir l'utilisateur d'une potentielle tentative d'intrusion, la seconde permet de fournir à l'utilisateur des éléments de

diagnostic, à partir de la détection d'une anomalie.

3.6.1 Détection

Dans le cadre d'une simple détection, les spectrogrammes traités sont fournis en entrée au modèle de référence établi qui va essayer de les reconstruire. Cette reconstruction permet de mesurer ce que nous appellerons *l'erreur de reconstruction*, qui correspond grossièrement à la différence entre les sorties obtenues et les entrées fournies. Lorsque l'auto-encodeur ne parvient pas à reconstruire les observations récoltées, celle-ci va augmenter. Cette mesure d'erreur est ensuite analysée par l'IDS qui va lever une alerte si elle augmente de manière significative pendant un temps donné ou si elle dépasse un seuil. Il est également possible de mesurer le cumul de cette erreur pour pouvoir lever une alerte lorsque celui-ci évolue rapidement. Ces différentes manières d'analyser cette mesure d'erreur sont appelés les mécanismes de génération d'alarmes. Le paramétrage de ces mécanismes est établi au préalable, en fonction du niveau de sécurité voulant être atteint (par exemple pour le seuil, si celui-ci est élevé, le détecteur sera moins sensible à l'évolution de l'erreur, tandis que si celui-ci est bas, il lèvera plus d'alertes, dont probablement quelques fausses alarmes). Dans le cas d'une détection, l'alerte consiste simplement à donner le moment où une anomalie a été détectée, c'est-à-dire le moment où l'erreur de reconstruction a déclenché le mécanisme de génération d'alarmes.

3.6.2 Diagnostic

Dans le cadre d'un environnement peu complexe, par exemple un domicile connecté, avec une connaissance exacte des objets, de leurs comportements et des utilisateurs, une alerte permet d'opérer rapidement un diagnostic manuel sans conséquences graves pour l'ensemble de cet environnement. Cependant, lorsque celui-ci est composé d'un grand nombre d'utilisateurs qui manipulent potentiellement leurs propres objets, rajoutant un dynamisme et une incertitude forte dans les comportements radios, il peut être nécessaire de fournir aux utilisateurs ou à un expert sécurité des informations complémentaires qui peuvent l'aider aux opérations d'urgence à mettre en place en cas d'intrusion. C'est par exemple le cas dans des usines connectées ou des environnements de travail connectés.

Les hypothèses de notre approche ne nous permettant pas d'obtenir le contenu des messages échangés, ni d'identifier avec précision l'émetteur des communications, il est nécessaire de définir les informations pouvant être remontées par le détecteur lorsque celui-ci détecte une anomalie. Tout d'abord, le diagnostic temporel est intéressant puisqu'il permet de borner le moment où une anomalie s'est produite. Cette information peut-être affinée en fonction des paramètres du détecteur et de la résolution temporelle de nos observations, elle est cependant déjà fournie dans le cas d'une alerte simple. Une autre information intéressante est la fréquence à laquelle a été repérée l'anomalie. Celle-ci permet notamment de commencer à identifier s'il s'agit d'objets connus de l'environnement, ou s'il s'agit d'objets introduits

pour exfiltrer des informations par exemple. Elle permet également de restreindre l'analyse seulement à certains objets. La dernière information de diagnostic pouvant être utilisée via des observations radios est l'information spatiale : sachant que les sondes observent les communications depuis plusieurs points distincts, il peut être possible d'obtenir des informations de localisation d'un émetteur en fonction de la puissance observée par les sondes. Sachant que l'hypothèse n'est pas d'obtenir une localisation spatiale en intérieur au centimètre, mais de permettre de cloisonner un espace comme étant celui à l'origine de l'anomalie.

Pour permettre de remonter ces éléments de diagnostic, un certain nombre de modèles spécifiques doivent être mis en place. Ces éléments seront détaillés dans la partie IV.

3.7 Conclusion

Ce chapitre aborde donc une description de l'approche récapitulée par la figure 3.1 présentée en début de chapitre. Il présente les différents éléments mis en oeuvre dans la solution pour permettre la détection d'anomalies dans un environnement connecté soumis aux hypothèses faites sur l'attaquant en début de chapitre. Ce chapitre présente également des éléments génériques permettant l'implémentation et le déploiement des sondes, ainsi que les spécificités liées à la génération du modèle et à son utilisation dans le cadre de la détection. Les deux prochaines parties présentent deux instanciations de cette approche dans des contextes précis et différents. La première concerne un déploiement au sein de domiciles connectés, ainsi que les différentes expérimentations permettant d'évaluer l'efficacité de notre solution et d'éprouver son principe. La seconde présente quant à elle un déploiement au sein d'un environnement professionnel, plus particulièrement focalisée sur les aspects modèles et apprentissage automatique. Cette partie détaillera également l'ensemble des expérimentations réalisées pour évaluer la pertinence de notre approche dans ce contexte.

Troisième partie

Déploiement et évaluation pour les domiciles connectés

Chapitre 4

Contexte et spécificités d'implémentation

Sommaire

4.1 Introduction **71**

4.2 Contexte des domiciles connectés **71**

4.2.1 Caractéristiques considérées et moyens mis en œuvre 72

4.3 Implémentation et déploiement de l'approche pour des domiciles connectés **73**

4.3.1 Implémentation de la phase d'apprentissage 74

4.3.2 Détection d'anomalies – IDS 79

4.4 Conclusion **80**

4.1 Introduction

Ce chapitre présente les problématiques spécifiques aux domiciles connectés, imposant un certain nombre de contraintes sur le déploiement de la solution et sur l'implémentation de ces composants. Nous commençons dans une première section par définir les spécificités du contexte dans lequel nous cherchons à déployer notre solution, un domicile connecté. Ensuite, nous présentons les caractéristiques propres à l'implémentation, tout d'abord concernant la mise en place des différents composants de l'approche, puis vis-à-vis de la phase d'apprentissage du modèle de référence. Finalement, les mécanismes de détection employés par l'IDS sont détaillés dans une dernière section avant de conclure.

4.2 Contexte des domiciles connectés

Le contexte associé au premier déploiement effectué pour l'approche générique concerne les domiciles connectés. Ceux-ci possèdent des caractéristiques propres, imposant certaines spécificités dans l'implémentation de notre solution, particulièrement du point de vue de l'apprentissage du modèle et des mécanismes de détection employés par l'IDS.

4.2.1 Caractéristiques considérées et moyens mis en œuvre

La première caractéristique des domiciles connectés est leur stabilité. En effet, le domicile d'un particulier est considéré plus stable qu'un environnement connecté de type industriel ou professionnel. Les utilisateurs d'un domicile sont en grande majorité clairement définis à l'instar des objets installés en son sein. Dans le cas d'un milieu plus professionnel, comme nous l'avons vu dans l'état de l'art des environnements IoT et comme nous en discuterons dans la prochaine partie, la prise en compte du dynamisme des entrées et sorties des utilisateurs, par exemple vis-à-vis des invités ou des nombreux intervenants extérieurs, est important. Au contraire, même si cela reste une généralisation et peut être non représentatif de ménages spécifiques, un milieu particulier se base sur un noyau d'utilisateurs au nombre limité, clairement identifiés et évoluant dans un environnement à la fois restreint en taille, mais également en nombre d'objets. Il est donc plus aisé de contrôler et de vérifier les comportements du type *Bring Your Own Devices (BYOD)*, qui consiste, pour des personnes extérieures, dans le fait de venir avec ses propres objets dans un environnement protégé.

Deuxièmement, une autre caractéristique des domiciles connectés concerne justement l'identification des objets utilisés par les utilisateurs du domicile. Préalablement à l'installation d'une architecture de sécurité, la liste des objets utilisés ainsi que leurs caractéristiques peut être définie. Lorsqu'un nouvel objet apparaît, celui-ci est donc rapidement identifié. De plus, dans le cas d'un visiteur aux intentions malveillantes, la connaissance du moment auquel une anomalie a été perçue pourrait permettre d'identifier rapidement le responsable. En outre, l'intérêt pour les utilisateurs dans ce contexte repose presque exclusivement sur leur capacité à fournir des informations à un organisme d'assurance pour être indemnisé lors d'un cambriolage ou lors de vols d'informations par exemple. Ainsi, l'objectif de l'approche ici est donc d'être de capable de certifier la détection d'une activité malveillante en tant que preuve pour ces organismes. Ainsi, nous avons choisi de limiter les fonctionnalités de l'IDS à la détection uniquement dans ce contexte. En effet, un simple mécanisme de détection permettrait d'obtenir dans la majorité des cas suffisamment d'informations pour permettre de prendre des mesures adéquates. De plus, la détection limitant ces informations à la date où une anomalie a été détectée, celle-ci pourrait être utilisée sans aucun problème pour prouver la présence d'une activité malveillante.

Une autre caractéristique concerne le profil des utilisateurs d'un domicile connecté. Ces derniers n'ont que très rarement l'expertise nécessaire pour mettre en place une architecture de sécurité complète tout en étant en mesure de la paramétrer. Il faut donc être en mesure de limiter au maximum la complexité liée à l'installation d'une solution telle que celle présentée dans ce document. Or, les éléments de diagnostic pouvant être mis en place dans notre approche ont pour objectif de permettre une analyse approfondie des caractéristiques d'une anomalie perçue via la remontée d'informations techniques. Les utilisateurs d'un domicile ne possédant pas l'expertise pour réaliser cette analyse, ce besoin est limité dans notre

contexte. Ce manque d'expertise influe également sur le déploiement et la configuration de notre approche. Par conséquent, ces étapes peuvent être réalisées au travers d'un expert de sécurité qui se déplace jusqu'au domicile. Cette hypothèse est raisonnable puisqu'elle est déjà utilisée pour la mise en place de système d'alarme détectant les intrusions physiques (placement des caméras, réglages, etc.).

Finalement, la dernière caractéristique de ce type d'environnement concerne le coût de mise en place d'une solution de sécurité dans un environnement particulier. En effet, dans les milieux professionnels, la contrainte de coût est limitée puisque l'objectif est souvent de protéger le fonctionnement global de l'entreprise ou de limiter des pertes importantes. Cependant, ce n'est pas le cas de l'environnement d'un particulier qui va chercher à utiliser une solution efficace sans se ruiner. Encore une fois, le diagnostic étant plus compliqué à mettre en oeuvre, notamment en terme d'implémentation et de visualisation, sa mise en place aura très certainement un impact sur le coût de la solution. Ceci renforce notre choix pour une solution de détection uniquement. Concernant le coût du déploiement, nous avons fait le choix dans notre implémentation de n'utiliser qu'une seule sonde pour couvrir les communications en présence, ce qui devrait être suffisant dans la plupart des cas. Bien entendu, cela dépend encore une fois fortement de l'espace dans lequel la solution est déployée. Si ce dernier est grand, une seule sonde ne pourrait potentiellement pas observer des communications éloignées sur des hautes fréquences qui traversent moins les murs. Le nombre de sondes à installer peut être estimé par rapport à la perception des échanges radios observés sur la plus haute fréquence à monitorer. Si une seule sonde ne permet pas d'écouter toutes les communications, alors une seconde est nécessaire, et ainsi de suite. Dans cette partie, nous prenons l'hypothèse d'un domicile type appartement d'une centaine de m^2. Concernant l'IDS, celui-ci peut être installé, comme expliqué précédemment, directement au sein d'un équipement de sécurité déjà présent dans les domiciles comme le point d'accès, ce qui permet de limiter encore une fois le coût de déploiement de notre solution.

4.3 Implémentation et déploiement de l'approche pour des domiciles connectés

Pour pouvoir mettre en place l'approche détaillée précédemment dans n'importe quel environnement connecté de type domicile, il est nécessaire de définir un certain nombre d'éléments généraux qui facilitent son implémentation. L'objectif de cette section est donc de détailler ces derniers, tout en respectant les contraintes discutées précédemment liées à ce type d'environnement dans lequel la solution doit s'intégrer.

L'IDS de la solution proposée a pour objectif de réaliser la détection des anomalies pouvant être perçues par les sondes. Il s'agit donc dans un premier temps de déterminer un modèle des communications légitimes via de premières observations : la phase d'apprentissage. Dans un second temps, la phase de détection, réalisée par l'IDS, consiste à calculer les déviations vis-à-vis de ce modèle pour détecter des anomalies en temps réel et lever des alertes en conséquence. Pour l'implémenta-

tion de la modélisation des activités légitimes, il est nécessaire de définir un certain nombre d'éléments importants. Tout d'abord, la manière dont sont pré-traités les spectrogrammes pour pouvoir être analysés par le modèle d'apprentissage automatique défini, l'auto-encodeur. Dans un second temps, il est nécessaire de comprendre comment doit se construire le modèle à partir des hypothèses faites sur les données et sur les objectifs de détection établis.

Cette section détaille donc les éléments d'implémentation des différents composants. En considérant que l'implémentation de la sonde est générique à l'architecture présentée, il est nécessaire par ailleurs de détailler celle du générateur du modèle de référence et de l'IDS, notamment vis-à-vis du coût d'implémentation et du paramétrage pour une mise en oeuvre adaptée au contexte.

4.3.1 Implémentation de la phase d'apprentissage

Concernant la phase d'apprentissage, celle-ci doit principalement s'effectuer à partir des observations considérées exemptes d'attaques durant un temps à déterminer. Ce temps doit être suffisant pour couvrir l'ensemble des comportements pouvant être observés lors de l'utilisation des objets connectés du domicile considéré. Sachant que le comportement des individus peut être différent en fonction des périodes, il peut être nécessaire de générer des modèles qui prennent en compte ces distinctions, en définissant par exemple un modèle pour la semaine et un modèle différent pour le week-end. Dans ce cas, un minimum d'une semaine d'observations est obligatoire.

Ensuite, les traitements consistent à : 1) extraire les attributs considérés pour réaliser l'apprentissage et 2) générer le modèle à partir de l'auto-encodeur implémenté. La première partie, dont l'implémentation est discutée dans la sous-section suivante, est générale à la phase d'apprentissage et de détection et ne nécessite qu'une puissance de calcul minimale qui peut tout à fait être réalisée par n'importe quel ordinateur ou architecture récente, et pourrait donc tout à fait s'intégrer à un équipement déjà présent dans le domicile de la même manière qu'un pare-feu par exemple. Elle pourrait également être réalisée directement sur la sonde. En revanche, le second traitement, c'est à dire la génération du modèle, nécessite une capacité de calcul suffisante pour réaliser l'apprentissage, ce qui nécessite parfois des accélérations matérielles via des unités de traitements graphiques (GPU). Nous considérons que dans le cas d'une implémentation, la phase d'apprentissage peut être donc réalisée hors-ligne sur une machine dédiée à partir des observations collectées. Deux solutions sont ici envisageables. Soit cette machine est un équipement loué par l'utilisateur possédant les caractéristiques nécessaires à l'apprentissage du modèle. Soit les utilisateurs enregistrent grâce à la sonde les observations requises puis les fournissent à l'expert ayant réalisé le déploiement, qui effectuera l'apprentissage du modèle sur un équipement adapté. Cette deuxième solution est cependant discutable du point de vue de la vie privée, les observations permettant d'identifier certains comportements des utilisateurs, notamment leur présence dans le domicile. La première solution est facilement envisageable puisqu'aujourd'hui, des architec-

tures type nano-ordinateurs [1] sont proposées par Nvidia à moins de 100€, permettant d'implémenter des modèles d'apprentissage à faible coût. Dans notre cas, nous avons choisi d'utiliser un serveur de calcul dédié pour accélérer les expérimentations, c'est-à-dire composé d'un processeur i7-7700 à 3.6 GHz et d'un GPU MSI GeForce GTX 1080 Ti avec 32 Go de RAM, pour un coût total d'environ 1600 €.

4.3.1.1 Traitement des données et extraction des attributs

Pour générer un modèle représentatif des communications radios légitimes, l'auto-encodeur requiert la définition d'attributs représentatifs de ces données d'entrée. Ces attributs ou *features* en anglais, doivent être extraits des spectrogrammes générés à partir des données collectés par les sondes. Ces spectrogrammes pourraient être directement utilisés comme entrées de l'auto-encodeur, de la même manière que pour les modèles de classifications d'images. Cependant, la très grande quantité des données collectées et le besoin de détection en temps réel nécessaire à notre approche rendent difficile l'utilisation de ces représentations brutes par du matériel peu coûteux. En outre, l'extraction d'attributs permet de diminuer la complexité de la prédiction faite par le modèle, et ainsi d'implémenter la phase d'apprentissage et la phase de détection sur du matériel peu coûteux et peu contraignant pour des particuliers.

C'est pourquoi le choix a été fait de traiter au préalable ces spectrogrammes pour en extraire des attributs représentatifs des communications effectuées. Par la suite, les attributs extraits doivent correspondre aux métriques principales qui caractérisent au mieux un signal radio, en prenant en compte les trois éléments pouvant être interprétés d'un spectrogramme : la fréquence, la puissance et le temps.

Concernant la fréquence, un spectrogramme W_i ou une bande de fréquence spécifique de ces W_i peuvent être utilisés. Plus précisément, chaque spectrogramme est découpé selon la bande de fréquence à analyser ou selon les fréquences intéressantes à observer. Si nécessaire, une bande de fréquence spécifique peut être découpée en sous-parties à intersections non nulles. Ainsi, un spectrogramme peut être découpé pour se focaliser sur les différents canaux d'un protocole connu (par exemple les 16 canaux du protocole Zigbee).

Concernant la puissance d'un signal observé, des métriques intéressantes correspondent à la distribution statistique des données : le maximum, le minimum, la moyenne, la médiane, l'écart-type et la somme par exemple. L'intérêt premier de ce type d'attributs repose sur le fait qu'ils sont très faciles et rapides à extraire des spectrogrammes enregistrés, et peuvent donc facilement être utilisés à la volée pour détecter une anomalie. Bien entendu, certaines métriques peuvent être inutiles pour permettre d'apprendre efficacement les comportements, comme par exemple la somme qui est équivalente à une moyenne à un facteur près. Le minimum peut sembler inintéressant de prime abord puisqu'il ne donnera que la puissance du bruit. Cependant en fonction de la bande de fréquence observée, un changement de niveau de bruit peut-être représentatif d'une attaque. Nous avons choisi ici d'utiliser

1. https ://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/

l'ensemble de ces métriques, pour ne pas limiter le modèle a priori, l'ajout ou le retrait d'une de ces métriques n'ayant de toute manière qu'un impact extrêmement minime sur la vitesse de traitement des spectrogrammes.

Finalement, concernant le temps, certains attributs peuvent être pris en considération comme l'horodatage t d'un spectrogramme. Cette notion temporelle est importante car elle peut aider le modèle à identifier des comportements répétés tous les jours par des utilisateurs (éteindre l'alarme tous les soirs en rentrant du travail par exemple). Ainsi, des comportements répétés tous les jours dans les mêmes tranches horaires sont appris par notre modèle d'apprentissage. De plus, pour prendre en compte le comportement et l'évolution des activités radios de manière plus locale et ainsi rajouter une dimension temporelle à notre modèle, les attributs précédents sont extraits sur une fenêtre glissante correspondant à un nombre R de spectrogrammes.

L'auto-encodeur minimise par la suite sa fonction de coût en utilisant un algorithme de rétropropagation du gradient ou *backpropagation* en anglais pour définir le modèle des communications et des comportements radios légitimes.

4.3.1.2 Pré-traitement et définition des attributs

Pour mettre en place la solution, et vis-à-vis des contraintes précédentes liées à l'environnement, il est nécessaire de pré-traiter les spectrogrammes provenant de la sonde radio pour limiter la quantité de données à partir desquelles l'IDS va devoir détecter des anomalies. Lors de la phase d'apprentissage, il est nécessaire d'être cohérent par rapport à l'IDS, et le modèle doit donc être conçu à partir des données de même forme. Comme défini dans la section abordant le traitement des données, cette phase de pré-traitement consiste principalement dans l'extraction d'attributs représentatifs des activités radios. Ces attributs sont donc définis à partir de métriques récupérables à partir de spectrogrammes. En considérant un intervalle de fréquences $f_s..f_e$ d'un spectrogramme S, ces métriques sont définies ainsi :

$\Omega(S, f_s, f_e) = \{\ S[l,k] \mid f \in [f_s..f_e],\ l \in [0..N[\ \}$

$\mathrm{Max}(S, f_s, f_e) = \max\ \Omega(w, f_s, f_e)$

$\mathrm{Min}(S, f_s, f_e) = \min\ \Omega(S, f_s, f_e)$

$\mathrm{Mean}(S, f_s, f_e) = \mathrm{mean}\ \Omega(S, f_s, f_e)$

$\mathrm{Median}(S, f_s, f_e) = \mathrm{median}\ \Omega(S, f_s, f_e)$

$\mathrm{Std}(S, f_s, f_e) = \mathrm{std}\ \Omega(S, f_s, f_e)$

$\mathrm{Sum}(S, f_s, f_e) = \sum\ \Omega(S, f_s, f_e)$

Pour prendre en compte les comportements des utilisateurs d'un domicile, il est également intéressant d'extraire des informations temporelles. L'horodatage t défini pour chaque spectrogramme permet d'avoir une représentation des dates à laquelle telle ou telle activité radio est observée. Cependant, un horodatage est une représentation linéaire du temps, et les comportements radios pouvant se répéter tous les jours ne peuvent donc pas être identifiés lors de l'apprentissage.

Pour pallier ce problème, nous avons implémenté la solution proposée par [London 2018]. Il s'agit de représenter un horodatage sous la forme d'un cosinus

et d'un sinus à partir des secondes pour modéliser la périodicité d'une journée. L'utilisation de ces deux fonctions permet de représenter à l'aide d'une fonction complexe cet horodatage. En effet, une fonction réelle continue et périodique passe plusieurs fois par le même point lors d'un cycle, ce qui n'est pas le cas d'une fonction complexe. Ainsi, un sinus égal à zéro et un cosinus égal à un correspond à minuit. Nous avons donc deux attributs $sin(t)$ et $cos(t)$ pour décrire t. Les comportements répétés quotidiennement sont donc appris par le modèle, qui les considèrent comme légitimes quand ils sont effectués dans certaines plages horaires de la journée. Dans notre implémentation, nous n'avons considéré que la périodicité des comportements à la journée. Bien entendu, il pourrait être intéressant de repérer les comportements qui sont effectués tous les samedis par exemple, en définissant la périodicité à la semaine.

En outre, il peut également être intéressant de prendre en compte les évolutions temporelles des activités radios à plus petite échelle. Les comportements automatiques, comme un enchaînement correspondant à l'interconnexion de plusieurs objets communiquants entre eux peuvent être ainsi appris. Par exemple, certaines ampoules connectées s'allument et s'éteignent lorsqu'une requête est émise depuis le téléphone vers une passerelle en WiFi, celle-ci va alors émettre la commande correspondante en Zigbee. Pour cela, nous définissons une fenêtre temporelle de R spectrogrammes permettant de prendre en considération ces évolutions. Nous avons donc en entrée les six attributs pour chacun des R derniers spectrogrammes, nous donnant ainsi une mémoire de $R \times (N \times T)$ secondes (sachant que $N \times T$ correspond au temps d'observation d'un spectrogramme). Ainsi, plus R augmente, plus cette mémoire est grande et permet d'apprendre l'évolution des activités radios dans une fenêtre temporelle large. Cependant, l'augmentation de R influe sur la quantité d'attributs en entrée du modèle, ainsi que sur la quantité de données nécessaires pour apprendre correctement celui-ci. Un compromis est donc à trouver.

En résumé, les attributs précédents sont donc extraits sur chacun des spectrogrammes de la fenêtre temporelle. Le nombre d'attributs en entrée de l'auto-encodeur est donc de $R \times 8$ dans notre cas, avec 8 le nombre d'attributs par spectrogramme correspondant aux 6 métriques définies précédemment ainsi qu'aux deux attributs de date.

L'extraction de ces attributs s'effectue en temps linéaire et peut donc très bien être réalisé par n'importe quelle architecture non spécifique. Cela peut donc être réalisé par l'équipement dédié à l'apprentissage du modèle de référence et par l'IDS.

4.3.1.3 Construction et implémentation du modèle

Pour construire et générer un modèle des activités radios légitimes à partir des attributs précédemment définis, il est à présent nécessaire de détailler l'implémentation et l'architecture de l'algorithme d'apprentissage considéré dans la solution, c'est-à-dire l'auto-encodeur. Notre implémentation a été réalisée via Tensorflow [2]

2. https ://www.tensorflow.org/

qui utilise notamment nativement certaines accélérations matérielles intéressantes dans notre cas.

Les différents hyperparamètres à déterminer dans le cadre d'un apprentissage à partir d'un auto-encodeur sont : 1) l'architecture de ce dernier, c'est-à-dire le nombre de couches et le nombre de neurones sur chaque couche, 2) la ou les fonctions d'activation implémentées dans chaque neurone et 3) la fonction de coût à mettre en oeuvre pour réaliser l'apprentissage.

Dans un premier temps, l'architecture implémentée est définie comme étant composée de :

— les entrées, composées de $R \times 8$ attributs ;

— une couche dense correspondant à 80% des entrées ;

— un goulot d'étranglement, une couche dense correspondant à 75% des entrées ;

— une autre couche dense de 80% des entrées ;

— une couche de sortie, composée de $R \times 8$ sorties.

Une couche dense dans un réseau de neurones correspond à la structure la plus simple mise en œuvre dans ce type de modèle. Dans cette structure, chaque neurone de la couche est connecté à tous les neurones de la couche précédente. En raison de la forte connectivité qui en résulte (s'il y a n nœuds dans les deux couches, il y a n^2 poids à apprendre), l'apprentissage peut nécessiter beaucoup de données et un long temps de calcul. Par contre, ces couches permettent d'identifier des corrélations fines entre les attributs fournis en entrée. Dans notre implémentation, les attributs sont peu nombreux donc l'utilisation de couches denses est adaptée à notre problème.

L'architecture précédente est principalement définie à partir des expérimentations de manière empirique. Lors d'une implémentation de l'approche, cette architecture peut être modifiée pour correspondre avec l'environnement dans lequel celle-ci se met en place.

Concernant la fonction d'activation utilisée, l'état de l'art présenté dans la section 2.1 montre des résultats intéressants sur des problèmes similaires avec des fonctions d'activation *softplus* ou *sigmoid* qui sont les plus utilisées. Notre choix s'est porté sur *softplus* encore une fois de manière empirique vis-à-vis des expérimentations.

Le dernier point concerne donc la fonction de coût mise en place pour permettre à l'apprentissage d'établir un modèle correspondant aux données d'entrée. Nous avons choisi d'implémenter ici une erreur quadratique moyenne défini ainsi :

$$L = MSE(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{|\mathbf{x}|} \sum_{i=1}^{|\mathbf{x}|} (\mathbf{x}(i) - \hat{\mathbf{x}}(i))^2 \tag{4.1}$$

L'erreur quadratique moyenne ou (*MSE*) est souvent utilisée dans les auto-encodeurs car elle est simple à calculer et permet de déterminer rapidement la différence ou l'erreur de reconstruction entre deux vecteurs distincts. Dans notre cas, les entrées, les attributs extraits des activités radios, sont comparés avec les sorties reconstruites par l'auto-encodeur. Durant la phase d'apprentissage, l'algorithme

minimise cette fonction de coût en injectant les données observées, et, lorsque le coût converge, c'est-à-dire qu'il n'évolue plus, le modèle est appris et correspond à un moyen de reconstruire des sorties proches des données d'entrée. Ce modèle est ensuite sauvegardé et fourni à l'IDS qui l'utilise comme moyen de détection.

4.3.2 Détection d'anomalies – IDS

L'IDS doit, à partir du modèle établi lors de la phase d'apprentissage, permettre de détecter les anomalies dans des communications : il s'agit de la phase de détection. Lorsque le modèle de référence est mis en place, l'IDS traite de la même manière les spectrogrammes reçus par la sonde de manière continue en extrayant les attributs représentatifs précédemment définis. L'erreur est ensuite directement calculée, toujours en continu, à partir des sorties obtenues. L'IDS implémente donc simplement le modèle appris, place en entrée les attributs extraits puis récupère les sorties et les compare pour en extraire une erreur. Par la suite, c'est sur la base de cette erreur qu'une alarme est déclenchée ou non. Ce mécanisme de déclenchement d'alarme en fonction d'une erreur est appelée la génération d'alarmes.

En considérant que les phases précédentes ont été réalisées, c'est-à-dire que la phase d'apprentissage a été réalisée et que le ou les modèles de référence appris soient implémentés par l'IDS, ce dernier est alors en mesure de mettre en place la génération d'alarmes permettant de réaliser une détection d'anomalies. Dans la génération d'alarmes mise en place par notre solution, le processus permettant à l'IDS de notifier la présence d'une anomalie au sein d'une fenêtre de R spectrogrammes se déroule en 4 étapes : 1) le vecteur d'attributs correspondant aux données de R spectrogrammes est extrait ; 2) ce vecteur est fourni en entrée du modèle de référence, qui va essayer de le reconstruire à partir de ses connaissances, c'est l'étape de prédiction ; 3) une erreur est calculée pour chaque couple d'entrée-sortie selon l'équation présentée en 4.2, formant ainsi un vecteur d'erreur $Erreur$; 4) à partir des données d'apprentissage, la densité de probabilité de l'erreur pour chaque attribut a été estimée tel que présenté dans [Luvison 2009] en approchant la distribution de chaque erreur par une gaussienne. Le vecteur d'erreur établi dans l'étape précédente est donc comparé avec cette densité de probabilité. Si une erreur dépasse un seuil S, appelé seuil de détection, tel qu'un grand pourcentage des erreurs obtenues soient inférieures à celle-ci, alors ce vecteur d'erreur est considéré comme peu probable, une alarme est donc levée par l'IDS.

$$Erreur(i) = sortie_i - entree_i, i \in \{1, R \times 8\} \tag{4.2}$$

Le paramétrage de ce seuil de détection est nécessaire pour éviter de détecter un trop grand nombre de faux positifs, c'est-à-dire de lever une alerte alors qu'aucune attaque n'a été effectuée. La détermination de cette valeur seuil dans l'implémentation dépend du niveau de sécurité à mettre en place dans l'environnement, si celui-ci est très élevé, alors le seuil de détection doit être bas, quitte à avoir des alertes ne correspondant à aucune attaque. Cependant, nous considérons que dans

le cadre d'un domicile connecté, un détecteur qui lève trop souvent des alertes risque d'exaspérer les utilisateurs. Dans notre implémentation, cette valeur seuil est donc déterminée à partir d'une partie des données d'apprentissage, appelé plus tard jeu de validation : lorsque le modèle est établi, des données sans attaques sont injectées dans l'IDS et les vecteurs d'erreurs sont calculés pour chaque vecteur d'attribut. La densité de probabilité de ces vecteurs d'erreurs est ensuite estimée, et le seuil de détection est défini comme étant la valeur telle que 97% des erreurs calculées soient inférieures à ce seuil. Ce 97% est également égal au calcul du TNR (Taux de Vrais Négatifs ou *True Negative Rate* en anglais), qui correspond à calculer le taux de données classées comme étant sans attaque. Cela signifie donc que seulement 3% des erreurs calculées à partir de données sans attaque lève une fausse alerte.

Concernant l'architecture nécessaire pour mettre en place ce détecteur, celui-ci n'effectue que deux actions qui ne sont que peu coûteuses. Comme montré précédemment, le pré-traitement des spectrogrammes ne requiert pas une puissance de calcul élevée et peut être effectué sur un composant à bas coût. Quant à l'implémentation du modèle de référence, celui-ci n'impose pas non plus d'architecture particulière, puisqu'il ne s'agit que d'effectuer une série d'opérations sur un peu moins d'une centaine de nombres flottants et de comparer une différence avec une densité de probabilité pour déterminer l'erreur. L'IDS peut donc tout à fait s'intégrer à un composant tel qu'un point d'accès du domicile.

4.4 Conclusion

Dans ce chapitre nous avons vu et discuté la possibilité de déploiement de l'approche générique présentée notamment dans un type d'environnement particulier : les domiciles connectés. Nous avons tout d'abord défini les problématiques et les hypothèses considérées liés à la connectivité et aux comportements des utilisateurs et des objets dans ce type d'environnement. Ensuite, nous avons présenté les éléments d'implémentation nécessaires, du point de vue de la phase d'apprentissage et de la phase de détection, pour une installation respectueuse des contraintes précédentes. L'implémentation réalisée pour ce contexte est fonctionnelle, et le prochain chapitre détaille les expérimentations que nous avons menées pour évaluer notre solution.

Chapitre 5

Expérimentations & Résultats

Sommaire

5.1 Introduction **81**
5.2 Environnement expérimental **82**
5.2.1 Mise en place d'un environnement de test réaliste 82
5.2.2 Composition de l'environnement et comportements 83
5.3 Protocole expérimental **85**
5.3.1 Installation de la solution 85
5.3.2 Collecte des données 86
5.3.3 Apprentissage du modèle 88
5.4 Évaluation **91**
5.4.1 Protocole d'évaluation 91
5.4.2 Génération d'attaques 92
5.4.3 Injection d'attaques 94
5.4.4 Métriques d'évaluation 95
5.4.5 Résultats et discussions 97
5.4.6 Quantification du risque associée à la détection d'anomalies . 100
5.5 Conclusion **102**
5.6 Synthèse de la partie III **102**

5.1 Introduction

Ce chapitre présente les expérimentations réalisées dans le cadre du déploiement de notre solution dans le contexte d'un domicile connecté. Dans une première partie, nous détaillons la mise en place de la solution dans un environnement réel représentatif, un appartement équipé d'un ensemble d'objets connectés. Ensuite, la composition de l'environnement est présentée, notamment les caractéristiques de celui-ci vis-à-vis des objets installés et de leurs comportements, ainsi que celui des utilisateurs de cette expérimentation. Le protocole expérimental réalisé pour vérifier et évaluer la solution est par la suite décrit, présentant notamment les jeux de données à récolter pour établir le modèle de référence puis la méthode d'injection ainsi que les attaques à mettre en oeuvre pour évaluer les capacités de détection de notre approche dans cet environnement. Finalement, les résultats sont présentés et discutés dans une dernière section avant de conclure.

5.2 Environnement expérimental

L'évaluation de l'implémentation de notre solution repose sur la réalisation d'expérimentations qui soient les plus réalistes possible. Dans un premier temps, il est donc nécessaire de décrire la mise en place d'environnement expérimental cohérent et pertinent. Cette section discute donc des éléments constitutifs d'une expérimentation qui soit réaliste et simple à mettre en oeuvre, s'articulant notamment autour d'un environnement réel. La figure 5.1 présente celui qui est utilisé dans tout ce chapitre, avec la solution déployée et les différents objets qui la composent.

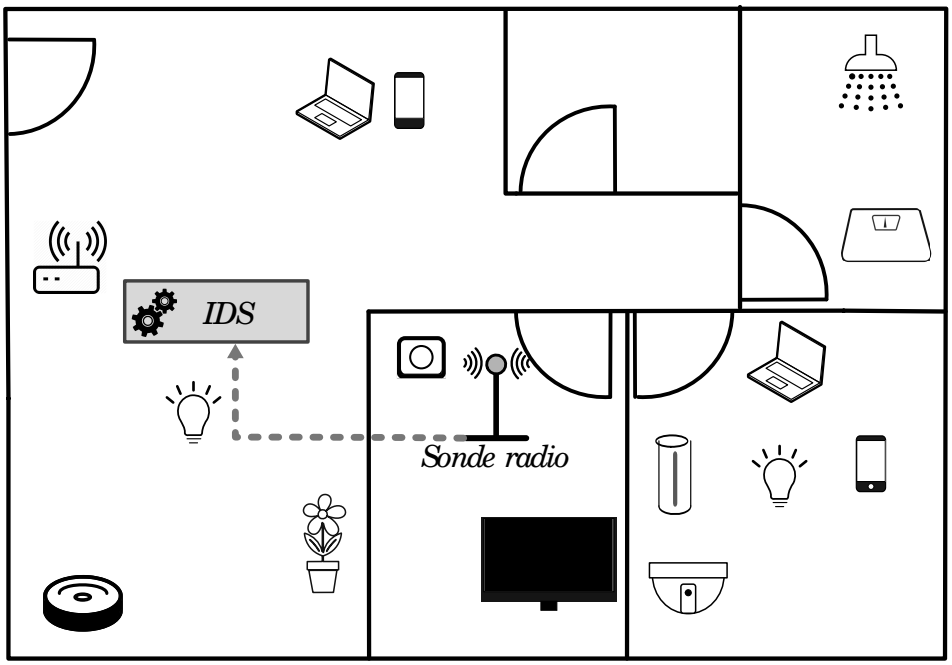

FIGURE 5.1 – Aperçu du domicile connecté expérimental

5.2.1 Mise en place d'un environnement de test réaliste

Pour réaliser un environnement expérimental de type domicile connecté, il est important de respecter les caractéristiques de celui-ci et de le rendre aussi réaliste que possible, pour montrer l'intérêt effectif de la solution en dehors d'une simple expérimentation. Tout d'abord, celui-ci doit posséder une architecture en terme de matériaux qui soit réaliste, par exemple en évitant de construire l'environnement dans une cage de Faraday. En effet, dans un environnement réaliste, les communications venant des voisins et de l'extérieur doivent être visibles. Ensuite, les objets installés dans l'environnement doivent correspondre aux caractéristiques de l'IoT pour les particuliers : 1) ils doivent utiliser de nombreux protocoles, hétérogènes et potentiellement propriétaires, 2) ils doivent être facilement trouvés dans le commerce ou facilement reproductibles, comme un capteur de présence ou de mouvement, et 3) ils doivent être également installés dans des pièces qui correspondent

à leur utilisation (un pommeau de douche connecté doit être installé dans la salle de bain). Finalement, concernant les utilisateurs eux-mêmes, ces derniers doivent être limités à un nombre restreint correspondant à un domicile réaliste (quelques personnes au maximum) avec potentiellement quelques invités, et doivent manipuler quotidiennement les objets sans forcer des comportements pour les besoins de l'expérimentation. Le meilleur moyen d'obtenir un environnement expérimental réaliste est bien entendu d'installer la solution dans un domicile connecté existant dans lequel des personnes utilisent réellement les différents objets. Dans notre expérimentation, nous avons choisi d'équiper un appartement existant d'une douzaine d'objets connectés du commerce ou simulés via un Raspberry Pi par exemple. Il est habité par deux personnes et est composé de 5 pièces pour un espace d'environ 60m^2 dans une résidence avec des voisins pouvant influer sur les communications radios. Pour éviter de forcer les utilisateurs à manipuler les objets, ces derniers ont été installés préalablement à l'expérimentation pour que les habitants les utilisent naturellement.

5.2.2 Composition de l'environnement et comportements

Comme expliqué dans la sous-section précédente, les objets installés dans l'environnement sont soumis à un certain nombre de contraintes qui vont influer sur le réalisme de l'expérimentation. Nous avons fait le choix de respecter l'ensemble de ces contraintes en installant dans l'appartement 11 objets du commerce, utilisant des protocoles connus et très utilisés au sein des domiciles connectés, ainsi qu'un objet simulé utilisant un protocole propriétaire sur une fréquence libre (433 MHz). Ces différents objets ainsi que leur protocole de communications et la bande de fréquence sur laquelle ils opèrent sont récapitulés dans la table 5.1.

Table 5.1 – Objets connectés installés et leurs caractéristiques

ID	Nom	Protocole	Fréquence (MHz)	Nombre
1	Capteur d'humidité	WiFi	2426-2448	1
2	Caméra de surveillance	WiFi	2426-2448	1
3	Téléphones, ordinateurs	WiFi	2426-2448	4
4	Pot de fleur	Bluetooth	2400-2480	1
5	Balance	BLE	2400-2480	1
6	Ampoule	ZigBee	2400-2480	2
7	Aspirateur robot	WiFi	2426-2448	1
8	Capteur simulé	Propriétaire	433-434	1

La présence de 12 objets dans une maison de 60m^2 nous permet tout d'abord de valider expérimentalement si notre solution est capable de détecter des attaques dans l'environnement malgré un trafic conséquent et très hétérogène. Ensuite, les protocoles associés à ces objets sont différents, couvrant majoritairement la bande de fréquence 2.4-2.5 GHz, qui représente un grand nombre d'objets connectés pour le grand public. Cependant, la domotique utilise souvent des protocoles proprié-

taires sur des bandes libres comme 433-434 MHz et 868 MHz, nous avons donc également choisi d'implémenter un capteur sur la bande 433-434 MHz à l'aide d'un HackRF One et d'un RaspberryPi. Celui-ci émet simplement périodiquement des données sur cette fréquence, il est donc facilement reproductible. Il simule ainsi des communications correspondant à un protocole propriétaire sur cette bande de fréquence, sans avoir besoin de créer un vrai protocole ou d'utiliser un existant, puisque seule l'activité radio nous intéresse. Les différents protocoles utilisés par les objets ont chacun leurs propres caractéristiques physiques, notamment en terme de modulation. Ainsi, les observations sont très différentes en fonction des manipulations, ce qui est une hypothèse réaliste dans la composition d'un environnement radio pour l'IoT.

Concernant le comportement radio de ces différents objets, ils ont été tout d'abord sélectionnés vis-à-vis de leurs caractéristiques très différentes au niveau de la couche physique. Cependant, l'idée était de construire un environnement avec des objets ayant des usages différents, générant ainsi des comportements distincts lorsque par exemple la caméra ou l'ampoule communique. En effet, certains de ces objets fonctionnent de manière apériodique, tels que les téléphones, les ordinateurs, la balance, les ampoules, l'aspirateur et le pot de fleur, c'est-à-dire qu'ils sont soumis à une requête de la part d'un des utilisateurs pour pouvoir générer une activité radio. Par exemple, les communications Bluetooth de la balance ne sont émises que lorsqu'une personne se pèse, ou les ampoules ne communiquent que lorsque l'utilisateur les allume ou les éteint. De plus, même dans ces comportements apériodiques, des distinctions importantes sont à identifier en terme de quantité d'informations échangées : les ordinateurs et les téléphones manipulent une grande quantité de données, générant ainsi beaucoup de trafic, tandis que l'aspirateur et le pot de fleur n'échangent que peu de messages, juste quelques commandes ou de la télémétrie. Les deux capteurs ont un comportement plus périodique, car ils s'activent et se désactivent périodiquement pour remonter des informations liées à l'environnement. Finalement, la caméra a un comportement un peu particulier, puisqu'elle est activée à chaque absence des utilisateurs de l'appartement. Lorsqu'elle est désactivée, celle-ci n'émet aucune information, tandis qu'elle diffuse un flux vidéo en continu lorsqu'elle est allumée.

Ainsi, l'environnement expérimental mis en place est très représentatif d'un environnement réel, notamment via la mise en place d'objets très différents dans leurs protocoles, mais également dans leurs comportements radios, autant en termes de quantité d'informations échangées que dans la forme de leurs communications. Il aurait été également possible d'automatiser un certain nombre d'interactions entre différents éléments, par exemple en éteignant la caméra en fonction d'un capteur de présence, mais nous avons choisi dans un premier temps de nous limiter à un ensemble d'objets indépendants.

Au niveau du comportement des utilisateurs, l'objectif est de n'imposer qu'un minimum de contraintes vis-à-vis de l'utilisation des objets. Ainsi, les utilisateurs manipulent naturellement les objets comme ils en ont l'habitude sans avoir à effec-

tuer des actions précises à des moments de la journée. La seule contrainte imposée dans notre cas concerne la caméra, qui doit être allumée ou éteinte en fonction de la présence ou non d'utilisateurs dans le domicile.

5.3 Protocole expérimental

L'environnement expérimental ayant été établi, il faut maintenant déterminer la manière dont nous allons évaluer notre solution, c'est-à-dire le protocole expérimental à mettre en oeuvre pour déterminer si celle-ci est efficace lors de la détection d'activités radios malveillantes.

Il est tout d'abord nécessaire de définir ce que l'on cherche à évaluer dans notre expérimentation. Sachant que l'objectif d'un IDS est principalement de détecter des attaques, l'évaluation de notre solution consiste à envoyer des attaques, puis à vérifier si une alerte est bien levée lors de chacune d'entre elles. Ceci correspond au *rappel* ou *recall* en anglais, qui mesure combien d'attaques injectées ont été réellement détectées par l'IDS. Ensuite, il faut vérifier que très peu voire aucune alerte n'est levée lorsqu'il n'y a pas d'attaques injectées. Il s'agit de la *précision* ou *precision* en anglais, qui mesure la proportion de détections qui sont correctes. La définition de ces termes peut être trouvée dans [Perry 1955].

Pour obtenir ces mesures d'évaluation de notre solution, il est important de définir l'ensemble des phases du protocole expérimental à mettre en place. Tout d'abord, les éléments composant la solution doivent être déployés dans l'environnement, notamment la sonde radio. Il faut donc positionner, puis paramétrer cette sonde pour pouvoir observer les activités radios des objets, notamment en définissant l'ensemble des bandes de fréquences $\{b\}$ à observer. Ensuite, la phase d'apprentissage doit être réalisée. Pour cela, il faut dans un premier temps collecter les données, en effectuant une campagne de mesures d'apprentissage, sans attaque, puis apprendre le modèle à partir de ce jeu de données. Comme nous le verrons dans la suite, nous utiliserons également celui-ci pour valider la pertinence du modèle appris. Lorsque ces deux étapes sont terminées et que le modèle est établi, il faut collecter un jeu de données permettant l'évaluation de la solution, contenant notamment des attaques. Ces différentes phases sont détaillées dans les sections suivantes avant de présenter les résultats obtenus.

5.3.1 Installation de la solution

L'environnement expérimental ayant été mis en place, il convient à présent de définir l'installation de la solution dans l'environnement. Les trois composants à installer sont : la sonde radio, le composant réalisant l'apprentissage et l'IDS chargé de la détection. Concernant le positionnement de la sonde radio, nous supposons dans ce document qu'il existe un moyen de déterminer la position optimale d'une sonde à partir d'observations préalables. Par exemple, si un seul point d'accès WiFi (situé sur la bande de fréquence 2.4-2.5 GHz) permet de couvrir l'ensemble de l'espace, alors une seule sonde placée au milieu de l'environnement sera en mesure

d'observer toutes les communications sur des fréquences plus faibles ou équivalentes. Ici, nous avons donc choisi de la placer au centre du domicile, pour être capable d'observer les communications de tous les objets répartis dans les différentes pièces. L'expérimentation ayant été réalisée dans un appartement de taille moyenne ($60m^2$) nous supposons que toutes les communications sont observables à l'aide d'une seule sonde.

Le deuxième et le troisième composant sont ceux qui vont permettre de réaliser la phase d'apprentissage, puis la phase de détection via l'IDS. Pour des besoins expérimentaux, notamment pour permettre d'effectuer un certain nombre de tests à partir des mêmes données selon différents paramétrages du modèle, nous avons choisi de réaliser ces deux étapes hors-ligne. L'IDS installé dans le domicile et présent sur la figure 5.1 ne s'occupe donc que du stockage des différents spectrogrammes observés par la sonde radio. Ainsi, nous pouvons conserver les jeux de données obtenus pour pouvoir les analyser afin d'optimiser le paramétrage de notre solution. La connectivité entre notre sonde radio et l'IDS se fait au moyen d'un réseau Ethernet, comme décrit dans le chapitre 3.4. La sonde va ensuite être paramétrée pour monitorer en continu les activités radios du domicile avant de les transformer en spectrogrammes qui seront sauvegardés par l'IDS.

5.3.2 Collecte des données

Pour pouvoir évaluer la solution, il faut tout d'abord réaliser l'apprentissage du modèle, puis vérifier les capacités de détection de l'IDS. Pour cela, deux jeux de données doivent être constitués. Le premier, le *jeu d'apprentissage*, permet d'effectuer l'apprentissage du modèle des communications légitimes. Bien entendu, aucune attaque ne doit être injectée dans ce jeu de données. Celui-ci est subdivisé en deux parties, une première permettant l'apprentissage, et une seconde permettant de valider le modèle appris. Il sert notamment à vérifier que le modèle ne lève pas ou peu d'alertes lors de communications légitimes venant du même jeu de données ainsi qu'à vérifier si celui-ci fait du sur-apprentissage sur la première partie. Le second, le *jeu de test*, permettra ensuite de vérifier les capacités de détection de l'IDS utilisant le modèle appris précédemment, notamment en injectant un certain nombre d'attaques dans les communications collectées par la solution.

La constitution de ces deux jeux données repose sur les observations effectuées par la sonde radio, il est donc important de définir dans un premier temps les différents paramètres nécessaires à la collecte. Tout d'abord, dans notre expérimentation, nous avons choisi de limiter la quantité de données en réduisant la résolution fréquentielle des balayages effectuées par le HackRF One. Pour cela, en utilisant les notations introduites dans la section 3.3.1, nous avons choisi $bw = 0.2$, soit une résolution fréquentielle correspondant à $1/bw = 1/0.2 = 5$ valeurs par MHz. Ainsi, sachant que $w = 8192$ points permettent de calculer les FFT, $V = 8000/0.2 = 40000$ valeurs par seconde sont générées en sortie du périphérique SDR. Le débit d'informations observées par les sondes est donc de 320 ko/s puisque la précision utilisée est double, soit 8 octets par mesure. Nous considérons que pour réaliser l'apprentissage

du modèle, il est nécessaire d'observer suffisamment longtemps les communications de l'environnement pour pouvoir enregistrer exhaustivement les communications pouvant être effectuées. Dans cette expérimentation, nous considérons qu'une semaine d'observation est suffisante pour satisfaire cette contrainte. En considérant 320 ko/s de données, une semaine de données correspond à environ 200 Go de données. A titre de comparaison, une semaine de données correspond environ à 1 To si l'on choisit $b = 0.04$ et un débit de 1.6 Mo/s. Bien entendu, en diminuant la résolution fréquentielle, nous obtenons moins de valeurs de puissance par fréquence, mais il faut trouver un compromis permettant de réaliser l'apprentissage hors-ligne sans avoir des besoins en terme de stockage considérables. Finalement, pour faciliter le traitement et l'observation des spectrogrammes à des fins d'analyse, ces derniers sont découpés en fichiers de $N = 100$ sweeps. Chaque jeu de données correspond ensuite à l'ensemble des vecteurs d'attributs extraits de ces fichiers. Pour rappel, 8 attributs, définis dans la section 4.3.1.2, sont extraits de chaque fichier de $N = 100$ sweeps. Un vecteur d'attributs contient les 8 attributs d'un nombre R de fichiers, R étant pour rappel le nombre de spectrogrammes au sein d'une fenêtre temporelle. Un vecteur d'attributs contient donc $8 \times R$ attributs.

Le deuxième paramètre important lié aux observations effectuées par la sonde radio concerne la bande de fréquence à monitorer. Il aurait été possible de balayer l'intégralité de la bande allant de 1 MHz à 8 GHz comme le propose le HackRF One. Cependant, pour balayer une aussi large bande, celui-ci a besoin d'une seconde. Certains des protocoles utilisés par les éléments sans-fil considérés dans notre approche utilisent des modes de communications courts, limitant à une émission brève les échanges effectués. Ainsi, si un seul balayage est effectué par seconde, la puissance reçue sur une fréquence particulière n'est observée qu'une fois toutes les secondes. Une telle émission est donc potentiellement trop courte pour être observée par un tel balayage. Plus généralement, la grande majorité des échanges s'effectue à l'échelle de la milliseconde. Une résolution temporelle d'une seule mesure par seconde est donc évidemment trop basse pour mesurer correctement ces activités radios. Ainsi, nous avons choisi de définir trois bandes de fréquences particulières à monitorer $\{b\}$: 400-500 MHz, 800-900 MHz et 2400-2500 MHz, donc un total de 300 MHz de bandes à balayer par la sonde, améliorant ainsi la résolution temporelle des balayages effectués par le périphérique SDR. En utilisant les notations introduites dans la section 3.3.1, cette dernière est définie par $1/T$. Sachant que $T = (\sum_i K_i)/V$, la résolution temporelle dépend principalement des bandes de fréquence considérées. Nous avons donc en considérant la somme des trois bandes en MHz : $K = \frac{\sum_i f_i^e - f_i^s}{b} = \frac{(500-400)+(900-800)+(2500-2400)}{0.2} = 1500$ valeurs par sweeps, soit avec $V = 40000$: $1/T = 40000/1500 = 26.67$ sweeps par secondes. Ainsi, en réduisant à ces trois bandes de fréquences, nous obtenons une résolution temporelle bien supérieure, permettant d'observer les puissances reçues pour chaque fréquence presque 27 fois par seconde. Encore une fois, il s'agit d'un compromis à trouver entre la résolution recherchée et les bandes que nous souhaitons monitorer. Ici, au vu des objets installés, vis-à-vis d'une première expérimentation et en considérant qu'une

grande partie des protocoles utilisés aujourd'hui dans le monde de l'IoT communique au sein de ces trois bandes de fréquences, le fait de nous limiter à celles-ci nous semble être un compromis acceptable pour améliorer la résolution temporelle.

Nous avons choisi de réaliser deux expérimentations différentes avec deux acquisitions de données pour évaluer notre solution. La première consiste à collecter deux semaines d'observations, soit environ 400 Go d'activités radios, sans injection d'attaque, donc sans jeu de test. Cette première expérimentation nous a permis de calibrer, paramétrer et valider le modèle. Ainsi, elle ne permet pas d'évaluer l'efficacité de notre solution dans son rôle de détection, mais permet cependant d'évaluer la pertinence de nos choix en terme de modèle et d'hyperparamètres. La deuxième expérimentation consiste en un jeu d'apprentissage de sept jours puis d'un jeu de test de deux jours dans lequel un certain nombre d'attaques a été injecté. Concernant la subdivision du jeu d'apprentissage, nous avons choisi de consacrer 70% de celui-ci à l'apprentissage du modèle et 30% pour permettre sa validation. Ce découpage n'est pas réalisé aléatoirement pour permettre d'apprendre sur l'ensemble de la période d'observation. Ainsi, l'ensemble du jeu d'apprentissage est rangé chronologiquement, et pour chaque tranche de 10 vecteurs d'attributs, les 7 premiers sont ajoutés à la partie apprentissage tandis que les 3 derniers sont ajoutés à la partie de validation.

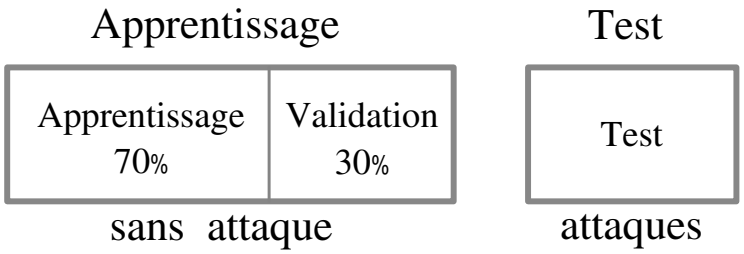

FIGURE 5.2 – Découpage des jeux de données pour les expérimentations

5.3.3 Apprentissage du modèle

Il s'agit maintenant de détailler la manière dont est réalisé l'apprentissage du modèle de référence dans ces expérimentations. Comme expliqué dans la sous-section précédente, deux acquisitions ont été réalisées, une première sans attaque de deux semaines puis une seconde contenant 9 jours de données dont 2 jours d'attaques. Pour ces deux expérimentations, la phase d'apprentissage s'effectue de la même manière. Celle-ci utilise le jeu d'apprentissage de chacune des deux collectes d'observations, c'est-à-dire la partie où aucune attaque n'a été injectée, pour apprendre un modèle de référence des activités radios effectuées dans le domicile connectée expérimental. Pour la première expérimentation, il s'agit donc de l'intégralité des deux semaines, tandis que pour la seconde expérimentation, il ne s'agit que des sept premiers jours.

Tel que détaillé dans la section 4.3 qui décrit l'implémentation de la solution dans un contexte de domicile connecté, la phase d'apprentissage s'effectue hors-

ligne. En effet, les données sont collectées au préalable puis traitées par un serveur de calcul dédié qui va s'occuper de réaliser l'apprentissage du modèle. Cette phase d'apprentissage s'effectue en trois parties. Tout d'abord, il faut traiter les spectrogrammes collectés et en extraire les attributs définis, puis réaliser l'apprentissage du modèle d'auto-encodeur implémenté sur le serveur dédié avant de valider celui-ci en utilisant une partie des données du jeu d'apprentissage.

Pour rappel, six différents attributs sont extraits de chacun des spectrogrammes compilés sous forme de fichiers : Max, Min, $Mean$, $Median$, Std et Sum. En plus de ces 6 attributs, deux autres attributs permettant de prendre en compte la périodicité d'une journée sont également extraits : $sin(t)$ et $cos(t)$, soit un total de 8 attributs. Finalement, chacun de ces attributs est extrait sur une fenêtre glissante de R spectrogrammes pour prendre en compte les évolutions des activités sur une courte échelle temporelle. Pour ces expérimentations, et sachant que chaque spectrogramme est un fichier de $N = 100$ sweeps, avec N la fenêtre temporelle d'extraction des attributs, nous avons décidé empiriquement de définir $R = 10$, correspondant à 10 spectrogrammes de 100 sweeps. L'apprentissage s'effectue donc sur une fenêtre glissante de 1000 sweeps. Sachant qu'un fichier contenant 100 sweeps correspond à $T = \frac{1500*100}{40000} = 3.75$ secondes d'observations, nous avons donc une fenêtre glissante correspondant à environ $T_f = 37.5$ secondes d'observations. Nous avons donc à extraire 8 attributs de chaque fichier de spectrogramme, puis à récupérer ces attributs sur les 10 derniers fichiers observés pour constituer un vecteur d'entrée à notre modèle. Au total, nous avons donc 80 attributs qui composent un vecteur d'entrée. Naturellement, ces 80 attributs sont normalisés entre 0 et 1 au préalable de leur injection dans le modèle.

Dans notre expérimentation, nous avons choisi de définir un modèle par bande de fréquence considérée. Ainsi, trois modèles sont appris, correspondant à chacune des 3 bandes de 100 MHz précédemment définies. En effet, les comportements sur chacune des bandes étant très différents les uns des autres, nous avons choisi d'apprendre un modèle pour chacune des bandes. Ainsi, chaque modèle est spécialisé dans la bande considérée, et est en mesure de reconstruire plus efficacement les activités radios observées. L'idée étant de ne pas se limiter en nombre de modèles appris dans la solution, chaque bande est ensuite elle-même découpée en sous-bandes de fréquence correspondant à des canaux de communication des protocoles connus (par exemple les 16 canaux du Zigbee). Cette hypothèse est réalisable lorsque la spécification d'un protocole est connue au préalable, et lorsqu'il peut être intéressant de se concentrer sur un des canaux de communication plutôt que sur une large bande globale (par exemple les canaux d'*advertisments* du BLE). Bien entendu, sachant que l'hypothèse reste de pouvoir détecter des attaques survenant sur des protocoles non-connus au préalable, les modèles basés sur les bandes de 100 MHz sont également utilisés. En tout, 25 modèles différents ont été définis pour ces expérimentations correspondant aux découpages en MHz des trois bandes de fréquences :

— Modèles généraux : 400-500, 800-900, 2400-2500 ;

— Modèles Zigbee (16 canaux) : 2400-2410, 2405-2415, 2410-2420, 2415-2425,

2420-2430, 2425-2435, 2430-2440, 2435-2445, 2440-2450, 2445-2455, 2450-2460, 2455-2465, 2460-2470, 2465-2475, 2470-2480, 2475-2485 ;

— Modèles BLE (3 canaux d'*advertisments*) : 2401-2403, 2425-2427, 2479-2481 ;

— Modèles WiFi (3 canaux principaux) : 2401-2423, 2426-2448, 2451-2473.

Concernant l'extraction des attributs, les fichiers de spectrogrammes sont subdivisés selon ces différents découpages, et les 80 attributs sont extraits pour chaque modèle. Dans notre expérimentation, nous avons fait le choix de définir un grand nombre de modèles pour couvrir précisément tous les canaux. Cependant, dans la réalité, il serait tout à fait possible de réduire le nombre de modèles en fonction des éléments présents dans le domicile, par exemple en connaissant le canal de communication Zigbee utilisé par les ampoules connectées et en apprenant seulement le modèle de celui-ci. Finalement, l'extraction de ces différents vecteurs d'attributs pour chaque modèle constitue les différents jeux de données qui sont utilisés pour l'apprentissage, la validation et l'évaluation. Ces jeux de données sont donc composées de l'ensemble des vecteurs d'attributs extraits à partir des fichiers de spectrogrammes. Bien que nous ayons fait le choix de restreindre notre expérimentation à des canaux précis et à des bandes de fréquences précises, cette hypothèse n'invalide pas la détection globale réalisable par notre solution sur les protocoles propriétaires et non-standards. En effet, quand les spécificités physiques d'un protocole peuvent être connues au préalable, leur prise en considération ne remet pas en question la possibilité d'observer également des attaques sur des bandes et des protocoles sans spécification connue.

Pour réaliser l'apprentissage en tant que tel, l'implémentation de chaque modèle est basée sur le même schéma, et est réalisée via la librairie Tensorflow sur le serveur de calcul dédié. L'architecture a été définie dans la section 4.3, et consiste principalement en un modèle d'auto-encodeur composé de 5 couches : une d'entrée composée de 80 entrées, une couche dense de $80 * 0.875 = 70$ neurones, un goulot d'étranglement de $80 * 0.75 = 60$ neurones, puis une couche dense également composée de 70 neurones et finalement d'une couche de 80 sorties qui va reconstruire les sorties à partir des entrées en minimisant la fonction de coût L définie par une erreur quadratique moyenne (MSE). Comme expliqué dans la section 5.3.2, le jeu d'apprentissage est subdivisé en deux parties : celle d'apprentissage et celle de validation.

L'apprentissage est considéré terminé lorsque le modèle converge, c'est-à-dire que le coût n'évolue plus. Pour une semaine de données, il faut environ une dizaine de minutes pour réaliser l'apprentissage sur le serveur de calcul dédié. Lorsque l'apprentissage est réalisé, chaque modèle est sauvegardé pour être ensuite validé à l'aide des données de validation. Cette dernière étape a pour objectif de vérifier que lorsque de nouvelles données légitimes non utilisées pour l'apprentissage sont fournies au modèle, celui-ci est capable de les reconstruire efficacement : notre méthode de détection expliquée dans la section 4.3.2 ne doit lever quasiment aucune alerte, puisqu'aucune attaque n'est effectuée.

5.4 Évaluation

Cette section est dédiée à la présentation de l'évaluation des performances de notre solution dans un contexte de domicile. L'objectif de cette évaluation est de vérifier les capacités de l'IDS à détecter des activités radios malveillantes à partir des observations radios effectuées par la sonde radio, et ce dans le contexte d'un domicile connecté. Lorsque l'apprentissage est réalisé tel que présenté dans la section précédente, les modèles de référence sont établis et l'IDS peut donc être mis en place. Dans notre évaluation, nous avons choisi pour des besoins d'analyse et de paramétrage d'utiliser une version hors-ligne de notre IDS, notamment en ayant collecté par avance les jeux de données qui vont nous servir à évaluer notre solution. Cependant, il serait tout à fait possible de le mettre en place dans l'environnement.

5.4.1 Protocole d'évaluation

L'évaluation de notre solution s'effectue en injectant des attaques volontaires dans l'environnement, pendant que la sonde radio enregistre les activités radios. Pour évaluer, il est important que ces attaques soient les plus réalistes possibles, notamment en étant représentatives d'attaques utilisées dans ces environnements. L'IDS doit ensuite être en mesure de détecter ces attaques en utilisant les modèles de référence appris lors de la phase d'apprentissage. L'évaluation se déroule donc en 3 étapes : 1) une campagne d'attaque est réalisée, où des attaques réalistes sont définies puis injectées dans l'environnement, 2) le jeu de test est constitué grâce aux observations réalisées par la sonde radio durant l'injection d'attaques puis analysé par l'IDS et 3) les alarmes levées par celui-ci sont reportées puis utilisées pour calculer les métriques permettant d'évaluer les performances de détection.

Les métriques étant utilisées pour évaluer les performances de notre solution sont celles habituellement calculées dans tous les systèmes comportementaux, notamment ceux utilisant l'apprentissage automatique. Ces dernières sont basées sur la récupération des alarmes levées par l'IDS et sur la pertinence de celles-ci, définies par les termes : Faux Positifs (FP), Vrais Positifs (TP), Faux Négatifs (FN), Vrais Négatifs (TN). Ces métriques sont ensuite utilisées pour mesurer la précision et le rappel, respectivement *precision* et *recall* en anglais. La précision correspond au taux d'alertes pertinentes (TP) c'est-à-dire correspondant à des vraies attaques par rapport à toutes les alertes ($FP+TP$), même celles ne correspondant à aucune attaque (FP). Le rappel correspond au taux d'attaques détectées (TP) par rapport à toutes les attaques effectuées ($FN+TP$), même celles non détectées. Ces métriques permettent également de calculer le TNR ou taux de vrais négatifs qui servira par la suite à définir le seuil de détection. La précision, le rappel et le TNR sont donc définis comme suit :

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$
$$TNR = \frac{TN}{TN + FP}$$

Ces différentes mesures sont réalisées pour chaque modèle, c'est-à-dire pour chaque détecteur vis-à-vis des attaques réalisées sur les bandes concernées par celui-ci.

5.4.2 Génération d'attaques

La campagne d'injection d'attaques utilisée dans notre évaluation se déroule en deux étapes. Tout d'abord, il s'agit de définir les attaques qui seront injectées, ainsi que leurs objectifs dans l'évaluation. Ensuite, il s'agit de définir comment sont injectées ces attaques.

Notre approche de détection reposant sur l'analyse des activités radios, il n'est pas nécessaire de réaliser des attaques réelles ayant un impact sur les données échangées par les objets et l'activité malveillante. Notamment, les charges utiles malveillantes n'ont pas besoin d'être exactes mais doivent permettre de simuler un impact sur les activités radio qui puisse être représentatif d'une attaque effectuée. Une attaque est donc paramétrée par : 1) l'intensité à laquelle elle est émise, 2) sa durée et 3) la fréquence sur laquelle elle est effectuée. L'intensité est définie comme étant soit "Forte" soit "Normale". La notion d'intensité étant très différente en fonction du protocole et de la fréquence, dans cette expérimentation, nous considérons qu'une puissance "forte" correspond à une émission effectuée volontairement à une puissance plus élevée que les émissions normales effectuées habituellement par les objets utilisant ce protocole ou cette fréquence. Une puissance normale, quant à elle, correspond à une émission qui respecte les limites de puissance autorisées sans essayer de les dépasser. Par exemple, la limite de puissance autorisée sur la fréquence 868 MHz est d'environ 26 dBm, ainsi, une attaque est considérée "forte" si elle dépasse volontairement cette limite. Il aurait pu être intéressant d'évaluer également l'impact de la position de l'émetteur malveillant, mais nous avons choisi de nous focaliser sur la détection à une position fixe. Toujours dans un objectif de réalisme dans notre expérimentation, les attaques injectées sont toutes basées sur des exemples existants dans la littérature et sont donc également facilement reproductibles.

Nous avons sélectionné un jeu de huit attaques qui sont détaillées dans le tableau 5.2. L'ensemble de ces attaques permet de vérifier un certain nombre d'éléments que l'IDS doit être en mesure de détecter, correspondant au modèle de menaces défini dans la section 3.1.

Attaques 1 et 8 Les attaques 1 et 8 sont représentatives d'attaques de type Déni-de-Service ou DoS. Elles correspondent à une émission malveillante d'un message à

Table 5.2 – Listes des attaques injectées

ID	Protocole	Type	Intensité	Durée	Fréquence
1	WiFi	Déni-de-Service (DoS)	Forte	20 min	2430 MHz
2	WiFi	Dé-authentification	Normale	1 min	2437 MHz
3	WiFi	PA pirate	Normale	4 min	2412 MHz
4	BLE	Homme-du-milieu	Normale	4 min	2400-2500 MHz
5	Zigbee	Fausse association	Normale	1 min	2470 MHz
6	Zigbee	Envoi de données	Normale	4 min	2470 MHz
7	868MHz	Envoi de données	Forte	1 min	868 MHz
8	433MHz	Déni-de-Service (DoS)	Forte	10 min	433 MHz

forte puissance puis d'un arrêt des communications radios de l'objet correspondant, respectivement le point d'accès du domicile pour l'attaque 1 et le capteur simulé pour l'attaque 8. Ces attaques permettent donc de vérifier si l'IDS est capable de repérer un changement de puissance perçue des activités radios, ou s'il est capable de repérer un changement dans le comportement des communications, visible par l'arrêt des activités d'un objet légitime.

Attaque 2 L'attaque 2 correspond à une attaque de type dé-authentification [Bellardo 2003]. Celle-ci a pour objectif de déconnecter un objet ou un point d'accès WiFi en envoyant continuellement des messages de dé-authentification définis dans la spécification du protocole WiFi. Cette attaque permet d'évaluer la détection lors d'un changement de comportement dans les activités radios, puisque ces messages sont envoyés de manière répétée par l'attaquant sans différence notable dans la puissance d'émission utilisée.

Attaque 3 L'attaque 3 consiste à créer un faux point d'accès (PA) malveillant sur lequel les utilisateurs peuvent se connecter, par exemple dans l'objectif de récupérer des informations d'authentification. Cette attaque permet de vérifier si l'IDS est capable de repérer une augmentation des messages WiFi échangés (notamment due à la présence d'un nouveau PA) ou des communications sur une fréquence auparavant non utilisée (sur un autre canal que le PA légitime).

Attaque 4 L'attaque 4 consiste à réaliser une attaque du type homme-du-milieu ou *Man-in-the-Middle* en anglais sur le protocole BLE. Celle-ci consiste à se placer au milieu des échanges effectués entre un esclave et un maître [Cauquil 2016]. En se plaçant ainsi, tous les messages échangées sont doublés puisque l'attaquant doit les retransmettre à l'esclave et au maître. L'IDS doit donc être en mesure de repérer ce changement de comportement dans les communications radios.

Attaques 5 et 6 Les attaques 5 et 6 correspondent à des envois d'informations sur le canal 24 Zigbee. Le canal n'étant pas utilisé par les objets installés, ces

attaques doivent être repérées par l'IDS comme étant émises sur une fréquence habituellement non utilisée. Ces attaques correspondent à des activités malveillantes si elles sont utilisées par exemple pour s'associer avec un objet (attaque 5) ou si elles cherchent à envoyer des commandes sans authentification (attaque 6).

Attaque 7 Finalement, l'attaque 7 correspond à un envoi d'informations simulé à l'aide d'un HackRF One sur la fréquence 868 MHz. Ce type de communication peut par exemple être associé à une attaque De Bruijn [Kamkar 2015], consistant à émettre un ensemble de séquences binaires pour essayer de reproduire, sans connaissance préalable, une commande recevable par un appareil, par exemple une porte de garage. Sachant qu'aucun objet installé dans l'environnement n'utilise la fréquence 868 MHz et que la puissance d'émission est forte, l'IDS doit être en mesure de détecter cette attaque.

5.4.3 Injection d'attaques

Ces différentes attaques sont ensuite injectées dans l'environnement tandis que les activités radios sont enregistrées par la sonde radio positionnée au même endroit que celui utilisé lors de la phase d'apprentissage. La constance dans la position de cette sonde est importante pour éviter une différence de perception par la sonde, qui pourrait amener l'IDS à détecter des anomalies dans les communications.

La phase d'injection des différentes attaques précédemment définies doit être effectuée au sein de l'environnement expérimental. Pour réaliser ces injections, il faut être en mesure d'automatiser l'émission des signaux correspondant à des attaques à différents moments de la journée, notamment pour pouvoir vérifier que l'IDS est capable de repérer qu'une activité radio a lieu à un moment inhabituel. Par exemple, dans le cas de notre environnement expérimental correspondant à un domicile, les activités radios nocturnes sont assez éparses, l'IDS doit donc pouvoir détecter que l'apparition d'une activité radio la nuit peut-être potentiellement malveillante. Bien entendu, cela dépend de l'apprentissage automatique réalisé et des différents modèles appris et utilisés par l'IDS. Pour être en mesure d'automatiser nos injections d'évaluation, nous avons choisi d'utiliser un framework d'attaques IoT intitulé *Mirage* [Cayre 2019], développé au LAAS par Romain Cayre, implémentant un certain nombre de modules simples permettant de réaliser des attaques sur un grand nombre de protocoles de l'IoT. L'intérêt d'utiliser cet outil spécifique repose notamment sur sa simplicité d'utilisation et sur la possibilité de réaliser des scénarios complexes exploitant plusieurs modules distincts. Dans le cadre de notre phase d'évaluation, il nous permet d'automatiser le lancement des modules les uns à la suite des autres, sans manipulation directe de la part des utilisateurs de l'environnement. Ainsi, il intègre dans ses modules initiaux l'ensemble des attaques précédemment définies, permettant de rapidement mettre en place notre phase d'injection. Il fournit également les moyens permettant d'enregistrer un certain nombre d'informations sur l'exécution de modules, notamment leur date d'exécution. *Mirage* étant un framework relativement léger, celui-ci peut facilement être intégré à

un Raspberry Pi équipé des périphériques nécessaires pour pouvoir être placé en tant que composant automatique d'injection dans notre environnement expérimental. Pour réaliser l'injection, ce composant a donc été placé dans l'environnement à environ deux mètres de la sonde radio, à portée des différents objets qu'il pourrait chercher à attaquer.

La phase d'injection nécessite, dans le cadre d'une évaluation, d'exécuter plusieurs fois l'ensemble des attaques précédemment définies, notamment à des moments différents de la journée, pour vérifier l'impact temporel sur la détection de l'IDS. Pour cela, un module spécifique a été développé, permettant d'automatiser l'exécution des modules d'attaques de manière chaînée sous forme de campagne d'attaques. Nous avons choisi d'injecter chaque attaque individuellement pour pouvoir évaluer les capacités de détection de l'IDS. Ainsi, deux attaques consécutives sont espacées de 20 minutes. La phase d'injection a duré environ 3 jours et a consisté en 20 campagnes d'attaques incluant six attaques : les attaques 2 à 7. La durée de chaque attaque est définie dans le tableau 5.2. Chaque campagne dure 3 heures et 40 minutes et deux campagnes successives sont espacées de 1 heure. Les attaques 1 et 8 n'ont quant à elle été exécutées qu'une seule fois, chacune à différentes périodes.

Pour permettre ensuite de vérifier la détection des différentes attaques injectées, le temps d'exécution ainsi que l'identifiant de chacune d'entre elles sont enregistrés dans un fichier au moment de l'injection. Lors de la détection d'une anomalie par l'IDS, ce fichier est utilisé notamment pour vérifier si celle-ci correspond bien à une attaque injectée vis-à-vis de sa date et de sa fréquence. Pour éviter les erreurs liées à la désynchronisation temporelle entre le composant automatique d'injection et la sonde radio, ces derniers sont synchronisés grâce au protocole NTP. Les dates des observations de la sonde correspondent donc bien aux dates d'injection par le composant d'injection.

La sonde radio va donc monitorer les activités radios de l'environnement expérimental contenant les injections effectuées. L'ensemble des observations, enregistré sous la forme de spectrogrammes, compose donc le jeu de tests utilisé dans notre évaluation pour vérifier les capacités de l'IDS à détecter des activités malveillantes. Une fois constitué, celui-ci est traité pour en extraire les différents attributs, c'est-à-dire le vecteur d'entrée utilisé par nos modèles. L'IDS utilise ensuite les modèles appris lors de la phase d'apprentissage pour réaliser la détection.

5.4.4 Métriques d'évaluation

Le jeu de tests ayant été constitué à l'aide de la phase d'injection grâce aux observations de la sonde radio installée dans l'environnement expérimental, il s'agit maintenant de définir comment utiliser celui-ci pour mesurer les différentes métriques d'évaluation de notre solution. La première étape de l'évaluation consiste à traiter le jeu de tests généré pour en extraire les différents attributs utilisés par les modèles appris lors de la phase d'apprentissage. Cette étape se déroule exactement de la même manière que pour l'extraction des attributs sur le jeu d'apprentissage et utilise le même algorithme de traitement que celui présenté dans la section 4.3.

L'IDS intègre les 25 modèles définis précédemment et chaque vecteur d'attributs extrait puis reconstruit par chacun des modèles. En fonction de l'erreur de reconstruction et de la probabilité d'erreur associée, une alarme est levée par l'IDS. Ces alarmes vont nous permettre de calculer les différentes métriques d'évaluation définies, c'est-à-dire TP, FP, TN et FN, ensuite utilisées pour calculer nos deux critères d'évaluations : la précision et le rappel.

Pour nous permettre d'évaluer les capacités de détection de notre solution, un ensemble de métriques a été défini lors de la description de notre protocole d'évaluation. Nous définissons les variables booléennes suivantes correspondant à deux évènements : alarme levée Al et attaque en cours At. Nous définissons également une fenêtre de détection de cinq minutes, et si un des deux évènements survient dans les observations réalisées dans cette fenêtre, une des métriques est incrémentée comme suit :

$$\begin{aligned} Al \wedge At &\Rightarrow TP + 1 \\ Al \wedge \neg At &\Rightarrow FP + 1 \\ \neg Al \wedge \neg At &\Rightarrow TN + 1 \\ \neg Al \wedge At &\Rightarrow FN + 1 \\ TP, FP, TN, FN &\in \mathbb{N} \end{aligned}$$

La fenêtre de détection de cinq minutes nous permet d'équilibrer les métriques. Durant cette fenêtre, si une observation dépasse le seuil de détection, alors une alarme Al est levée, si celle-ci correspond à une attaque At, alors TP est incrémenté. Ainsi, l'incrémentation des métriques ne se fait que toutes les cinq minutes pour l'évaluation. Sans cette fenêtre de détection, chaque nouvelle observation (correspondant à un nouveau fichier de $N = 100$ sweeps) dépassant le seuil déclenche une alarme Al. Sachant que les attaques durent au minimum 1 minute, plusieurs alarmes Al peuvent être levées pour une seule attaque At, ce qui augmente artificiellement le nombre de TP par rapport aux autres métriques. Ainsi, en utilisant cette fenêtre de détection, le nombre de TP par rapport aux autres métriques est donc équilibré, améliorant la pertinence de l'évaluation de notre solution.

Ces métriques sont donc mesurées à partir du jeu de tests constitué et décrit dans la sous-section précédente. La section 4.3.2 décrit avec précision la méthode de détection implémentée dans notre IDS. Pour récapituler, celle-ci se base sur l'erreur entre les entrées et les sorties pour chaque attribut, à l'aide de l'équation suivante :

$$Erreur(i) = sortie_i - entree_i, i \in \{1, 80\}$$

Ensuite, la méthode de détection implémentée et définie à la section 4.3.2 va permettre de comparer ces vecteurs d'erreurs avec la densité de probabilité de chaque erreur estimée à partir des données du jeu de validation. Si l'erreur excède un seuil S donné tel que 97% (défini empiriquement) des valeurs d'erreur soient inférieures à ce seuil, une alarme est levée par l'IDS. Pour évaluer, la date de l'alarme est ensuite comparée avec les dates des attaques. Si une injection est en cours au moment

de l'alarme, alors il s'agit d'une vraie détection et donc d'un vrai positif TP, etc. Comme expliqué dans l'implémentation de la solution à la section 4.3.2, le seuil S est estimé pour chaque modèle appris de manière empirique à partir du TNR mesuré sur la partie validation du jeu d'apprentissage.

5.4.5 Résultats et discussions

La première expérimentation, basée sur le premier jeu de données de deux semaines, a été utilisée pour évaluer si nos modèles étaient en mesure de reconstruire des observations ne contenant aucune attaque sans erreur. Il ne contient donc qu'un jeu d'apprentissage, avec une partie de validation permettant d'évaluer la capacité de reconstruction des modèles appris par le biais de la partie d'apprentissage. Lorsque le taux d'erreur, c'est-à-dire le pourcentage de données de la partie de validation détectée comme anomalie, est suffisamment faible (0.1%), nous considérons que les modèles sont bien paramétrés et sont en mesure de détecter correctement des activités malveillantes.

La seconde expérimentation se compose également d'un jeu d'apprentissage, qui permet d'apprendre les différents modèles correspondant aux activités radios légitimes utilisés par l'IDS durant la phase de détection. Cependant, la partie validation a pour objectif non seulement de vérifier que nos modèles sont en mesure de reconstruire correctement les observations avec peu d'erreurs, mais également de déterminer le seuil de détection S. Pour cela, nous évaluons l'évolution du TNR en fonction de différentes valeurs de S. Les erreurs étant normalisées entre 0 et 1, nous faisons évoluer S entre 0 et 1 avec un pas de 0.1. Lorsque le TNR dépasse 97% pour une valeur de seuil, correspondant à moins de 3% d'observations sans attaque dépassant le seuil de détection, alors celle-ci est conservée comme seuil lors de l'évaluation sur le jeu de test.

Les figures 5.3 et 5.4 présentent une visualisation de l'erreur pour le modèle 800-900 MHz en fonction du temps pour respectivement la partie validation du jeu d'apprentissage et le jeu de tests de cette seconde expérimentation. En appliquant la technique d'évolution du seuil S, nous trouvons que grâce à un seuil $S = 0.6$ notre détecteur obtient un nombre de faux positifs bas, car la majorité des erreurs se situe en dessous de ce seuil. Le TNR calculé pour un seuil de 0.6 est en effet de 99.60% tel que présenté dans le tableau 5.3, c'est donc un seuil acceptable selon nos conditions. En visualisant le seuil sur la figure 5.3, nous voyons effectivement que peu d'erreurs le dépassent. La figure 5.4 montre quant à elle les erreurs obtenues à partir des observations réalisées lors de la phase de détection. Les marques bleues sur le haut du graphique identifient les dates de début de chaque injection effectuée sur cette bande, correspondant à l'attaque 7. On identifie rapidement visuellement une augmentation importante de l'erreur lorsqu'une attaque est injectée, dépassant dans la majorité des cas le seuil établi lors de la phase d'apprentissage.

L'IDS est donc en mesure de détecter des activités ayant lieu à un moment inhabituel. Les valeurs de précision et de rappel correspondant à notre évaluation sont détaillées dans le tableau 5.3. Ce tableau 5.3 décrit également les résultats obtenus

sur un sous-ensemble des modèles appris et utilisés par notre IDS, correspondant aux différents découpages considérés. Les attaques concernées par chaque modèle sont détaillées dans la colonne "Attaque ID".

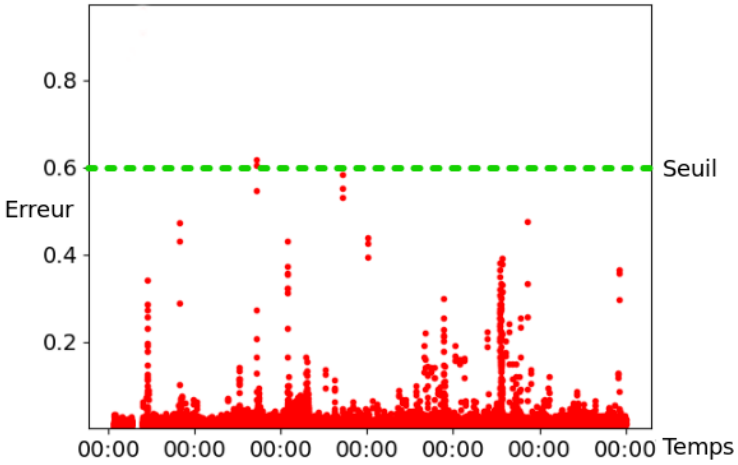

FIGURE 5.3 – Erreur jeu de validation - 800-900MHz

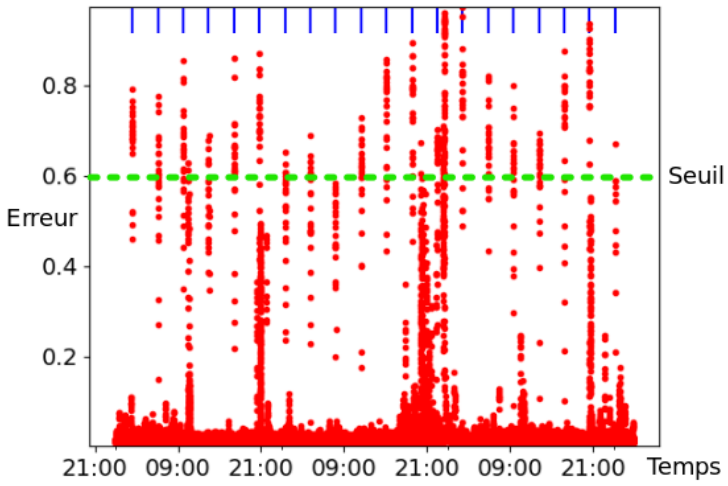

FIGURE 5.4 – Erreur jeu de test - 800-900MHz

En analysant les résultats du tableau, nous voyons rapidement que dans le cas des modèles 400-500 MHz et 800-900 MHz, quasiment toutes les attaques (7 et 8) sont détectées et peu de fausses alarmes sont levées. Ces bandes étant aujourd'hui assez peu couvertes par les solutions de sécurité existantes, ces résultats montrent déjà un intérêt non négligeable dans le déploiement de notre approche dans ce type d'environnement. En outre, la bande 800-900 MHz étant également utilisée par un

grand nombre de protocoles comme LTE ou LoRaWAN, certaines fausses alarmes correspondant par exemple à un appel téléphonique peuvent facilement être écartées en se focalisant sur une bande plus spécifique correspondant à la fréquence utilisée par les appareils domotiques (autour de 868 MHz). Nous pouvons voir notamment les résultats du modèle 860-870 MHz qui permettent d'obtenir un taux de précision de 100% pour un seuil de détection beaucoup plus faible. La fenêtre de détection mise en place nous permet également de voir que l'attaque injectée sur la bande 400-500 MHz a été détectée dans son intégralité, autant vis-à-vis de l'émission forte de départ que dans l'arrêt des communications de l'objet concerné par cette attaque. Notre IDS est donc en mesure non seulement de détecter une puissance anormale, mais également un changement de comportement des communications. La détection des attaques sur 800-900 MHz nous permet de montrer que l'IDS est en mesure de détecter des émissions effectuées sur une fréquence non utilisée dans l'environnement légitime. Lorsque nous utilisons un modèle plus précis autour de la fréquence d'intérêt, les résultats sur cette bande sont de plus grandement améliorés.

Sans surprise, les résultats sur la bande 2400-2500 MHz pour l'ensemble des modèles qui la couvrent sont fluctuants et mitigés. Le modèle qui couvre l'intégralité de cette bande ne nous permet pas de détecter efficacement les attaques injectées, celles-ci étant souvent cachées au milieu des communications constantes ayant lieu. Cette bande étant très utilisée par les protocoles sans-fil récents qui se la partagent, nous avons fait le choix de nous focaliser sur des bandes plus étroites et plus précises pouvant correspondre à des canaux de certains de ces protocoles. Ainsi, les résultats s'améliorent de manière significative : les attaques WiFi de type PA pirate (3) et DoS (1) sont relativement bien détectées, montrant ainsi que l'IDS est en mesure de détecter des changements de comportements. En se focalisant sur le premier canal d'advertisment BLE en 2402 MHz avec le modèle 2400-2410 MHz, les attaques sur ce protocole (4) peuvent être détectées en partie, notamment durant des périodes d'inactivité comme la nuit. De plus, les attaques non détectées dépendent principalement de l'utilisation d'un périphérique SDR et du paramétrage de celui-ci imposant un certain nombre de contraintes. En effet, le protocole BLE utilise un

Table 5.3 – Résultats en fonction du modèle utilisé

			Validation	Test	
Modèle	Attaque ID	Seuil	TNR	Precision	Rappel
400-500	8	0.3	99.40%	99.83%	100.00%
800-900	7	0.6	99.60%	82.76%	92.31%
860-870	7	0.2	99.80%	100.00%	96.15%
2400-2500	1,2,3,4,5,6	0.3	98.61%	83.33%	1.15%
2410-2420	3	0.3	99.00%	98.99%	79.89%
2400-2410	4	0.3	97.01%	98.55%	57.46%
2465-2475	5,6	0.4	98.01%	10.00%	1.67%
2430-2440	1,2	0.6	99.60%	100.00%	4.00%
2420-2430	1	0.4	99.60%	100.00%	100.00%

mécanisme de saut de fréquence très rapide pour échanger des messages, s'évaluant en microsecondes. La vitesse de balayage du HackRF s'évaluant en millisecondes, la résolution temporelle de ce périphérique est potentiellement trop faible pour être en mesure d'observer efficacement des communications BLE. Des articles récents montrent la possibilité d'améliorer la résolution temporelle de la SDR en modifiant la manière dont est effectué le balayage [Guddeti 2019], ces résultats pourraient donc sans aucun doute être améliorés dans le futur. Finalement, les attaques Zigbee (5 et 6) ainsi que l'attaque de dé-authentification (2) sont quant à elles très mal détectées. Concernant l'attaque de dé-authentification, cela peut s'expliquer par le fait que celle-ci est relativement courte, et s'effectue sur la même bande de fréquence que le point d'accès légitime du domicile. Une augmentation de l'activité WiFi sur cette fréquence peut donc passer pour du trafic légitime. Les attaques Zigbee sont injectées à une puissance très faible. La configuration matérielle définie dispose d'une sensibilité radio faible ne permettant pas de percevoir des signaux à faible puissance sur la bande 2400-2500 MHz. En modifiant ou améliorant le matériel utilisé, par exemple en utilisant une antenne plus sensible, nous pensons que ces attaques pourraient être détectées. Concernant l'attaque DoS sur le Wi-Fi (1), un phénomène intéressant a été observé vis-à-vis des performances de détection. Le signal émis au départ est assez mal détecté, notamment à cause du nombre important de communications sur cette bande. Cependant, les effets de l'attaque sont clairement détectés, les communications s'arrêtant soudainement tandis que certains objets WiFi du domicile se mettent à émettre un grand nombre de messages sur une courte période. De plus, après la fin de l'attaque, le point d'accès légitime s'est reconfiguré sur un autre canal WiFi. Ce changement a ainsi généré une évolution importante de l'erreur sur le canal précédent, du fait de la disparition de son activité, et sur le nouveau canal, du fait de son apparition. Dans notre expérimentation, nous avons considéré que cet effet correspondait à une attaque, puisqu'il résultait d'une action illégitime.

5.4.6 Quantification du risque associée à la détection d'anomalies

Dans le chapitre 4.3.2, nous utilisons un auto-encodeur pour modéliser les comportements radios dans l'environnement. Cet auto-encodeur nous fournit une erreur de reconstruction, à partir de laquelle nous décidons s'il y a ou non une attaque. La méthode proposée dans un premier temps ne ne permet pas de quantifier correctement le risque associé à l'erreur obtenu en l'état. Nous proposons de modifier la méthode de détection utilisée pour améliorer la compréhension des alertes levées.

La nouvelle méthode fonctionne de la manière suivante : nous apprenons les distributions des probabilités P_i des erreurs de reconstruction e_i de chaque attribut i sur le jeu d'apprentissage (sans attaques). Pour cela, nous faisons l'hypothèse que ces erreurs de reconstruction suivent une loi normale. La phase de détection se déroule de la manière suivante : un spectrogramme est placé en entrée de l'auto-encodeur, qui renvoie un vecteur des erreurs de reconstruction e_i. Notre objectif est d'obtenir la probabilité totale d'observer une erreur plus grande que l'ensemble des

erreurs mesurées et de comparer cette probabilité à un seuil : si les erreurs mesurées sont très peu probables, c'est certainement qu'il y a attaque. Nous cherchons donc à calculer : $P(x_0 \geq e_0, x_1 \geq e_1, \ldots)$. En supposant les e_i indépendants, cette formule peut se réécrire :

$$P(x_0 \geq e_0, x_1 \geq e_1, \ldots) = \prod_i P_i(x \geq e_i) \tag{5.1}$$

Étant donné qu'il s'agit d'un produit de nombres potentiellement très petits, nous allons plutôt calculer son logarithme :

$$E(e_i) = \sum_i \log P_i(x \geq e_i) \tag{5.2}$$

Par exemple, si l'erreur totale $E(e_i) = -20$, cela signifie que nous n'avions qu'une chance sur $e^{20} \approx 500000000$ d'observer cette erreur alors qu'il n'y a pas d'attaques.

La mise en place de cette nouvelle méthode de détection nous permet de mettre à jour les résultats obtenus à la fin de l'expérimentation dans les domiciles connectés. Le tableau 5.4 présente les résultats obtenus après cette modification. Dans l'ensemble, les attaques qui étaient précédemment bien détectées le restent, avec une légère amélioration des résultats sur la bande 2.4-2.5 GHz. Les cases vertes correspondent à une amélioration supérieure à 1%, tandis que les cases rouges correspondent à une perte supérieure à 1%. Bien entendu, l'intérêt de cette modification ne repose pas sur l'amélioration de la détection, mais sur une meilleure compréhension des raisons de la levée d'une alerte. En utilisant cette méthode, il est plus aisé de comprendre le risque associé à une alerte. En effet, cette méthode renseigne sur la probabilité qu'une erreur de reconstruction au moins aussi grande que celle observée arrive au hasard.

Table 5.4 – Résultats en fonction du modèle utilisé - mis à jour

			Validation	Test	
Modèle	Attaque ID	Seuil	TNR	Precision	Rappel
400-500	8	-8.5	100.00%	100.00%	100.00%
800-900	7	-29.0	99.66%	88.89%	92.31%
860-870	7	-6.1	100.00%	100.00%	96.15%
2400-2500	1,2,3,4,5,6	-5.8	99.79%	83.33%	1.15%
2410-2420	3	-3.6	97.18%	95.90%	95.38%
2400-2410	4	-4.1	98.46%	97.25%	74.47%
2465-2475	5,6	-6.1	97.26%	8.00%	3.33%
2430-2440	1,2	-14.0	100.00%	100.00%	4.00%
2420-2430	1	-6.5	100.00%	100.00%	100.00%

5.5 Conclusion

Dans ce chapitre, nous avons détaillé l'ensemble du protocole expérimental pour évaluer une implantation de notre solution dans un environnement spécifique, à savoir les domiciles connectés. L'ensemble des éléments d'évaluation, notamment la mise en place d'un environnement de test réaliste ont tout d'abord été définis, avant de détailler avec précision le protocole expérimental à mettre en oeuvre pour installer et apprendre les différents comportements radios légitimes. Finalement, la méthode d'évaluation, composée du protocole d'injection d'attaques et des métriques à mesurer, a été présentée avant de terminer sur les résultats associés à la détection de ces attaques et à leur discussion. Ces résultats indiquent un intérêt non négligeable dans l'utilisation de cette approche dans le cas des bandes de fréquence aujourd'hui assez peu protégées.

5.6 Synthèse de la partie III

Cette IIIème partie du manuscrit a donc permis de présenter une instanciation de notre approche générique dédiée à la mise en place d'une architecture de sécurité IoT dans le cadre d'un domicile connecté. Les éléments spécifiques à ce type d'environnement concernant le déploiement de notre solution ont été d'abord détaillés, notamment vis-à-vis de l'implémentation des différents composants. Une expérimentation complète a ensuite été présentée, permettant d'évaluer l'intérêt d'une approche de ce type dans cet environnement spécifique qu'est le domicile connecté. Ce dernier étant un environnement relativement stable, la prochaine partie présente l'utilisation de notre approche dans un contexte plus complexe, un environnement professionnel. Ce type d'espace est soumis à des contraintes très différentes qui imposent de mettre l'accent notamment sur les modèles mis en place et sur la mise en place d'éléments de diagnostic.

Quatrième partie

Adaptation à un environnement professionnel et modèles de diagnostic

Chapitre 6

Détection et diagnostic des intrusions radios

Sommaire

6.1 Introduction 105
6.2 Adaptation de l'approche aux environnements professionnels 106
6.2.1 Spécificités de l'environnement 106
6.2.2 Moyens mis en œuvre 107
6.2.3 Vue d'ensemble de l'approche 108
6.3 Estimation de l'erreur de reconstruction 109
6.3.1 Modèle auto-encodeur pour la détection 109
6.3.2 Pré-traitement : suppression du bruit et passage à l'échelle . 111
6.3.3 Définition de l'erreur de reconstruction pour la détection . . . 112
6.4 Diagnostics temporel et fréquentiel 113
6.4.1 Diagnostic temporel 114
6.4.2 Diagnostic fréquentiel 115
6.5 Diagnostic spatial 117
6.5.1 État de l'art sur la localisation 117
6.5.2 Fusion des anomalies 119
6.5.3 Données de calibration et apprentissage 119
6.5.4 Modèle de diagnostic spatial 120
6.6 Conclusion 120

6.1 Introduction

Ce chapitre présente les éléments qui constituent la seconde proposition d'application de notre solution générique à un environnement spécifique, les environnements professionnels. Dans un premier temps, nous détaillons les spécificités et les besoins de ce contexte. Nous présentons par la suite les modèles implémentés pour détecter des anomalies et pour fournir des éléments de diagnostic, notamment une estimation de la durée exacte de l'attaque (diagnostic temporel), sa fréquence principale (diagnostic fréquentiel) et son origine géographique estimée (diagnostic spatial).

6.2 Adaptation de l'approche aux environnements professionnels

Contrairement à un environnement particulier du type domicile, les contraintes d'un environnement professionnel sont multiples. Notamment, le grand nombre d'utilisateurs, leurs comportements distincts et le besoin de sécurité accrue de ces environnements influent sur l'implémentation de notre approche générique. Cette section décrit les contraintes de ces environnements, puis les moyens pouvant être mis en place dans l'approche pour y répondre. Finalement, nous présentons une vue d'ensemble de notre solution et de ses étapes de fonctionnement.

6.2.1 Spécificités de l'environnement

Par environnement professionnel connecté, nous considérons par la suite qu'il s'agit d'un espace communautaire au sein d'une entreprise ou d'une structure d'employés équipés d'objets connectés soumis aux contraintes évoquées dans le contexte de ce manuscrit. Nous le distinguons d'un environnement industriel, comme un chaîne d'usinage, qui est soumis à d'autres types de contraintes que nous ne cherchons pas à traiter.

Un environnement professionnel est soumis à des contraintes différentes d'un domicile. Dans ce dernier, les utilisateurs sont peu nombreux et facilement identifiables, il est donc aisé de détecter une présence anormale ou de repérer des activités radios malveillantes venant d'un invité ou d'une personne extérieure. Dans un milieu professionnel, les utilisateurs sont plus nombreux et potentiellement non authentifiés auprès de la structure dans laquelle ils évoluent. Il se peut en effet que des invités, par exemple des partenaires industriels ou des sous-traitants, se rendent dans les locaux pour effectuer des missions ou participer à des réunions. Ce grand nombre d'utilisateurs multiplie les usages et les comportements, et il devient donc complexe de définir précisément qui réalise telle ou telle action. Sans prendre en compte ces invités, les employés peuvent être nombreux, avec des horaires et des activités déjà très hétérogènes. En outre, le nombre d'objets peut être immense, et leurs caractéristiques peuvent également être différentes de celles observées dans les domiciles. Dans ce type d'environnement, les utilisateurs ont également l'habitude d'adopter des comportements du type *Bring Your Own Devices* (BYOD), consistant pour ces derniers à employer des objets personnels au sein de leur lieu de travail. Cette mobilité importante pose de nombreux problèmes de sécurité, puisque ces objets peuvent transiter dans des environnements peu sécurisés dans lesquels ils peuvent subir des attaques et être compromis, pour ensuite infecter par transitivité le réseau interne de l'entreprise. Les objets connectés constituent donc une surface d'attaque importante qui met en danger la sécurité de l'entreprise lorsque ceux-ci sont employés à la fois à l'extérieur et à l'intérieur du réseau. Finalement, les environnements professionnels sont souvent équipés de services dédiés à la sécurité des systèmes d'informations, qui ont besoin de connaître rapidement l'état de la sécurité en exploitant les échanges effectués, ainsi que les potentiels points d'entrée

vulnérables pour des attaquants. Ils doivent être en mesure d'agir rapidement dans le cas d'une détection d'attaque pour protéger les systèmes et plus généralement l'entreprise.

Ces différentes spécificités peuvent donc être récapitulées de la manière suivante :

— Plus grande surface à couvrir qu'un domicile ;
— Utilisateurs nombreux et non identifiables, aux comportements hétérogènes ;
— Objets nombreux, hétérogènes et fragilisant la sécurité interne de l'entreprise ;
— Besoin de sécurité accrue, car une entreprise a plus de risque d'être la victime d'une attaque ciblée qu'un particulier ;
— Présence d'experts de sécurité capable d'interpréter des éléments de diagnostic.

Clairement, les spécificités de l'environnement professionnel nous empêchent d'utiliser directement l'approche adaptée aux domiciles. La prochaine section détaille la manière dont nous adaptons notre architecture générale à cet environnement.

6.2.2 Moyens mis en œuvre

Pour répondre à ces différentes spécificités, nous nous devons d'adapter l'approche générique présentée dans le chapitre 3.

Surface à couvrir. Il est nécessaire, pour couvrir une grande surface, de faire reposer l'approche sur plusieurs sondes. Celles-ci doivent être positionnées dans l'environnement pour couvrir le plus exhaustivement possible les communications. L'apprentissage et l'utilisation du modèle devront donc être également adaptés à l'approche multi-sondes.

Utilisateurs et objets nombreux et non-identifiables. Cette contrainte nous pousse à créer un modèle plus complexe que dans le cas d'un domicile connecté où le comportement est bien plus simple et régulier.

Besoin accru de sécurité et présence d'experts. La simple détection d'activités illégitimes est suffisante dans le cas d'un domicile, puisque le faible nombre d'objets et d'utilisateurs permettent à ces derniers d'identifier rapidement la source de la malveillance. Au contraire, les contraintes en terme de taille et de nombre d'objets et d'utilisateurs rendent complexes cette identification dans un environnement professionnel, surtout lorsque ces objets viennent des comportements BYOD des employés. Cependant, un avantage conséquent de ces environnements est la présence d'experts de sécurité, qui peuvent analyser les levées d'alertes de l'IDS pour en chercher l'origine. Dans cet objectif, nous avons décidé d'améliorer les éléments de détection de l'IDS en proposant des mécanismes de diagnostic à partir des données monitorées par les sondes. Nous sommes en mesure de fournir à ces experts trois éléments de diagnostic pour les aider lors du traitement d'une alerte :

1. *Diagnostic temporel* : dates de début et de fin d'un anomalie ;
2. *Diagnostic fréquentiel* : fréquence centrale à laquelle une anomalie a été détectée.
3. *Diagnostic spatial* : position approximative de l'émetteur d'une activité radio anormale.

La présence de plusieurs sondes, un environnement radio plus complexe et la production d'éléments de diagnostic nous poussent à repenser un IDS qui puisse prendre en compte toutes ces contraintes. La sous-section suivante en présente une vue d'ensemble.

6.2.3 Vue d'ensemble de l'approche

Notre solution est composée de quatre blocs, en quatre phases, correspondant aux lignes pointillées sur la figure 6.1. Cette figure représente l'ensemble de notre approche et sera progressivement détaillée au travers des sections suivantes.

Le bloc "acquisition des données" est basé sur le déploiement et l'utilisation du principe de multi-sondes de notre approche. En dehors de l'utilisation de plusieurs sondes, ce bloc est implémenté de la même manière que pour l'approche générique.

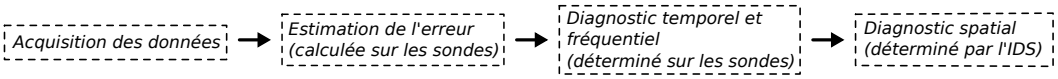

FIGURE 6.1 – Un aperçu des phases de l'approche

Les données acquises en sortie de ce premier bloc sont ensuite traitées dans un deuxième bloc, appelé "Estimation de l'erreur". Son rôle est de fournir une erreur en fonction du temps. Celle-ci est estimée par un auto-encodeur qui va chercher à reconstruire les données et calculer une erreur de reconstruction associée.

Le bloc suivant implémente deux des trois diagnostics : les diagnostics temporel et fréquentiel. Le premier estime les dates de début et de fin d'une anomalie à partir de la fonction d'erreur. Étant donné que notre approche est *demodulation-free*, c'est-à-dire qu'elle se passe de démodulation des signaux physiques (pour garder une approche indépendante des protocoles), nous ne pouvons pas déduire d'informations sur le protocole utilisé. À la place, nous décidons de calculer la fréquence centrale de chaque anomalie, basée sur les données extraites lors de la phase d'acquisition des données. Le diagnostic fréquentiel a donc pour objectif d'estimer cette fréquence centrale en mesurant la différence entre le spectrogramme mesuré et celui reconstruit durant l'anomalie (dont les dates ont été préalablement estimées). Ce bloc est exécuté localement sur chaque sonde, en temps réel. Ainsi, la majorité des données est traitée en local, ce qui limite les communications entre les sondes et minimise la surface d'attaque sur des données sensibles vis-à-vis de la vie privée.

Finalement, le diagnostic spatial est réalisé dans un bloc dédié puisque son implémentation diffère des diagnostics temporel et fréquentiel. En effet, le processus de diagnostic spatial ne peut pas être effectué en local sur chaque sonde mais sur

l'IDS, puisqu'il requiert l'agrégation des données de plusieurs sondes. Le diagnostic spatial est réalisé en trois étapes :

— Premièrement, les informations temporelles et fréquentielles associées à la détection réalisée au niveau des sondes sont agrégées. Par exemple, la première sonde pourrait détecter une anomalie de t=5 s à t=19 s avec une fréquence centrale de 868 MHz tandis que la seconde sonde pourrait repérer la même anomalie et estimer sa durée de t=4 s à t=19 s avec une fréquence centrale de 866 MHz. Ces deux détections sont regroupées en une seule.

— Deuxièmement, les activités radios correspondant à l'anomalie sont extraites en interrogeant les sondes. Ceci est possible car le système a connaissance du temps et de la fréquence de l'anomalie.

— Troisièmement, un algorithme des k plus proches voisins est utilisé pour estimer la position de l'émetteur de l'anomalie vis-à-vis de ces activités radios.

Finalement, notre solution donne trois diagnostics pour chaque anomalie détectée : ses dates de début et de fin, sa fréquence centrale et la position estimée de l'émetteur. Ces informations pourront ensuite être utilisées par des experts pour guider les investigations. Dans la suite de ce chapitre, nous nous focalisons sur la description de chaque bloc.

6.3 Estimation de l'erreur de reconstruction

L'objectif de ce bloc est de calculer une erreur en fonction du temps. Cette erreur doit être élevée en présence d'une anomalie, et faible dans le cas inverse. Nous avons choisi de définir une fonction d'erreur pour chacune des bandes de fréquences que nous monitorons, et ce pour chaque sonde. Ce choix est motivé par l'hypothèse selon laquelle les anomalies peuvent être détectées en analysant chaque bande de fréquence indépendamment. Ainsi, la phase d'apprentissage de chaque auto-encodeur est plus simple puisqu'elle se base sur une seule bande de fréquence et requiert donc peu de données pour apprendre le modèle. La figure 6.2 présente les différents éléments constitutifs de cette phase.

6.3.1 Modèle auto-encodeur pour la détection

Dans le cas d'une implémentation de l'approche pour les domiciles connectés, présenté dans le chapitre 4.3, nous avons choisi d'utiliser un auto-encodeur simple. Celui-ci est composé de 5 couches denses exploitant les informations provenant de 8 attributs extraits de chacun des spectrogrammes monitorés par l'unique sonde déployée dans l'environnement. Dans le cas d'un déploiement dans un contexte professionnel, les spécificités précédemment énoncées, ainsi que le besoin en terme de diagnostic nous obligent à repenser la structure de notre modèle d'auto-encodeur.

Dans la définition de la structure de notre modèle, nous avons fait le choix d'utiliser une première couche convolutive 1D. Pour expliquer leur utilité, le plus simple reste d'expliquer leur cas d'utilisation. Avant l'application des réseaux de

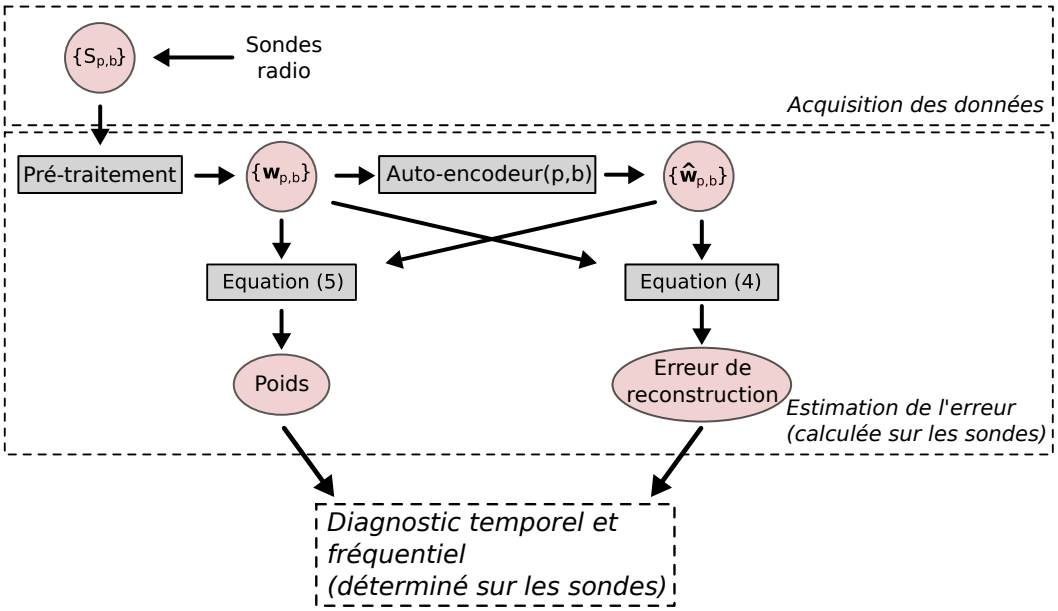

FIGURE 6.2 – Détails du bloc d'estimation de l'erreur de reconstruction

neurones à la vision artificielle, on pouvait utiliser des *noyaux* pour traiter une image. Un noyau est une opération (généralement une convolution) qui s'applique à une petite partie de l'image (par exemple un carré de 3 par 3 pixels) pour modifier un pixel en fonction des ces voisins. Ces noyaux sont donc des opérations simples et localisées appliquées à toute l'image. On peut facilement augmenter la netteté, ajouter du flou ou faire ressortir les contours avec certains noyaux. Une couche convolutive a pour objectif d'apprendre automatiquement plusieurs noyaux (appelés filtres) pour produire plusieurs images intermédiaires : un filtre peut mettre en avant certains contours, un autre détecter les changements de couleurs, etc. Encore une fois, ces filtres sont appliqués à toute l'image, ce qui signifie que si un réseau de neurones convolutif a appris à reconnaître un chat, celui-ci sera reconnu même s'il est déplacé (par translation) dans l'image : même si le chat est un peu plus à gauche, les mêmes filtres lui seront appliqués et il sera reconnu de la même manière. On dit que les couches de convolution sont invariantes par translation.

Dans notre cas, il y a une invariance par translation temporelle : un signal normal légèrement déplacé sur l'axe temporel d'un spectrogramme devra toujours être considéré comme normal. Par contre, il n'y a pas invariance par translation fréquentielle : par exemple la forme d'une communication radio WiFi ne devrait pas apparaître sur la fréquence 433 MHz. En utilisant une couche convolutive 1D, nous nous assurons d'une invariance par translation temporelle mais pas par translation fréquentielle.

Nous avons de plus fait le choix d'appliquer en sortie du goulot d'étranglement une couche dense, qui s'est révélée empiriquement plus adaptée à notre problème.

Notre architecture se compose de deux couches cachées, il y a donc quatre couches au total : 1) les entrées, 2) une couche convolutive temporelle 1D, 3) un

goulot d'étranglement correspondant à une couche dense et 4) une couche dense de sorties. Le détail de l'implémentation, notamment concernant les hyperparamètres et les paramètres structurels seront détaillés dans le chapitre 7. Un exemple de cette reconstruction est illustré plus loin sur la figure 6.4.

Notre système de détection est basé sur l'erreur de reconstruction obtenu à l'aide d'un modèle de type auto-encodeur. Le principe d'apprentissage d'un auto-encodeur peut être récapitulé comme suit : chaque coupe de spectrogramme du jeu d'apprentissage est fournie en entrée de celui-ci et une erreur d'apprentissage ou *loss* est mesurée à partir d'une fonction de coût. Tout en continuant à fournir ces informations au modèle, les poids des différents neurones du réseaux (correspondant aux paramètres appris) sont modifiés pour minimiser cette erreur.

6.3.2 Pré-traitement : suppression du bruit et passage à l'échelle

Une fonction de coût très utilisée dans l'apprentissage d'un auto-encodeur est la racine carrée de l'erreur quadratique moyenne (RMSE en anglais) entre les entrées $\mathbf{x}$ et les sorties $\hat{\mathbf{x}}$ qui peut être exprimé ainsi :

$$RMSE(\mathbf{x}, \hat{\mathbf{x}}) = \sqrt{\frac{1}{|\mathbf{x}|} \sum_{i=1}^{|\mathbf{x}|} (\mathbf{x}(i) - \hat{\mathbf{x}}(i))^2} \tag{6.1}$$

Cependant, nous ne pouvons directement utiliser cette fonction de coût à cause de la forte proportion de bruit présente dans les spectrogrammes. Voici un exemple afin d'illustrer ce problème. Nous définissons $S_{p,b}$ correspondant au découpage d'un spectrogramme comme défini dans la section 3.3.2, qui consiste en 99% de bruit et 1% de signal. Cette proportion n'est pas absurde, puisque nous monitorons de larges bandes de fréquence. Les valeurs du bruit varient autour d'une valeur moyenne. Comme le bruit est aléatoire, l'auto-encodeur ne peut que l'estimer à partir de cette valeur moyenne. Supposons que l'erreur absolue moyenne de la reconstruction du bruit soit de 5 dB. En supposant que l'auto-encodeur ne reconstruit pas parfaitement le signal, l'erreur moyenne absolue est de 30 dB. La racine de l'erreur quadratique moyenne est donc de $\sqrt{\frac{1}{100}30^2 + \frac{99}{100}5^2} \approx \sqrt{9+25} \approx 5.83$.

Comme nous pouvons le voir dans cet exemple, le bruit aléatoire influe largement sur l'erreur d'apprentissage, et donc sur l'apprentissage du modèle, en minimisant l'impact du signal effectif sur ce dernier. Ce phénomène est amplifié par le fait que ce signal peut être présent seulement dans une faible proportion des fréquences monitorées. Pour résoudre ce problème et améliorer l'apprentissage du modèle, nous proposons de supprimer l'erreur de reconstruction du bruit en saturant ce dernier à une valeur unique. Plus précisément, les mesures inférieures à un seuil minimum B_l sont fixées à 0, et celles supérieures à un seuil maximum B_u sont fixées à 1, tandis que le reste des mesures est normalisé linéairement entre 0 et 1. Cette normalisation entre $[0; 1]$ est adaptée à l'algorithme d'apprentissage de l'auto-encodeur.

Plus formellement, nous définissons [1] $\mathbf{w}_{p,b}$ le découpage d'un spectrogramme $S_{p,b}$

1. Pour des questions de clarté, nous omettrons par la suite p, b et écrirons $\mathbf{w}$ et S.

et sa fonction de coût correspondante :

$$\mathbf{w}(t,f) = \begin{cases} 0 \text{ if } S(t,f) < B_l \\ \frac{S(t,f)-B_l}{B_u-B_l} \text{ if } B_l \leq S(t,f) \leq B_u \\ 1 \text{ if } S(t,f) > B_u \end{cases} \tag{6.2}$$

$$RMSE(\mathbf{w}, \hat{\mathbf{w}}) = \sqrt{\frac{1}{|\mathbf{w}|} \sum_t \sum_f (\mathbf{w}(t,f) - \hat{\mathbf{w}}(t,f))^2} \tag{6.3}$$

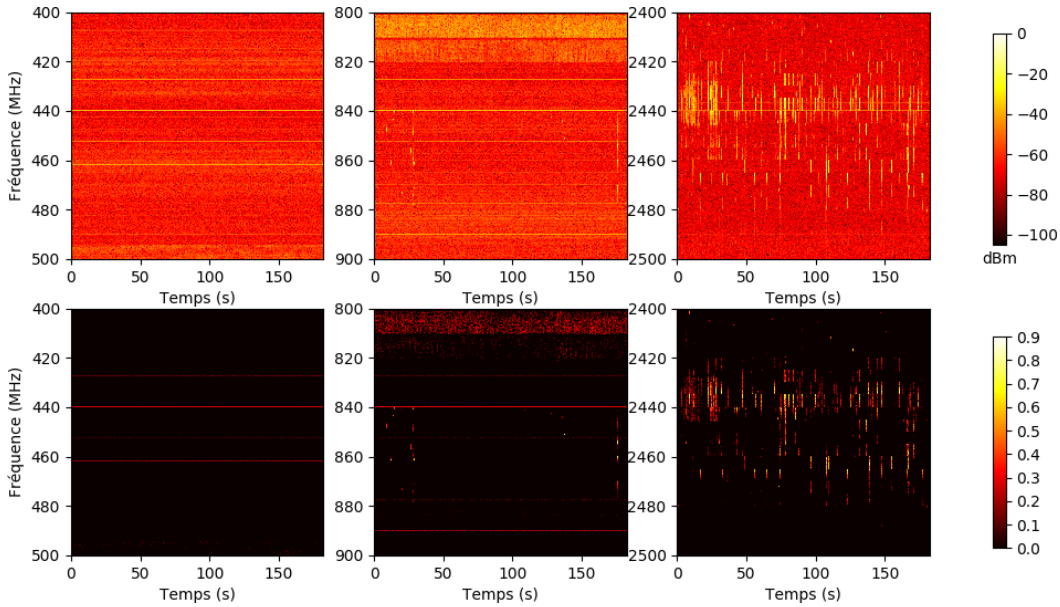

FIGURE 6.3 – Un spectrogramme (en haut) et sa version pré-traité (en bas)

Un exemple de l'effet de ce pré-traitement est présenté dans la figure 6.3. Les trois spectrogrammes du haut correspondent à trois bandes monitorées non pré-traitées, tandis que celles du bas sont leurs équivalents pré-traités. Comme nous pouvons l'observer, le bruit a disparu. Inévitablement, certains détails pertinents sont également perdus.

6.3.3 Définition de l'erreur de reconstruction pour la détection

Les modèles sont entraînés en minimisant la racine de l'erreur quadratique moyenne des entrées reconstruites à l'aide de l'*adam optimizer* [Kingma 2015]. Pour rappel, cette erreur est appelée *loss* ou erreur d'apprentissage. Nous appliquons une fenêtre temporelle glissante sur les spectrogrammes pour générer nos entrées. Deux fenêtres consécutives se superposent à 80% pour maximiser la quantité de données disponibles lors de l'apprentissage.

L'estimation de l'erreur est basée sur la différence entre les spectrogrammes et leurs reconstructions. Sur la figure 6.4, un spectrogramme est présenté à gauche,

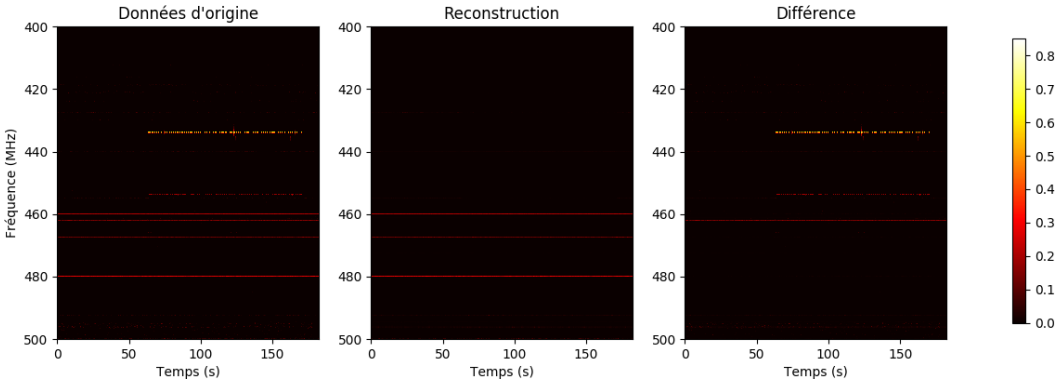

FIGURE 6.4 – Un spectrogramme, sa reconstruction et leur différence absolue. La ligne jaune pointillée horizontale correspond à une attaque "de Bruijn"

sa reconstruction au centre et leur différence absolue sur la droite. La ligne pointillée jaune horizontale est une attaque. Nous pouvons observé que celle-ci n'est pas reconstruite sur la figure centrale, ce qui est mis en avant dans la dernière figure représentant la différence.

Bien que l'auto-encodeur ait appris en minimisant la racine de l'erreur quadratique moyenne des entrées reconstruites, nous n'utilisons pas cette métrique dans le cas de la détection pour la même raison que celle expliquée dans la section 6.3.2 : les anomalies n'affectant généralement qu'une partie limitée du spectre monitoré, l'erreur d'apprentissage associée peut être diluée dans les nombreuses petites erreurs du trafic normal. Pour limiter cet effet, nous nous assurons que l'erreur de reconstruction utilisée pour la détection, correspondant plus simplement au terme *erreur de reconstruction* dans notre approche, pénalise plus spécifiquement les erreurs significatives. Pour cela, nous utilisons la norme 4 de la distance : $d_4(a,b) = \sqrt[4]{(a-b)^4}$. Nous adaptons donc la RMSE à cette distance pour obtenir la définition suivante de l'erreur de reconstruction :

$$RE(\mathbf{w}, \hat{\mathbf{w}}) = \sqrt[4]{\frac{1}{|\mathbf{w}|} \sum_t \sum_f (\mathbf{w}(t,f) - \hat{\mathbf{w}}(t,f))^4} \tag{6.4}$$

Pour chaque entrée, nous produisons un score à l'aide de l'erreur de reconstruction définie par l'équation (6.4). Pour obtenir une meilleure résolution temporelle de cette erreur, nous utilisons une fenêtre temporelle glissante sur les spectrogrammes utilisés en entrée. Ainsi, deux fenêtres consécutives se superposent à 20%. Cette reconstruction d'erreur est ensuite traitée par le localisateur temporel.

6.4 Diagnostics temporel et fréquentiel

L'objectif de ce bloc est de déterminer une estimation des dates de début et de fin d'une attaque ainsi que de sa fréquence centrale. Ces informations peuvent être ensuite utilisées pour identifier un attaquant, une attaque ou le type d'objet ciblé.

La figure 6.5 détaille le contenu de ce bloc.

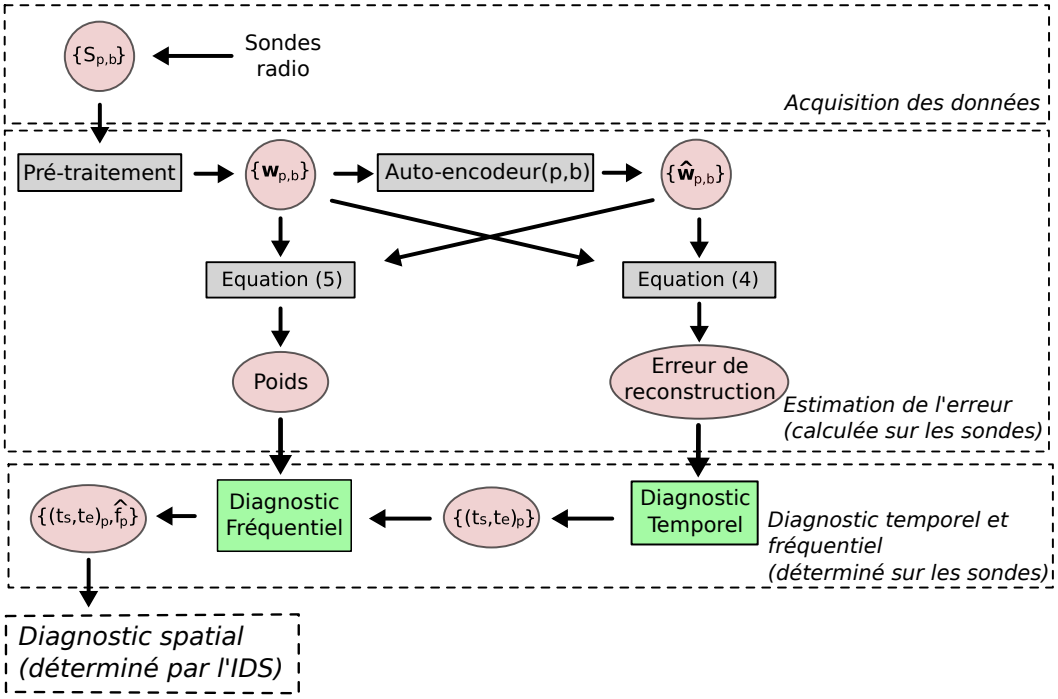

FIGURE 6.5 – Détails du bloc de diagnostic temporel et fréquentiel associés aux blocs précédents

6.4.1 Diagnostic temporel

Pour détecter une anomalie, le localisateur temporel analyse les séries temporelles de l'erreur de reconstruction et vérifie si celle-ci est trop élevée, c'est-à-dire si elle franchit un seuil $\tau(t)$. Ce seuil peut être appris à partir du jeu d'apprentissage puisqu'il dépend des activités radios de l'environnement. Un exemple de série temporelle est présenté sur la figure 6.6.

Deux contraintes entrent en compte lors du choix de ce seuil. Premièrement, l'environnement ne peut être parfaitement contrôlé, ce qui signifie que l'apprentissage peut contenir des anomalies. Par exemple, durant nos expérimentations (présentées dans la section 7.2), l'activité radio d'un objet Bluetooth apparaît brièvement au sein des données d'apprentissage. Ceci nous empêche de positionner le seuil à l'erreur de reconstruction maximum mesurée lors de l'apprentissage. Pour estimer un seuil robuste, nous calculons le 99-percentile des séries temporelles du jeu d'apprentissage pour chaque bande et chaque sonde. Ensuite, l'erreur de reconstruction est corrélée avec le trafic, qui varie au cours du temps. Pour cette raison, le seuil est une fonction du temps $\tau(t)$. Par exemple, le seuil peut être plus faible la nuit, puisque ces périodes correspondent à une réduction de l'activité au sein d'un environnement professionnel).

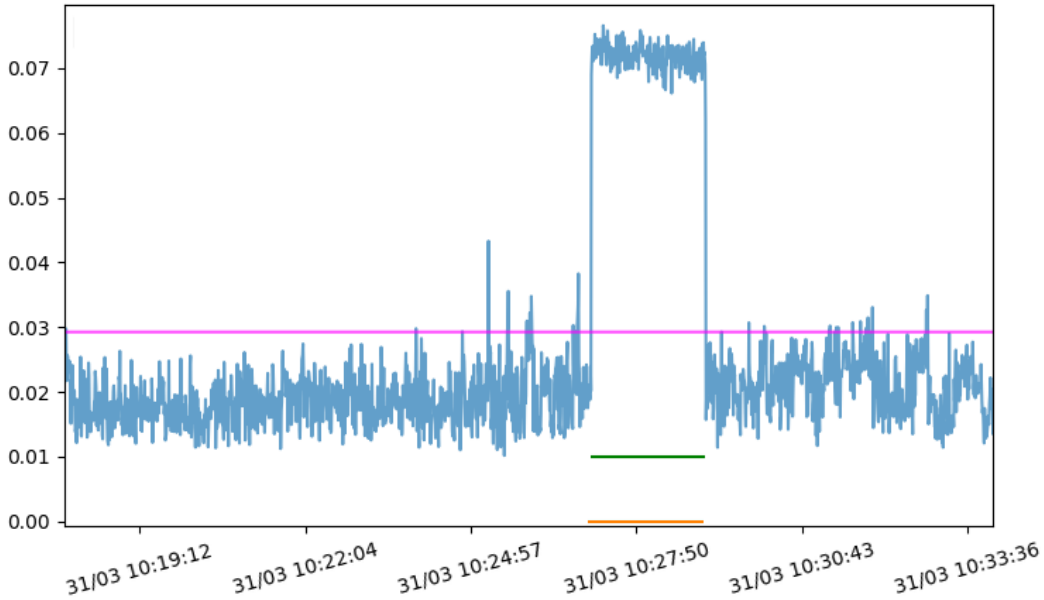

FIGURE 6.6 – Une série temporelle correspondant à la reconstruction (en bleu), le seuil (en rose), l'intervalle réel d'attaque (en orange) et l'intervalle détecté (en vert)

Le localisateur temporel estime le début (t_s) et la fin (t_e) d'une anomalie. La date de début correspond simplement au temps auquel l'erreur de reconstruction dépasse le seuil $\tau(t)$; de manière similaire, la date de fin correspond au temps auquel cette erreur passe sous ce seuil. Cependant, en tant que tel, cette méthode peut générer trop de faux positifs. Par exemple, sur la figure 6.6, l'erreur de reconstruction peut dépasser aléatoirement le seuil. Ainsi, pour avoir un détecteur efficace, nous avons choisi de calculer une erreur cumulative, qui correspond à la somme des erreurs ayant dépassé le seuil. Lorsque l'erreur de reconstruction passe sous le seuil $\tau(t)$, l'erreur cumulative est comparée avec un autre seuil appelé *seuil cumulatif* Sc. Si celui-ci est dépassé, la détection est confirmée, sinon celle-ci est annulée. Cette erreur cumulative permet à notre solution de détecter à la fois des anomalies longues et subtiles ainsi que des anomalies courtes et intenses.

L'algorithme 1 décrit ces étapes de fonctionnement. Il retourne $D_b(p)$, correspondant à l'union de tous les intervalles de détection de la bande b par la sonde p.

6.4.2 Diagnostic fréquentiel

Le diagnostic fréquentiel est basé sur la norme 4 de la distance entre chaque point du spectrogramme reconstruit $\hat{\mathbf{w}}$ et le spectrogramme mesuré $\mathbf{w}$, et ce entre la date de début et de fin de l'anomalie trouvée lors de la phase de diagnostic temporel. Pour estimer la fréquence d'une anomalie, nous faisons la somme des normes 4 de la distance pour chaque fréquence monitorée, soit :

Algorithme 1 : Algorithme de diagnostic temporel

Input : e, Fonction erreur RE (bloquante si aucune donnée) pour la sonde p sur la bande b,
τ seuil d'erreur qui dépend du temps t,
Sc, seuil d'erreur cumulative

Output : $D_b(p)$, l'union des intervalles de détection

Algorithm `TemporalLocalizer(`e, τ, Sc`)`

```
1  t ← FirstDate(e)
2  D_b(p) ← ∅
3  loop                                                    // boucle infinie
4      while e(t) < τ(t) do  t ← t + 1                     // dépassement du seuil
5      cumulative_error ← 0
6      detection ← False
7      t_s ← t
8      while e(t) ≥ τ(t) do  // continue tant qu'au dessus du seuil
9          cumulative_error ← cumulative_error + |e(t) − τ(t)|
10         if ¬detection ∧ cumulative_error ≥ Sc then
11             AdvertiseAlarm() // lève une alerte dès que possible
12             detection ← True
13         t ← t + 1
14     t_e ← t − 1
15     if detection then  D_b(p) ← D_b(p) ∪ {(t_s, t_e)}      // mémorise le
        nouvel intervalle
16 return D_b(p)
```

$$weight(f) = \sum_{t=t_s}^{t_e} d_4(\mathbf{w}(t,f), \hat{\mathbf{w}}(t,f)) \tag{6.5}$$

La fréquence estimée est la fréquence de poids maximum, comme indiqué à l'équation (6.6) :

$$\hat{f} = \arg\max_f weight(f) \tag{6.6}$$

Cet estimateur se concentre sur les anomalies n'ayant lieu que sur une fréquence unique (par exemple une attaque sur 868 MHz), ou sur celles qui impliquent de multiples fréquences proches (comme un arrêt du point d'accès Wi-Fi sur un canal). Une autre approche possible serait d'utiliser le percentile pondéré pour estimer la bande de fréquence d'une anomalie (en calculant, par exemple, la bande de fréquence qui contient 90% de la différence de reconstruction). Nous pensons également qu'un localisateur fréquentiel plus sophistiqué pourrait fournir la liste des fréquences qui contiennent une anomalie en appliquant des méthodes de clustering sur les poids précédemment définis. Cependant, un tel système n'a pas été étudié dans ce manuscrit.

6.5 Diagnostic spatial

Comme présenté dans la section 6.2.2, une de nos principales contributions repose sur notre capacité à fournir des informations de diagnostic pour les experts de sécurité. En parallèle du diagnostic temporel et fréquentiel des anomalies, le diagnostic spatial de l'émetteur associé à une anomalie est une information intéressante pour ces experts. Pour répondre à ce problème de localisation, nous combinons les informations des différentes sondes, en considérant qu'il y a anomalie lorsqu'au moins une sonde l'a détectée. La figure 6.7 récapitule les différentes éléments de la localisation spatiale associés aux précédentes phases de notre approche, présentant ainsi l'architecture et ses étapes de fonctionnement dans sa globalité.

6.5.1 État de l'art sur la localisation

La localisation spatiale a déjà été étudiée dans la littérature. E. Martin et al. [Martin 2010] ont présenté une approche pour fournir une localisation précise en intérieur. Ils ont développé une application qui utilise les capacités d'un téléphone portable (notamment le magnétomètre et l'accéléromètre) pour déterminer la position d'un utilisateur. Cependant, dans notre contexte, sachant que les objets de l'environnement ne peuvent pas être modifiés, il est impossible d'utiliser ce type de technique. D'autres approches se focalisent sur la détermination d'une position à partir de signaux radios et sans modification d'objets. Youssef et al. [Youssef 2007] ont étudié ce type de solutions dans le contexte de systèmes de détection d'intrusions pour les maisons de retraite. Leur travail utilise les informations de la couche

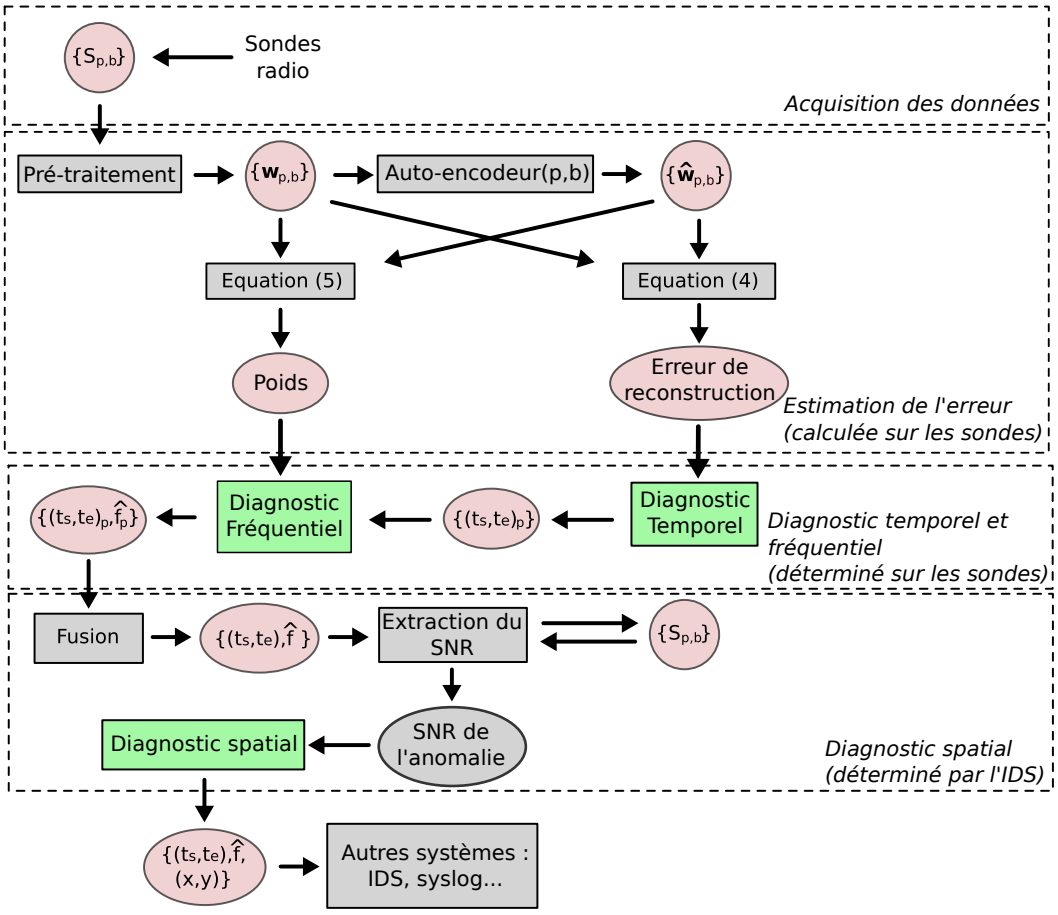

FIGURE 6.7 – Présentation globale de l'approche et détails du bloc de diagnostic spatial

liaison, telles que les *Received Signal Strength Indicators* (RSSI) pour fournir une position approximée des utilisateurs de l'environnement. Cependant, cette information est spécifique à certains protocoles de niveau liaison, et donc à seulement certains récepteurs (ceux qui utilisent le standard 802.11). Pour obtenir une approche qui reste générique, caractéristique essentielle dans le milieu de l'IoT, l'utilisation de ces indicateurs est impossible. En outre, leur travail se concentre sur la position des utilisateurs et non des émetteurs, en proposant cependant des résultats très intéressants.

Dans notre solution, nous voulons étudier l'utilisation d'informations physiques indépendantes des protocoles, i.e. les activités radios, pour déterminer la position approximative d'un émetteur malveillant. En combinant cette information aux informations temporelles et fréquentielles, un expert ou un administrateur devrait être en mesure d'identifier quel émetteur est malveillant ainsi que sa position. À notre connaissance, aucune solution sans modification ou instrumentation d'émetteurs et se basant sur les activités radios n'a été proposée dans la littérature.

6.5.2 Fusion des anomalies

Le diagnostic spatial utilisant les informations provenant de plusieurs sondes, il est tout d'abord nécessaire d'agréger les résultats de chaque localisation venant de ces sondes. Plus formellement, l'agrégation des données du diagnostic temporel D_b est définie par l'union de $D_b(p)$: $D_b = \bigcup_p D_b(p)$. La fusion des informations de fréquence est réalisée à la suite ; c'est simplement la moyenne des fréquences de l'anomalie associée à chaque sonde ayant détectée cette anomalie. Ces informations fusionnées sont ensuite transmises au mécanisme de diagnostic spatial.

6.5.3 Données de calibration et apprentissage

P. Bahl et al. [Bahl 2000] propose une approche pour réaliser une *Active Radio Map*, ou Carte Active Radio, c'est-à-dire une carte radio générée de manière active, via une phase de calibration. Durant cette phase, ils utilisent un objet au sein de l'environnement qui effectue des petites transmissions à différentes positions qui sont monitorées par des sondes mesurant les activités radios. Ils obtiennent ainsi la manière dont les sondes perçoivent les activités en fonction de la position d'un émetteur. Cette carte peut ensuite être utilisée pour déterminer la position d'un individu selon les ondes radios réceptionnées par les sondes. Bien entendu, la position des sondes doit rester la même entre le moment où la calibration est effectuée et le moment où celle-ci est utilisée. Nous avons choisi d'utiliser cette stratégie pour établir une carte active de l'environnement radio pour notre diagnostic spatial.

Les spectrogrammes mesurés par les sondes lors de ces transmissions de calibration seront appelés *données de calibration* dans la suite de ce document. Ils sont utilisés pour établir le modèle de diagnostic spatial qui va faire correspondre la puissance mesurée du signal reçu par chaque sonde aux coordonnées cartésiennes de l'émetteur. Plus précisément, le ratio signal-sur-bruit (SNR) du signal mesuré par chaque sonde est extrait de chaque spectrogramme. Ce SNR est estimé en faisant la différence entre la puissance médiane du signal reçu et le niveau du bruit. Nous choisissons la valeur médiane car celle-ci est plus robuste que d'autres valeurs comme la moyenne aux potentiels changements des activités radios entre la phase de calibration et la phase de détection.

La phase d'apprentissage pourrait être réalisée directement à partir des puissances médianes. Cependant, le choix d'utiliser le SNR n'affecte pas l'efficacité du modèle et permet de faciliter le diagnostic. En effet, les SNR sont plus faciles à interpréter qu'une puissance brute et se révèlent donc utiles dans le cas d'un diagnostic.

La correspondance à établir lors de l'apprentissage correspond à un problème supervisé : les entrées correspondent aux SNR des données de calibration et les sorties sont les coordonnées cartésiennes x, y de l'émetteur. Plus précisément, il s'agit d'un problème de régression et la fonction de coût la plus simple associée à ce problème est la distance euclidienne entre p, la position estimée, et q, la position

attendue.

$$d(p,q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2} \tag{6.7}$$

6.5.4 Modèle de diagnostic spatial

Les horodatages et les fréquences des anomalies sont utilisés pour récupérer auprès de chaque sonde les spectrogrammes bruts monitorés $\{S_{p,b}\}$. Ensuite, si nous définissons (t_s, t_e) les dates d'une anomalie et $\hat{f}$ la fréquence estimée de l'attaque, la puissance médiane d'une sonde p sur la bande de fréquences b (contenant $\hat{f}$) est estimée en extrayant une bande restreinte de δ_f MHz autour de $\hat{f}$ entre t_s et t_e, comme présenté dans l'équation (6.8).

$$MedianPower(p, t_s, t_e, \hat{f}) = \text{median}\{S_{p,b}(t,f) \mid t \in [t_s, t_e], f \in [\hat{f} - \frac{\delta_f}{2}, \hat{f} + \frac{\delta_f}{2}]\} \tag{6.8}$$

La puissance médiane d'une anomalie telle que perçue par chaque sonde est ensuite analysée par l'algorithme des k plus proches voisins (k-nn) pour estimer la position de son émetteur. Dans un premier temps, nous avons expérimenté l'utilisation d'un réseau de neurones, avec des résultats proches de ceux obtenus via k-nn, plus largement étudié dans la littérature. Nous utilisons l'algorithme des k plus proches voisins (k-nn) pour estimer la position de l'émetteur d'une anomalie. Cet algorithme peut être résumé ainsi : pour estimer la position de la source d'une anomalie à partir de son SNR mesuré, nous trouvons les k données de calibration les plus proches (à l'aide de la distance euclidienne) et calculons la moyenne des positions associées à ces données.

6.6 Conclusion

Au sein de ce chapitre, nous avons décrit et présenté les éléments spécifiques pour réaliser du diagnostic à l'aide de notre approche générique. Pour cela, nous avons tout d'abord défini les informations de diagnostic pertinentes à partir des données radios physiques, qui sont : le diagnostic temporel, fréquentiel et spatial. Ensuite, nous avons expliqué comment extraire ces informations des spectrogrammes monitorés par un ensemble de sondes positionnées dans l'environnement. Le prochain chapitre décrit les expérimentations réalisées pour valider cette approche et montrer la pertinence des améliorations de la solution pour le diagnostic.

Chapitre 7

Expérimentations

Sommaire

7.1 Introduction **121**
7.2 Environnement expérimental **122**
7.2.1 Composition de l'environnement et déploiement de l'approche 122
7.2.2 Caractéristiques techniques des composants de l'approche . . 124
7.3 Paramétrage des modèles **124**
7.3.1 Pré-traitement 124
7.3.2 Estimation de l'erreur de reconstruction 124
7.3.3 Diagnostic temporel 125
7.3.4 Diagnostic spatial 125
7.4 Protocole expérimental **125**
7.4.1 Évaluation de la détection d'anomalies 126
7.4.2 Évaluation du diagnostic temporel 126
7.4.3 Évaluation du diagnostic fréquentiel 127
7.4.4 Évaluation du diagnostic spatial 127
7.5 Injection d'attaques réalistes **127**
7.5.1 Attaques injectées 128
7.5.2 Anomalies non prévues 129
7.6 Résultats de l'évaluation **129**
7.6.1 Résultats de la détection d'anomalies 129
7.6.2 Résultats du diagnostic temporel 130
7.6.3 Résultats du diagnostic fréquentiel 132
7.6.4 Résultats du diagnostic spatial 133
7.7 Discussions **134**
7.7.1 Limites globales de l'approche 135
7.7.2 Comparaison expérimentale des environnements étudiés . . . 136
7.7.3 Vie privée 138
7.7.4 Amélioration du diagnostic spatial 139
7.8 Conclusion **140**
7.9 Synthèse de la partie IV **140**

7.1 Introduction

Ce chapitre décrit les expérimentations réalisées pour évaluer le déploiement et l'implémentation de notre solution dans un contexte professionnel. Tout d'abord,

nous présentons l'environnement expérimental mis en place pour évaluer notre approche. Ensuite, nous détaillons les différents paramétrages des modèles utilisés pour réaliser les diagnostics. Le protocole expérimental et les moyens mis en oeuvre pour évaluer ces mécanismes sont détaillés dans la section suivante, en insistant sur l'injection d'attaques représentatives. Finalement, nous listons les résultats d'évaluations associées à chacun des éléments de diagnostic mis en oeuvre, en précisant leurs limites et leur efficacité, avant de discuter ces résultats dans une dernière section.

7.2 Environnement expérimental

Pour évaluer notre solution, nous mettons en place un environnement expérimental professionnel réaliste, c'est-à-dire composé d'un grand nombre d'objets hétérogènes et de nombreux utilisateurs aux comportements distincts. Ensuite, nous injectons des attaques dans l'environnement pour vérifier les capacités de détection et de diagnostic de notre approche déployée.

7.2.1 Composition de l'environnement et déploiement de l'approche

L'environnement expérimental utilisé pour valider l'approche proposée est illustré par la figure 7.1. Il correspond à une salle de travail qui accueille les étudiants de Master durant leur période de stage. C'est une salle relativement occupée et visitée toute l'année, notamment car elle est utilisée pour les pauses cafés. Celle-ci a été équipée d'un grand nombre d'objets connectés représentatifs et pouvant être achetés dans le commerce. La table 7.1 décrit l'ensemble des objets installés, ainsi que leur rôle et la technologie qu'ils utilisent. Les comportements des utilisateurs ne sont pas simulés, ils peuvent ainsi interagir avec les objets connectés ou venir avec leurs propres objets, pour recréer les comportements BYOD. Cependant, pour assurer des activités radios régulières, nous automatisons certaines actions sur tous les objets positionnés (par exemple, la sonnette sonne aléatoirement deux fois par jour). Pour réaliser ces tâches automatisées, nous instrumentons deux HackRF qui rejouent des commandes auprès des différents objets.

Trois sondes sont déployées (numérotées 1, 2 et 3), chacune basée sur deux composants, comme présenté dans la section 3.3 : un HackRF One comme périphérique SDR et un Raspberry Pi 3B pour traiter les *sweeps*. Chaque Raspberry Pi est connecté en Ethernet au serveur implémentant l'IDS. Pour des facilités de développement, nous avons décidé de stocker l'ensemble des spectrogrammes au sein de ce serveur.

Nous monitorons trois bandes de fréquences distinctes : 400-500 MHz et 800-900 MHz, qui sont généralement employées au sein des objets domotiques et des smart-phones, ainsi que 2.4-2.5 GHz, utilisées par un grand nombre de technologies sans-fil (WiFi, Zigbee, etc.) [Al-Fuqaha 2015]. Chaque sonde est équipée d'une antenne sensible à l'une de ces bandes : 400-500 MHz pour la sonde 1, 800-900 MHz

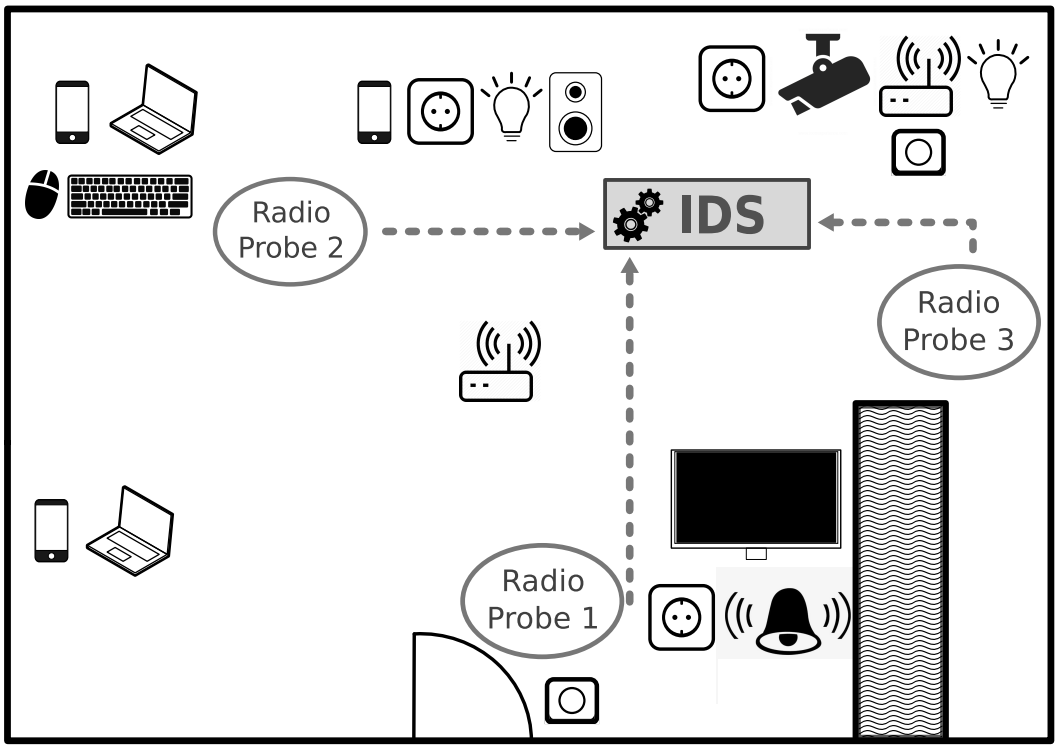

FIGURE 7.1 – Environnement connecté

TABLE 7.1 – Objets connectés de la salle d'expérimentation

Nom	Rôle	Technologie	Qté
D-Link Camera	Vidéo-surveillance	WiFi/2.4–2.5 GHz	1
Phones, Laptops	Téléphonie et Internet	WiFi/2.4–2.5 GHz	5+
Philips Hue	Ampoule	ZigBee/2.4–2.5 GHz	1
Beewi Smartbulb	Ampoule	BLE/2.4–2.5 GHz	1
Keyboard & mouse	Périphériques sans-fil	ESB/2.4–2.5 GHz	2
WiFi Access Points	Point d'accès	WiFi/2.4–2.5 GHz	2
Smart outlet Nityam	Prise électrique	868 MHz	3
Bricelec doorbell	Sonnette	433 MHz	1
HackRF One	Émetteur automatique	433 MHz	1
HackRF One	Émetteur automatique	868 MHz	1
Samsung Smart TV	Télévision connectée	470–800 MHz/2.4–2.5 GHz	1

pour la sonde 2 et 2.4-2.5 GHz pour la sonde 3.

Nous établissons la résolution fréquentielle des sweeps ($1/bw$) à 10 mesures par MHz, ce qui correspond à 3000 mesures pour l'ensemble des trois bandes de 100 MHz. Chaque *sweep* prend 37.5 ms pour scanner toutes les bandes, ce qui correspond à une résolution temporelle $1/T$ de 27 *sweeps* par seconde. Pour rappel, les notations utilisées ici sont présentées dans la section 3.4. Contrairement à notre im-

plémentation pour les domiciles, chaque mesure est sauvegardée sur un octet, pour limiter la taille des données enregistrées. Ainsi, 80 ko de données sont mesurées par chaque sonde chaque seconde. Au total, plus de 2 Go sont collectés chaque jour.

7.2.2 Caractéristiques techniques des composants de l'approche

Dans notre expérimentation, l'IDS s'exécute sur un serveur équipé d'un Intel Core i7-7700 cadencé à 4.2 GHz, 32 Go de RAM et d'une carte graphique Nvidia GTX-1080-Ti. Cependant, nous ne pensons pas qu'il soit nécessaire d'avoir un serveur aussi puissant. En effet, un ordinateur 30 fois plus lent serait suffisant pour traiter les données venant des sondes en temps réel. Dans le cas de notre expérimentation, les données sont traitées hors-ligne mais nous sommes confiants que notre système pourrait fonctionner en ligne puisqu'il nous faut environ 20.407 s pour traiter l'ensemble des spectrogrammes enregistrés, correspondant à 700,815 s de mesure.

7.3 Paramétrage des modèles

7.3.1 Pré-traitement

Dans notre expérimentation, le niveau de bruit est situé à environ -65 dBm et les sondes ne mesurent aucune puissance dépassant les 0 dBm, nous établissons donc le seuil maximum B_u et le seuil minimal B_l à respectivement 0 dBm et -50 dBm.

7.3.2 Estimation de l'erreur de reconstruction

Nous découpons chaque spectrogramme en morceaux dont les dimensions sont 16×1000 : $N = 16$ étant la dimension temporelle (600 ms) et 1000 étant la dimension fréquentielle (une des trois bandes monitorées), c'est-à-dire le nombre de mesures par *sweep*. La dimension temporelle de 600 ms est un compromis : si celle-ci est trop faible, l'auto-encodeur ne sera pas capable d'apprendre de longues activités radios ; si elle est trop grande, le nombre de paramètres à apprendre pour le modèle serait trop élevé et l'apprentissage nécessiterait trop de données. Concernant les hyperparamètres de l'auto-encodeur, nous choisissons les valeurs suivantes :

1. les entrées, composées de 16×1000 attributs ;
2. une couche convolutive à une dimension avec 500 filtres et une fenêtre temporelle de 5, utilisant la fonction d'activation ReLu ;
3. la couche du goulot d'étranglement, une couche dense avec 2000 attributs, employant la fonction d'activation sigmoïde ;
4. une couche dense de sortie, avec 16×1000 sorties, utilisant la fonction d'activation sigmoïde.

Nous utilisons la même architecture pour chaque bande, ainsi les résultats ne sont pas affectés par le comportement spécifique de chaque bande. Ici, la taille

optimale de la couche d'étranglement dépend du trafic sur la bande considérée. En effet, les bandes avec peu de communications (par exemple 800-900 MHz) pourraient employer une couche plus fine pour éviter le surapprentissage ; au contraire, les bandes très utilisées (comme 2.4-2.5 GHz) pourraient nécessiter une couche plus large pour apprendre correctement les comportements radios. Nous avons établi un compromis en définissant 2000 attributs. Ce modèle est implémenté et appris grâce à Keras [Chollet 2015] reposant sur Tensorflow.

Une recherche plus exhaustive des hyperparamètres améliorerait sans aucun doute nos résultats, mais ces travaux ne rentrent pas dans le cadre de notre proposition qui se focalise sur la preuve de concept liée à une approche de diagnostic indépendante de la spécification des protocoles.

Pour la phase d'apprentissage, nous établissons à 80% la superposition des fenêtres glissantes consécutives pour maximiser la quantité de données disponibles. Dans le cadre du jeu de test, celle-ci est de 20%.

7.3.3 Diagnostic temporel

Nous utilisons deux seuils d'erreurs τ : un pour les jours de la semaine entre 8 heures et 19 heures 30, et un pour le reste (week-end et nuits). Le jeu de test comporte un jour férié. Cependant, cette information n'est pas fournie à l'IDS. Nous estimons le seuil d'erreur τ à l'aide du 99th percentile du score des séries temporelles du jeu d'apprentissage pour chaque bande et chaque sonde, et nous estimons T (le seuil d'erreur cumulatif) au 99.995^{th} percentile des erreurs cumulées à l'aide de la fonction d'erreur appliquée au jeu d'apprentissage, et ce pour chaque sonde et chaque bande.

7.3.4 Diagnostic spatial

La calibration est effectuée en envoyant trois signaux à différentes puissances (5 dBm, 20 dBm et 40 dBm) sur la fréquence centrale de chacune des bandes (450 MHz, 850 MHz et 2.45 GHz) à 40 positions distinctes. Ces signaux sont ensuite monitorés par les sondes positionnées dans l'environnement. Nous implémentons l'algorithme des k plus proches voisins pour estimer la position d'une anomalie à partir de ces données de calibration. Nous fixons $k = 6$ et utilisons une distance euclidienne classique entre les SNR. La largeur de la bande autour de la fréquence centrale est fixée à $\delta_f = 0.2\,\text{MHz}$.

7.4 Protocole expérimental

Pour le jeu d'apprentissage, nous réalisons la collecte entre le 19 mars et le 27 mars 2019 (8 jours). Nous collectons deux jeux de tests. Le premier jeu, mesuré entre le 28 mars et le 3 avril, ne contient qu'une seule position d'injection d'attaque, la position F (visible sur la figure 7.2). Le second jeu, mesuré le 7, 8, 13 et 14 mai 2019, contient quant à lui des attaques effectuées à plusieurs positions. Nous réalisons 4

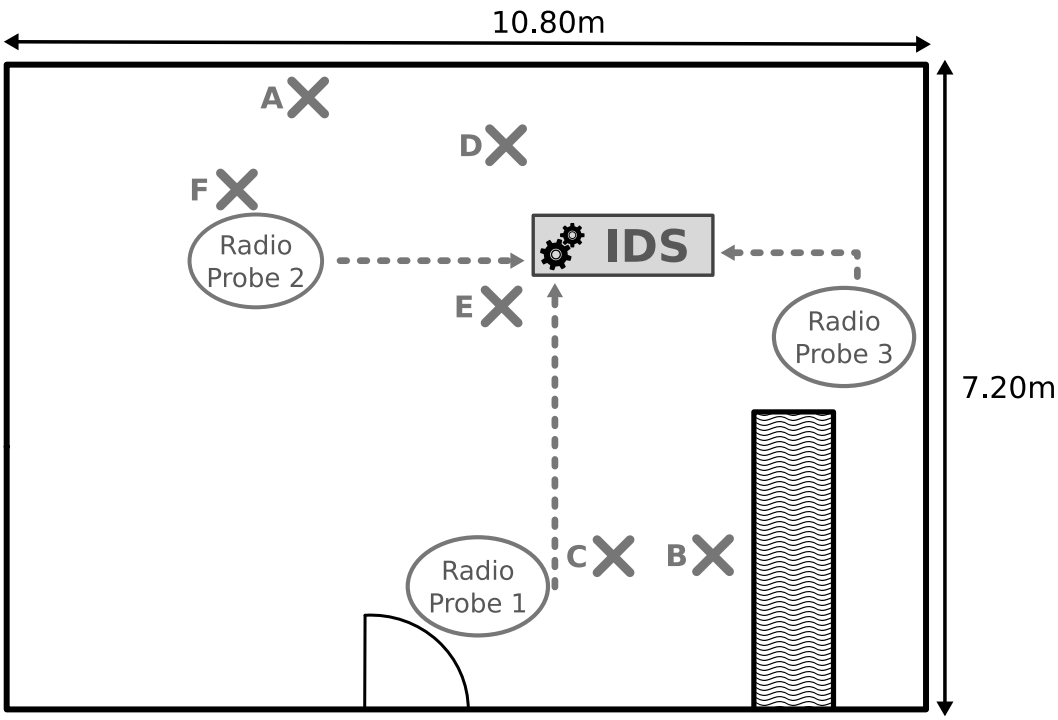

FIGURE 7.2 – Position des attaques dans l'environnement

évaluations distinctes sur ces jeux de données, pour vérifier la pertinence de la détection, puis des différentes phases de diagnostic réalisées.

7.4.1 Évaluation de la détection d'anomalies

Pour évaluer les capacités de détection de notre solution, nous mesurons la proportion d'attaques ayant une intersection avec les intervalles de détection estimés par les sondes sur la même bande. Autrement dit, une attaque est considérée détectée si une alarme est levée durant son exécution.

7.4.2 Évaluation du diagnostic temporel

Pour évaluer l'efficacité du diagnostic temporel, nous étendons à des intervalles les concepts usuels de précision et de rappel définis par [Perry 1955]. Nous notons A_b l'union des intervalles de temps d'attaques sur la bande b et $A_{b,i}$ l'union des intervalles de temps associée à l'attaque i sur la bande b. Ces définitions sont basées sur D_b, l'union des intervalles de temps dérivée des dates de détection des attaques appliqués à la bande b (défini dans la section 6.5.2). Un modèle est établi pour chaque bande, nous évaluons donc chaque modèle en fonction des attaques injectées sur sa bande. Il faut noter que cette méthode d'évaluation peut réduire la précision et le rappel car elle ne prend pas en compte les "détections opportunes" qui peuvent survenir lorsqu'un modèle prédit une attaque injectée sur une autre bande par coïncidence.

Nous définissons la précision par bande P_b et le rappel par attaque R_i ainsi :

$$P_b = \frac{|A_b \cap D_b|}{|D_b|} \tag{7.1}$$

$$R_i = \frac{\sum_{b=1}^{3} |A_{b,i} \cap D_b|}{\sum_{b=1}^{3} |A_{b,i}|} \tag{7.2}$$

Dans notre expérimentation chaque attaque affecte seulement une bande de fréquence, ce qui signifie que pour chaque attaque i, seulement une partie parmi $\{A_{b,i}\}_b$ est non-vide. Il faut noter qu'il est impossible de mesurer la précision par attaque puisque le type d'attaque n'est pas une sortie de notre système. Nous pourrions calculer le rappel par bande, mais nous pensons qu'un taux de rappel par attaque est plus pertinent dans notre évaluation.

7.4.3 Évaluation du diagnostic fréquentiel

Le diagnostic fréquentiel identifie une fréquence à chaque détection. En se basant sur le diagnostic temporel précédent, ce diagnostic souffre inévitablement des erreurs potentielles propagées. Ainsi, pour vérifier les performances de notre diagnostic fréquentiel, nous évaluons seulement les fréquences prédites correspondant à la détection d'une attaque injectée. Plus précisément, un intervalle de détection $[d_s, d_e]$ est considéré valide s'il existe une attaque $[a_s, a_e]$ sur la même bande tel que, si nous notons $[i_s, i_e] = [d_s, d_e] \cap [a_s, a_e]$, les deux conditions suivantes sont vérifiées : $\frac{i_e - i_s}{d_e - d_s} \geq 0.5$ (au moins la moitié de la détection couvre l'attaque) et $\frac{i_e - i_s}{a_e - a_s} \geq 0.5$ (au moins la moitié de l'attaque est détectée).

7.4.4 Évaluation du diagnostic spatial

Pour la même raison que précédemment dans l'évaluation fréquentielle, nous évaluons les positions prédites qui correspondent à une attaque réellement injectée (en utilisant le même critère) avec une fréquence prédite correcte (ayant une erreur inférieure à 1 MHz). La métrique évaluée est la médiane de la distance euclidienne entre la position réelle de l'émetteur d'une attaque et la position prédite par notre IDS.

7.5 Injection d'attaques réalistes

Comme pour la solution appliquée aux environnements particuliers présentée dans la partie précédente, la campagne d'injection d'attaques utilisée dans notre évaluation se déroule en deux étapes. Tout d'abord, il s'agit de définir les attaques qui seront injectées, ainsi que leurs objectifs dans l'évaluation. Ensuite, il s'agit de définir comment sont injectées ces attaques.

7.5.1 Attaques injectées

Les anomalies que nous générons doivent être aussi proches que possible d'activités malveillantes qui pourraient être effectuées dans un environnement connecté. Nous étudions donc un large panel de classes d'attaques qui visent les objets connectés et les environnements IoT. Les attaques choisies ici sont en partie basées sur des attaques réelles qui ont été discutées dans la section 1.2.3.2. En particulier, les attaques 5, 6 et 17 présentées dans le tableau sont tirées respectivement des publications suivantes : [Bachy 2019], [Kamkar 2015] et [Ronen 2018].

Table 7.2 – Classes d'attaques injectées et anomalies non attendues (en orange)

ID	Nom	Technologie	Type	Freq./Bande	#Inj.
			400–500 MHz		
1	scan433-17	433 MHz	Scan 17 dBm	433 MHz	251
2	scan433-10	433 MHz	Scan 10 dBm	433 MHz	11
3	DoS433-27	433 MHz	DoS 27 dBm	433 MHz	50
4	DoS433-40	433 MHz	DoS 40 dBm	433 MHz	16
5	TV-spoofing	DVB-T	Flux DVB-T	485–499 MHz	15
6	bruijn	433 MHz	Injection de Bruijn	433 MHz	15
7	probfail-1	Sonde radio	Arrêt de la sonde	400–500 MHz	×
8	anomaly462	Écran	Rayonnement écran	462 MHz	×
			800–900 MHz		
9	scan868	868 MHz	Scan 20 dBm	868 MHz	276
10	DoS868	868 MHz	DoS 35 dBm	868 MHz	82
11	probfail-2	Sonde radio	Arrêt de la sonde	800–900 MHz	×
			2.4–2.5 GHz		
12	blescan	BLE	Scan	2.4–2.5 GHz	174
13	zigbeescan	Zigbee	Scan	2.4–2.5 GHz	27
14	deauth	WiFi	Deauthentication	2.451–2.473 GHz	33
15	rogueAP	WiFi	PA pirate	2.461–2.483 GHz	83
16	esbinject	ESB	Injection	2.4–2.5 GHz	66
17	injectzigbee	Zigbee	Injection	2.48 GHz	33

Pour injecter ces attaques, nous utilisons un framework nommé Mirage [1] [Cayre 2019], développé au LAAS-CNRS, nous permettant de générer de nombreuses attaques sur différentes technologies à différentes positions (voir figure 7.2). Il fournit également une interface de script nous autorisant à définir des scénarios d'attaques automatiques, pour par exemple générer des comportement malveillants complexes. Celui-ci a été développé dans un objectif d'outil d'assistance au tests de pénétration IoT et implémente un certain nombre d'attaques réalistes sur les objets connectés. À partir de ces travaux, le tableau 7.2 décrit les différentes attaques qui ont été réalisées dans cette expérimentation. Chaque attaque a été effectuée pendant 2 minutes à plusieurs moments et positions. Au total, 1100 attaques ont été

1. https ://redmine.laas.fr/projects/mirage

injectées dans l'environnement.

7.5.2 Anomalies non prévues

En parallèle de nos attaques injectées, trois anomalies inattendues sont apparues durant l'expérimentation. Les anomalies notées probfail-1 et probfail-2 correspondent à des erreurs imprévues sur les sondes 1 et 2. Durant celles-ci, les sondes ne se sont pas arrêtées, mais ont continué à enregistrer du bruit blanc sans aucun signal, sur les bandes 400-500 MHz et 800-900 MHz. Nous aurions pu retirer ces anomalies de jeu de test, mais sachant qu'il s'agit d'une anomalie potentielle qui doit être détectée, nous avons choisi de la conserver.

La dernière anomalie de ce type, notée anomaly462, correspond à une activité observée mais dont l'origine ne semble venir d'aucun des objets installés dans l'environnement expérimental. Nos investigations montrent que cette anomalie a certainement été produite par l'écran d'ordinateur d'un stagiaire présent dans la salle. Nous ne nous attendions pas à un signal aussi fort venant d'un écran [2]. Cet exemple illustre la représentativité de notre environnement expérimental, puisqu'il s'agit d'une anomalie non prévue mais détectable. L'intérêt de fournir des informations de diagnostic à un expert se voit donc confirmer par ce type de comportement, puisque les informations récupérées nous ont permis d'identifier rapidement la provenance et les caractéristiques de l'anomalie.

7.6 Résultats de l'évaluation

Cette section décrit l'ensemble des résultats obtenus associés tout d'abord à la détection, puis aux mécanismes de diagnostic que l'approche fournit.

7.6.1 Résultats de la détection d'anomalies

La proportion d'anomalies détectées est présentée dans le tableau 7.3. La couleur bleue indique la sensitivité d'une sonde à une bande spécifique, liée à l'antenne qui est utilisée par cette sonde, comme expliqué dans la section 7.2. La sonde sensible à une bande spécifique possède généralement un taux de détection plus élevée sur celle-ci. Nous pouvons voir que la sonde 2 détecte la majorité des attaques, ce qui est probablement dû à sa position dans la pièce : comme présenté sur la figure 7.2, celle-ci est proche de plusieurs positions d'attaques.

Trois groupes d'attaques peuvent être identifiés :

— les attaques très bien détectées (toutes les attaques sur 400-500 MHz et 800-900 MHz, excepté scan433-10 et scan868) : au moins 85% des attaques sont détectées ;

— les attaques bien détectées (scan433-10 et scan868) : au moins 60% des attaques sont détectées ;

2. L'anomalie est visible à posteriori en monitorant les puissances reçues entre 400-500 MHz avec un périphérique SDR placé près d'un écran ou d'un ordinateur portable.

— les attaques non détectées (sur 2.4-2.5 GHz) : moins de 1% des attaques sont détectées.

Le faible taux de détection sur les attaques de la bande 2.4-2.5 GHz ne sont pas surprenantes et mettent en avant les limites de notre approche. Les raisons suivantes apportent selon nous des explications à cette faible détection :

— Certaines attaques ne peuvent pas être détectées par une solution qui s'affranchit de la démodulation. Ces attaques sont par exemple WiFi deauthentication, rogueAP et les injections ESB, puisque leurs activités sont très similaires à celles d'activités légitimes.

— La résolution temporelle des sondes basées sur la SDR est trop faible pour détecter certaines activités radios, comme les communications BLE, qui effectuent des sauts de fréquences très rapides qui peuvent être manquées par la sonde. Ce problème est déjà présenté dans la section 5.4.5, et peut-être résolu par des solutions comme SweepSense [Guddeti 2019].

— Les activités radios sur la bande 2.4-2.5 GHz sont de manière générale plus difficiles à apprendre puisqu'elles sont plus nombreuses et que cette bande est surchargée de différents protocoles utilisées dans l'IoT.

Pour résumer, ces résultats illustrent l'efficacité de notre approche et de son implémentation pour les environnements professionnels dans la détection d'attaques survenant sur les bandes 400-500 MHz et 800-900 MHz. Ce résultat est intéressant en soit, puisque les solutions de la littérature ont souvent tendance à proposer des solutions pour les protocoles de la bande 2.4-2.5 GHz, en omettant les protocoles propriétaires souvent présents sur la fréquence 433 MHz ou 868 MHz.

7.6.2 Résultats du diagnostic temporel

Le rappel et la précision du diagnostic temporel sont récapitulés dans les tableaux 7.4 et 7.5. Le rappel est très proche de la proportion d'attaques détectées (présentée dans le tableau 7.3), ce qui signifie que les intervalles temporels de détection couvrent presque en totalité les attaques détectées. La seule exception concerne l'anomalie probfail-2 : puisqu'il n'y a qu'une occurrence de cette anomalie avec un rappel faible, nous en concluons que cette détection est opportune. En effet, l'erreur d'une sonde sur 800-900 MHz est difficile à détecter puisque les sondes ne remontent que du bruit et il n'y a que peu de trafic sur cette bande.

La précision est satisfaisante : entre 77.19% et 96.09% pour la bande 400-500 MHz et entre 92.44% et 97.00% pour la bande 800-900 MHz. Même si la précision sur la bande 2.4-2.5 GHz est faible (moins de 16.00%), la précision globale reste élevée (entre 79.00% et 93.53%) puisque le temps total de détection sur cette bande est plus court que ceux des deux autres bandes. Autrement dit, bien que le détecteur n'est pas efficace sur la bande 2.4-2.5 GHz, celui-ci ne lève que rarement des alarmes comparé à ceux des autres bandes. La forte précision montre que les intervalles du diagnostic temporel ne sont pas trop grandes vis-à-vis des intervalles d'attaques.

Table 7.3 – Anomalies détectées

Nom	Sonde 1	Sonde 2	Sonde 3
400–500 MHz			
scan433-17	52.19%	96.81%	0%
scan433-10	0%	63.64%	0%
DoS433-27	100%	100%	4.00%
DoS433-40	87.50%	87.50%	87.50%
TV-spoofing	86.67%	93.33%	86.67%
bruijn	80.00%	93.33%	86.67%
probfail-1	100%	100%	100%
anomaly462	100%	100%	100%
800–900 MHz			
scan868	65.58%	75.36%	64.86%
DoS868	81.70%	92.68%	63.41%
probfail-2	0%	0%	100%
2.4–2.5 GHz			
ID 12–17	$\leq 1\%$	$\leq 1\%$	$\leq 1\%$

Table 7.4 – Rappel diagnostic temporel

Nom	Sonde 1	Sonde 2	Sonde 3
400–500 MHz			
scan433-17	51.13%	95.91%	0%
scan433-10	0%	62.27%	0%
DoS433-27	99.19%	99.21%	4.00%
DoS433-40	86.75%	87.40%	83.98%
TV-spoofing	81.00%	88.85%	80.83%
bruijn	67.49%	90.39%	73.22%
probfail-1	99.83%	99.88%	9.59%
anomaly462	3.58%	100%	5.45%
800–900 MHz			
scan868	64.74%	73.55%	64.33%
DoS868	80.09%	91.05%	62.39%
probfail-2	0%	0%	3.32%
2.4–2.5 GHz			
ID 12–17	$\leq 1\%$	$\leq 1\%$	$\leq 1\%$

Dans l'ensemble, sur les bandes 400-500 MHz et 800-900 MHz, la précision et le rappel sont élevés. Nous pouvons également conclure que les intervalles de détection estimés par le diagnostic temporel sont très proches des intervalles d'attaques.

Table 7.5 – Précision du diagnostic temporel par bande

Bande	Sonde 1	Sonde 2	Sonde 3
400–500 MHz	96.09%	77.19%	87.77%
800–900 MHz	93.36%	92.44%	97.00%
2.4–2.5 GHz	15.90%	8.03%	5.65%
Toutes bandes	93.53%	79.00%	92.61%

7.6.3 Résultats du diagnostic fréquentiel

Les résultats de l'évaluation du diagnostic fréquentiel sont présentés dans le tableau 7.6. Comme expliqué dans la section 7.4.3, nous sommes seulement intéressés par les attaques ayant été correctement détectées lors du diagnostic temporel. En effet, les cellules avec × correspondent aux attaques n'ayant pas été détectées par ce dernier (cf. Tableau 7.4), et qui ne peuvent donc pas être évaluées.

À part quelques exceptions, l'erreur des fréquences estimées est toujours environ égale ou inférieure à 0.1 MHz. C'est un très bon résultat puisque l'information est suffisamment précise pour aider un expert de sécurité à investiguer, en cas de détection, le type d'objet à l'origine de l'attaque. En outre, la fréquence estimée peut ensuite être utilisée pour extraire le SNR d'une attaque (cf. Section 6.5.4). Les attaques détectées sont donc associées à une fréquence centrale (ou à une bande continue, par exemple de le cas du TV-spoofing), ce qui signifie que nous ne pouvons pas évaluer notre diagnostic fréquentiel sur des attaques s'effectuant sur plusieurs bandes en même temps.

Table 7.6 – Erreur médiane de fréquence

Nom	Fréquence	Sonde 1	Sonde 2	Sonde 3
400–500 MHz				
scan433-17	433 MHz	0.1 MHz	0.1 MHz	×
scan433-10	433 MHz	×	63.2 MHz	×
DoS433-27	433 MHz	0.1 MHz	0.1 MHz	63.4 MHz
DoS433-40	433 MHz	0.1 MHz	0 MHz	0 MHz
TV-spoofing	485–499 MHz	0 MHz	0 MHz	0 MHz
bruijn	433 MHz	0.1 MHz	0.1 MHz	0.1 MHz
800–900 MHz				
anomaly462	462 MHz	×	0.1 MHz	×
scan868	868 MHz	0.1 MHz	0.1 MHz	0.1 MHz
DoS868	868 MHz	0.1 MHz	0.1 MHz	0.2 MHz

Deux fréquences estimées sont imprécises : DoS433-27 sur la sonde 3 et scan433-10 sur la sonde 2. Le rappel de la sonde 3 sur le DoS433-27 est déjà faible, ce qui peut signifier que ces détections sont basées sur la chance, puisque le détecteur diagnostique l'attaque sur un canal DVB-T (495 MHz) et non à 433 MHz. L'estimation imprécise du scan433-10 sur la sonde 2 est plus probablement liée à la faible

puissance à laquelle l'attaque a été émise, ce qui peut également justifier son faible taux de détection.

7.6.4 Résultats du diagnostic spatial

Les résultats du diagnostic spatial sont résumés dans le tableau 7.7. Chaque cellule contient la distance euclidienne médiane pour chaque attaque détectée comme expliqué dans la section 7.4.4. La figure 7.2 montre la position de l'émetteur des attaques injectées.

Table 7.7 – Distance euclidienne médiane pour chaque attaque et position

Nom	Pos. A	Pos. B	Pos. C	Pos. D	Pos. E	Pos. F	Moyenne
scan433-17	-	-	-	-	-	3.64 m	3.64 m
DoS433-27	-	-	-	-	-	4.84 m	4.84 m
DoS433-40	MA	3.11 m	3.72 m	MA	2.68 m	-	3.17 m
TV-spoofing	MA	1.84 m	2.42 m	MA	0.42 m	-	1.56 m
bruijn	MA	2.86 m	3.00 m	MA	×	-	2.93 m
scan868	1.0 m	×	-	×	×	2.68 m	1.84 m
DoS868	1.94 m	3.09 m	3.89 m	3.31 m	×	1.89 m	2.82 m
Moyenne	1.47 m	2.73 m	3.26 m	3.31 m	1.55 m	3.27 m	2.78 m

Dans le tableau, la valeur "-" dans une cellule indique que l'attaque n'a pas été effectuée à cette position durant l'expérimentation, donc aucun résultat ne peut être fourni. Par exemple, les attaques scan433-17 et DoS433-27 n'ont été effectuées que durant la première campagne d'injection et sont donc toutes injectées depuis la position F. Pour l'attaque scan868, cell-ci n'a pas été injectée à la position C. Les cellules avec × représentent les attaques qui n'ont pas été détectées par le diagnostic temporel, et pour lesquelles nous ne pouvons donc pas positionner l'émetteur. La valeur "MA" (pour Multiple Anomalies), représente les attaques pour lesquelles il est impossible de localiser l'émetteur en raison de l'attaque anomaly462 qui a lieu en même temps. En effet, notre algorithme de diagnostic fréquentiel (voir la section 6.4.2) n'estime qu'une fréquence centrale. Ainsi, puisque la fréquence est utilisée pour extraire le SNR de l'attaque, nous ne pouvons le faire lorsque plusieurs attaques sont effectuées au même moment. Ceci représente une des limites de notre approche.

Dans l'ensemble, les résultats sont stables (avec un écart type de 1.04 m). Notre diagnostic spatial nous permet d'identifier la position de l'émetteur d'une attaque détectée avec une précision moyenne de 2.78 m. En considérant la taille de la pièce et les informations temporelles et fréquentielles obtenues par les mécanismes de diagnostic précédent, un expert serait tout à fait en mesure d'identifier rapidement l'objet malveillant ou fautif.

Il faut noter que les positions prédites pour le scan433-17 et le DoS433-27 sont généralement moins précises que les autres. Ceci peut être dû au fait que l'attaque DoS433-27 n'a été testée qu'à partir d'une seule position et que le scan433-17 est

plus difficile à détecter au vue de sa faible puissance d'émission, rendant l'estimation de sa position difficile.

Dans notre expérimentation, la détection est réalisée hors-ligne, ce qui nous oblige à stocker l'ensemble des données de notre jeu de test. Pour limiter la quantité de données, la précision de chaque point des spectrogrammes est stockée sur un seul octet. Dans le cas d'une solution effective en ligne, cette restriction est inutile puisque seulement une courte fenêtre doit être sauvegardée. Cela signifie que le SNR pourrait être plus précis, et dans ce cas, nous nous attendons à obtenir un diagnostic spatial plus précis.

7.7 Discussions

L'expérimentation a été réalisée dans un environnement professionnel complexe et réaliste, qui inclut un grand nombre d'objets différents trouvables dans le commerce et qui utilisent des protocoles de communications hétérogènes. Seulement un faible nombre de comportements radios ont été automatisés, tout en étant basés sur des scénarios réalistes. Les attaques sont toutes basées sur des exemples réels et ont été injectées avec l'outil Mirage, un framework de tests de pénétration pour les réseaux IoT. De manière intéressante, certaines anomalies non prévues ont pu être détectées par notre solution, ce qui confirme l'intérêt de notre approche.

Cependant, dans l'état, seulement des attaques élémentaires ont été injectées. Dans le futur, certains scénarios d'attaques plus complexes combinant de multiples malveillances radios sur des bandes différentes pourraient être implémentées pour évaluer l'efficacité de la détection et du diagnostic. En outre, un plus grand nombre de scénarios automatisées pourraient être testés dans le futur pour évaluer la pertinence de notre approche vis-à-vis d'environnements plus automatisés qui sont de plus en plus représentatifs des environnements IoT.

Concernant le coût de l'approche, celle-ci reste abordable en considérant que le composant principal de notre solution est la sonde radio, qui coûte environ 275 €. Si les modèles sont préalablement appris, un ordinateur à faible puissance serait tout à fait en mesure de réaliser la détection en temps réel. Cependant, dans le cas du diagnostic spatial, il nous faut un nombre non-négligeable de données de calibration qui ne sont pas simples à obtenir de manière automatique. En outre, dans un environnement plus grand, plus de données et plus de temps sont nécessaires pour générer un modèle pertinent. La position des sondes est également un point qui n'a pas été abordé, mais de notre point de vue, ce choix doit être réalisé par un expert de l'environnement, comme c'est le cas pour les systèmes de sécurité physique traditionnels (caméra, alarme, etc.).

L'expérimentation que nous avons menée constitue une preuve de concept préliminaire pour démontrer la pertinence de notre approche. Les résultats pour l'ensemble des mécanismes de détection et de diagnostic sont prometteurs, en particulier dans les bandes de fréquences qui ne sont pas couvertes par les solutions de sécurité traditionnelles de la littérature. La partie nécessitant le plus d'améliora-

tions est celle du diagnostic spatial. Les résultats préliminaires sont encourageants puisqu'ils permettent de fournir des informations intéressantes pour un expert de sécurité qui chercherait à identifier les origines d'une attaque dans un environnement complexe. Finalement, il faut noter que l'approche proposée n'est pas considérée comme une solution tout-en-un, et il pourrait être pertinent de réfléchir à des solutions complémentaires auxquelles l'associer pour détecter des attaques au niveau des couches protocolaires, notamment sur les protocoles connus opérant sur la bande 2.4-2.5 GHz.

7.7.1 Limites globales de l'approche

En choisissant de nous baser uniquement sur les activités radios pour détecter des malveillances, nous avons conscience de limiter les capacités de détection de notre IDS. En effet, avoir la possibilité d'étendre l'analyse de ce dernier en étudiant le contenu du trafic échangé entre les entités d'un réseau permettrait à l'IDS d'être plus précis dans sa levée d'alertes, et donc de minimiser le risque de fausses alarmes. Cependant, comme nous avons pu le constater en première partie de ce document, la réalité commerciale et technique des objets connectés ne donne pas la possibilité de surveiller tous les échanges et ce qu'importent les spécificités des protocoles employés. En nous focalisant sur les activités radios, nous avons pu cependant montrer l'intérêt d'une détection partielle mais précise de malveillances qui modifient le comportement des échanges effectués dans l'espace radio. Cependant, bien que nos expérimentations se veuillent les plus réalistes possible pour simuler des environnements réels et des malveillances existantes, il convient dans le futur d'éprouver cette approche vis-à-vis de comportements plus complexes. Des attaques utilisant certains objets de l'extérieur comme pivot pour corrompre des objets légitimes du réseau interne sont des malveillances qu'il faut être en mesure de détecter dans tous les types d'environnements connectés.

L'objectif de notre approche était de détecter efficacement des anomalies tout en restreignant les capacités d'observation des échanges. Les résultats obtenus, notamment sur les bandes qui ne sont aujourd'hui pas couvertes par les solutions existantes, sont très intéressants. En effet, ces derniers montrent la pertinence de notre approche quant à la détection d'attaques exploitant des protocoles sans-fil propriétaires ou non standardisés. Cependant, les expérimentations révèlent des résultats limités sur des bandes de fréquences surchargées de communications comme celle du WiFi (2.4-2.5 GHz). À cette limite, nous exposons deux hypothèses qui se révèlent à notre avantage. Tout d'abord, bien que ces bandes soient surchargées, la tendance actuelle des industriels du secteur est de proposer des alternatives sur d'autres bandes aux protocoles existants, pour des raisons évidentes d'interférences. Ainsi, dans un futur proche, nous verrons sans aucun doute apparaître des protocoles communiquant de manière plus éparse sur l'ensemble des fréquences de communication disponibles. Il sera donc plus aisé de surveiller des bandes particulières en apprenant le comportement de celles-ci. La seconde hypothèse concerne la large proposition de solutions se concentrant sur les protocoles connus de l'IoT.

Un certain nombre couvre efficacement la détection d'activités malveillantes sur un ou deux protocoles sans chercher à faire une approche générique. Notre solution ayant la capacité de détecter efficacement des attaques sur n'importe quel protocole, notamment ceux propriétaires ou non-standardisés, une combinaison de solutions multi-niveaux serait envisageable. Dans les perspectives que nous présentons à la fin de notre chapitre, nous explorons la complémentarité de notre solution en la combinant avec l'une de ces solutions existantes. Finalement, la radio logicielle étant un domaine relativement récent dans la mise en place de solutions de sécurité, il est tout à fait envisageable de voir apparaître dans la littérature des améliorations de cette technologie.

7.7.2 Comparaison expérimentale des environnements étudiés

En s'attardant et comparant les résultats obtenus dans les précédents chapitres lors de l'évaluation de nos solutions déployées, nous pouvons identifier quelques points sur lesquels il nous semble important de revenir. Bien que les résultats soient très similaires, avec une proportion d'attaques détectées sur les bandes 400-500 MHz et 800-900 MHz très importante, deux résultats de la bande 2.4-2.5 GHz sont étonnants. En effet, lors de l'implémentation de notre solution pour les particuliers, deux attaques injectées sur cette bande sont relativement bien détectées, mais restent complètement indétectables dans notre seconde implémentation :

1. PA pirate (ID 3 et ID 15 pour respectivement l'environnement particulier et professionnel) : très bien détectée dans le premier cas (98.99% de précision et 79.89% de rappel) et quasiment indétectable dans le second (moins de 1% en rappel et précision).
2. Attaques BLE (ID 4 et ID 12) : relativement bien détectée dans le premier cas (98.55% de précision et 57.46% de rappel) et quasiment indétectable dans le second (moins de 1% en rappel et précision).

L'hypothèse principale avancée pouvant expliquer ces résultats est celle d'une différence très importante dans les comportements radios des deux environnements étudiés. En effet, et comme présenté dans chacune des parties les concernant, chaque environnement possède ses propres spécificités. Une de ces spécificités concerne la différence importante du comportement et du nombre d'utilisateurs présents. Nous pensons que l'environnement particulier dans lequel évolue seulement un ou deux utilisateurs, avec relativement moins d'objets communiquants, possède des comportements radios qui sont à la fois plus stables (c'est-à-dire relativement similaires d'un jour à l'autre) et moins dynamiques (de nouvelles communications n'ont pas tendance à disparaître et apparaître régulièrement) qu'un environnement professionnel.

Nous nous proposons donc dans cette sous-section d'étudier, à partir des deux jeux de données récoltés, la différence des comportements radios. Pour cela, nous calculons la variance des puissances des spectrogrammes enregistrés par des sondes de mêmes caractéristiques (antennes) durant une journée sur les deux jeux, et nous

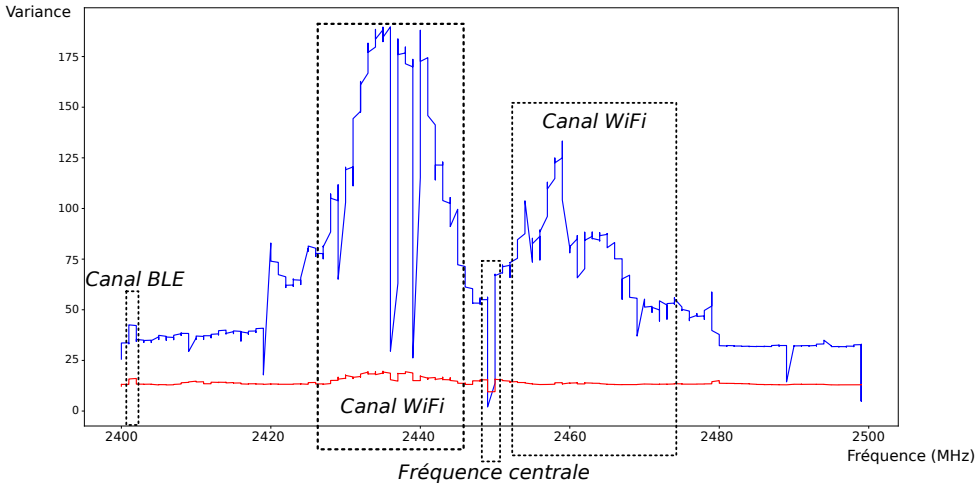

FIGURE 7.3 – Différence de variances des données liées à l'environnement professionnel (bleu) et particulier (rouge)

comparons la différence entre les deux courbes obtenues pour vérifier notre hypothèse. Si celle-ci s'avère correcte, la courbe de variance correspondant à l'environnement professionnel devrait être nettement plus élevée, donc associée à un espace radio plus complexe à modéliser. La figure 7.3 présente ces deux courbes : en bleu la courbe de la variance associée aux spectrogrammes des données provenant de l'environnement professionnel, et en rouge la courbe associée à l'environnement particulier. Visuellement, il est évident que la variance de l'environnement professionnel est beaucoup plus élevée, puisque les comportements et les objets sont plus nombreux. La variance associée aux données provenant du domicile connecté est bien plus faible, mais on retrouve les mêmes augmentations de variance sur le canal BLE et un des deux canaux Wi-Fi. L'environnement professionnel étant notamment composé de plusieurs réseaux Wi-Fi distincts, la variance des puissances reçues sur les fréquences supérieures à 2450 MHz est plus élevée. Cette différence notable de variance valide notre hypothèse, puisque généralement, les communications provenant d'un environnement professionnel sont bien plus hétérogènes, donc bien plus complexes à modéliser et à apprendre.

Concernant les deux attaques détectées dans les domiciles mais non détectées dans un environnement professionnel, l'apparition de communications sur un nouveau canal Wi-Fi (correspondant par exemple à une attaque de type PA pirate) est plus facile à repérer dans le premier cas, puisqu'il n'existe pas ou peu de communications en dehors de celle de l'AP légitime. Concernant les attaques BLE dans le domicile connecté, la variance associée à des communications sur un canal BLE est au même niveau que les communications sur l'ensemble de la bande de fréquence, une modification des puissances reçues sur ce canal sera donc facilement identifiable par notre IDS.

7.7.3 Vie privée

Un point important concerne l'aspect vie privée de notre solution. En effet, celle-ci est amenée à modéliser les comportements radios d'un environnement spécifique, et ce à partir de données recueillies au sein de cet environnement. Ces données peuvent être sensibles, puisqu'elles permettent, au vue des différentes expérimentations effectuées, de modéliser le comportement des objets et des utilisateurs de l'environnement. Un attaquant pourrait recueillir les informations collectées pour en inférer, de manière similaire, certains comportements en vue de malveillances, par exemple en ayant connaissance des heures de présence et d'absence des utilisateurs. Cependant, il est évident que les simples puissances perçues par une sonde d'émissions radios effectuées dans un environnement sont des informations qu'un attaquant proche serait tout à fait en mesure de récupérer lui-même, sans avoir à attaquer le système de sécurité présenté. De plus, il est inconcevable de récupérer le contenu des paquets à partir des spectrogrammes bruts enregistrés par les sondes, puisque les données qu'ils contiennent ne peuvent pas être démodulées. En implémentant des mécanismes de sécurité suffisants dans notre solution pour assurer la confidentialité et l'intégrité des données collectées, tels que le chiffrement de celles-ci, l'attaquant privilégierait une approche moins coûteuse pour obtenir ces informations. En outre, l'échange entre les sondes et l'IDS étant effectué via des liaisons filaires, leur confidentialité est assuré dans la majorité des cas.

Concernant les données recueillies, celles-ci n'identifient jamais directement les utilisateurs, et ne révèlent que succinctement leurs comportements. Le nombre d'utilisateurs dans un environnement n'est par exemple pas une information pouvant être obtenue à partir de ces données. En outre, lors de nos expérimentations, nous avons choisi de sauvegarder l'ensemble des données pour faciliter les traitements et leur analyse. Bien entendu, une implémentation plus industrielle serait tout à fait en mesure de ne sauvegarder qu'une fenêtre temporelle limitée de données, ce qui rendrait plus complexe la récupération de ces données par une entité malveillante. Dans le cas de notre implémentation pour les environnements professionnels, il est encore plus difficile de récupérer ces informations. En effet, dans un déploiement effectif, les spectrogrammes sont traités directement au sein de chaque sonde, qui détectent ou non la présence d'une attaque. Les seules informations échangées entre celles-ci et l'IDS sont celles liées à la détection, qui sont utilisées pour déterminer la position approximative de l'émetteur. Or, ces informations ne sont que le résultat du calcul des SNRs correspondant aux attaques, et non pas les spectrogrammes mesurés par les sondes, qui sont quant à eux traités localement. Il est inconcevable de pouvoir déterminer des informations personnelles à partir des simples SNRs correspondant à une potentielle anomalie.

Cependant, nous identifions deux manières d'implémenter notre approche. La première consiste à externaliser la phase d'apprentissage en dehors de l'environnement. Dans ce cas, une phase de collecte est tout d'abord réalisée, puis les données collectées sont fournies à une entité qui s'occupe de faire l'apprentissage du ou des modèles. Dans ce cas, l'entité externe qui réalise cette phase d'apprentissage doit

respecter les réglementations concernant la vie privée. Cette solution n'est pas celle qui prime dans notre esprit, puisqu'elle impose de fournir ces données potentiellement privées à une entité externe. La seconde solution est plus élégante, puisqu'elle propose de ne pas externaliser la phase d'apprentissage. Au vue des différentes hypothèses et des expérimentations effectuées, il est tout à fait envisageable de fournir un équipement qui réalisera l'apprentissage localement pour l'utilisateur qui souhaite bénéficier de notre solution. En effet, les besoins en terme de puissance de calcul sont loin d'être immenses, et un équipement loué serait tout à fait en mesure de répondre à ces besoins. Dans ce cas, l'équipement étant local à l'environnement, les données ne sont pas externalisées et restent au sein de cet environnement.

7.7.4 Amélioration du diagnostic spatial

Dans la section 6.5.4, il a été défini que nous estimions la position de l'émetteur à l'aide de l'algorithme k-nn. Cependant, après plusieurs itérations sur les méthodes de localisation spatiale employées dans la littérature, nous avons pu améliorer nos résultats en utilisant l'algorithme du centroïde pondéré (*Weighted Centroid Algorithm*) [Blumenthal 2007].

À partir de la position (x_p, y_p) de la sonde p et de la puissance médiane d'une anomalie $MedianPower(p, t_s, t_e, \hat{f})$ mesurée par la sonde p, la position (x_a, y_a) de l'émetteur d'une anomalie est estimée par :

$$\underset{(x_a,y_a)}{\arg\min} \sum_p \Big(MedianPower(p, t_s, t_e, \hat{f}) - s_{min}\Big)^{\lambda} d((x_p, y_p), (x_a, y_a))^2 \tag{7.3}$$

Où d représente la distance Euclidienne, s_{min} représente la puissance seuil de détection et λ est un paramètre libre.

La puissance seuil de détection est établie à $s_{min} = -65$ dBm (le niveau de bruit moyen de l'espace d'expérimentation) et le paramètre libre du modèle à $\lambda = 2$. La largeur de bande autour de la fréquence centrale est fixée à $\delta_f = 0.2\,\text{MHz}$.

7.7.4.1 Résultats mis à jour

Le tableau 7.8 présente les résultats mis à jour après application de cette nouvelle méthode de localisation spatiale. Chaque cellule indique la distance Euclidienne médiane pour chaque classe d'attaques détectée. Les différentes positions des émissions d'attaques sont présentées sur la figure 7.2.

De la même manière que les résultats présentés dans la section 7.7, "-" signifie que l'attaque n'a pas été émise depuis la position associée. "×" correspond aux attaques n'ayant pas été détectées, tandis que "MA", représente les attaques n'ayant pas pu être localisées en raison de la présence de l'anomalie "anomaly462".

Globalement, les estimations se révèlent nettement plus précises que lors de l'utilisation de la méthode initiale, avec plus d'un demi-mètre de précision supplémentaire en moyenne (2.17 m). Les estimations associées à certaines positions sont plus précises car l'algorithme du centroïde pondéré a tendance à mieux estimer la

Table 7.8 – Distance Euclidienne médiane pour chaque classe d'attaques en chaque position

Nom	Pos. A	Pos. B	Pos. C	Pos. D	Pos. E	Pos. F	Moyenne
scan433-17	-	-	-	-	-	3.63 m	3.63 m
DoS433-27	-	-	-	-	-	3.96 m	3.96 m
DoS433-40	MA	0.82 m	0.51 m	MA	1.24 m	-	0.86 m
TV-spoofing	MA	0.45 m	0.35 m	MA	1.92 m	-	0.91 m
bruijn	MA	0.95 m	0.32 m	MA	×	-	0.64 m
scan868	2.45 m	×	-	×	×	2.91 m	2.68 m
DoS868	3.16 m	2.47 m	2.67 m	3.13 m	×	2.19 m	2.72 m
Moyenne	2.81 m	1.17 m	0.96 m	3.13 m	1.58 m	3.17 m	2.17 m

position des émissions proches du barycentre des communications. Il est également à noter que les estimations de certaines classes d'attaques sont maintenant précises à moins d'un mètre, montrant bien l'intérêt de la modification de notre méthode de localisation.

7.8 Conclusion

Dans ce chapitre, nous avons proposé une expérimentation permettant d'évaluer notre solution et ses différents composants de diagnostic pour la détection d'anomalies dans les environnements professionnels. Nous pouvons conclure de celle-ci que notre solution est en mesure de détecter de nombreuses attaques, notamment sur les bandes peu étudiées comme 400-500 MHz et 800-900 MHz. Une telle solution s'avère donc intéressante pour compléter celles existantes qui couvrent déjà les protocoles reposant sur des standards de communication connus.

7.9 Synthèse de la partie IV

Dans l'ensemble de cette IVème partie du manuscrit, nous avons spécifié les éléments d'implémentation spécifiques à l'adaptation de notre solution pour des environnements professionnels. Nous avons identifié que l'interprétabilité des alertes levées par notre approche était un point important dans ce contexte, et que fournir des éléments de diagnostic pouvaient apporter un soutien non négligeable aux experts de sécurité de ces environnements. Nous avons donc développé trois mécanismes de diagnostic : temporel, fréquentiel et spatial. Dans un second temps, nous avons évalué notre solution en la déployant dans un environnement réel composé d'objets hétérogènes et d'un grand nombre d'utilisateurs. Il convient à présent de discuter de l'ensemble de l'approche et d'identifier les limites de son applicabilité en fonction des environnements.

Conclusion

L'Internet des Objets étant un sujet relativement récent, les problématiques de sécurité qui en découlent commencent seulement à être étudiées dans la littérature. Cependant, l'engouement industriel et commercial autour de cette technologie est important, et les environnements auparavant peu complexes en terme de communications radios, tels que les domiciles et les locaux d'entreprises, intègrent désormais un nombre important d'objets connectés soumis à de nombreuses contraintes en matière de sécurité. La relative jeunesse et le manque d'expérience des fabricants impliqués dans la réalisation d'objets sûrs imposent de nouvelles réflexions au sein de la communauté scientifique pour pallier les faiblesses des environnements nouvellement équipés. Une des principales problématiques avancée dans cette thèse est la forte hétérogénéité de la nature des protocoles radios cohabitant dans un même espace, rendant complexe l'analyse exhaustive des communications. Les solutions traditionnelles ne se focalisent que sur un nombre restreint de protocoles, s'affranchissent d'analyser les protocoles non standardisés ou propriétaires. Elles ne sont donc pas suffisantes pour détecter l'ensemble des tentatives d'intrusions. L'objectif de cette thèse était focalisé sur l'étude de la faisabilité et la réalisation d'une solution de sécurité réseau générique, permettant d'effectuer de la détection d'anomalies. La solution proposée est un système de détection d'intrusions (IDS) comportemental, basé sur deux éléments principaux : la radio logicielle (*Software-Defined-Radio* (SDR)) pour écouter les communications au niveau de la couche physique et l'apprentissage automatique dont l'objectif est de définir un modèle des communications légitimes.

7.10 Contributions

Notre première contribution offre un moyen générique de surveiller les communications sans-fil effectuées dans un environnement sans tenir compte des spécifications des protocoles. Pour cela, l'usage de la technique basée sur la radio logicielle ou SDR nous a permis de nous affranchir des caractéristiques de modulation du signal des émetteurs pour ne nous concentrer que sur leurs activités radios physiques. Après avoir présenté notre approche, nous avons vérifié son applicabilité dans des environnements réels de types domiciles ou professionnels. Nous nous sommes focalisés dans un premier temps sur les méthodes de monitoring radio, puis dans un second temps sur les modèles et l'interprétabilité des alarmes levées par notre solution. Ces deux contributions nous ont permis de confirmer l'intérêt d'une telle approche dans des environnements aux caractéristiques différentes. En effet, les résultats avancés pour les deux déploiements de notre solution sur les bandes 400-500 MHz et 800-900 MHz montrent une très bonne détection des attaques effectuées, avec un taux de précision et de rappel proches de 100%. De plus, l'ajout de mécanismes de diagnostic dans notre seconde implémentation apporte des informations non négligeables

lors de la détection, permettant d'assister des utilisateurs avertis dans l'analyse des anomalies détectées. Ces mécanismes de diagnostic se révèlent en effet très précis, notamment concernant le diagnostic fréquentiel et temporel, tout en fournissant une estimation de la position de l'émetteur d'une anomalie. Les résultats obtenus démontrent l'intérêt de fournir une solution générique de détection des anomalies réseaux pouvant survenir sur des protocoles très hétérogènes.

7.11 Perspectives

Pour la suite de nos travaux, nous identifions un certain nombre de perspectives intéressantes sur lesquelles nous souhaiterions travailler.

Un point sur lequel notre approche ne s'est pas focalisée concerne l'étude de la complexité des environnements, telle que ceux présentés dans la section 7.7.2. Nous pourrions envisager de vérifier au préalable la difficulté de modélisation des comportements radios d'un environnement en s'aidant du monitoring réalisé par les sondes. Pour cela, chaque environnement à protéger pourrait être équipé d'une sonde qui récolterait les spectrogrammes sur plusieurs jours, et ces derniers seraient analysés statistiquement pour déterminer par exemple le besoin en couverture, et donc en nombre de sondes, pour modéliser efficacement les comportements. Cette phase s'intégrerait parfaitement à la phase de déploiement dans l'environnement.

Ensuite, la phase de déploiement des sondes présentée dans le chapitre 3 n'a pas été étudiée au cours de nos travaux. Or, cette phase est particulièrement importante pour assurer une couverture exhaustive des environnements dans lesquels la solution cherche à s'intégrer. Pour rappel, celle-ci concerne la détermination du nombre et de la position des sondes dans l'environnement. Dans la suite de nos travaux, nous aimerions nous concentrer sur cette étape en étudiant tout d'abord la couverture d'une sonde en fonction de ses caractéristiques (antennes, configuration) et sa portée de détection d'émissions dans un environnement intérieur bruité et meublé. Bien que nos expérimentations ont cherché à couvrir les environnements en répartissant les sondes équitablement dans l'espace, un positionnement efficace n'est pas trivial. Par exemple, deux sondes étant trop proches ne révèlent que peu d'informations différentes, tandis que deux sondes trop éloignées ne permettent pas de réaliser un diagnostic spatial suffisamment précis.

Les expérimentations réalisées lors de nos travaux nous permettent d'assurer la détection de comportements anormaux sur les bandes habituellement non ou peu couvertes par les solutions traditionnelles. En nous focalisant sur ce résultat, nous pourrions être en mesure de modifier notre approche pour permettre la détection de canaux de fuite radios dans les environnements professionnels. Dans notre seconde expérimentation pour les environnements professionnels, nous avons notamment identifié la présence d'une anomalie non attendue sur la fréquence 462MHz. En diagnostiquant cette anomalie avec l'aide de nos différents modèles de diagnostic, nous avons pu en déterminer rapidement les caractéristiques. En continuant notre audit de l'environnement de test, nous avons repéré des signaux similaires prove-

nant de certains écrans d'ordinateurs portables ou de moniteurs de la salle d'expérimentation. Cette anomalie ainsi que sa détection par notre approche montre les capacités de cette dernière à repérer des anomalies nouvelles exploitant des bandes peu utilisées, qui pourraient correspondre à des canaux de fuite. Les canaux de fuite radios sont des moyens utilisées par des entités malveillantes pour extraire des informations confidentielles depuis l'intérieur d'un environnement protégé en passant par des communications radios. Des exemples tels que *The Thing*[3] laissent penser que ces canaux de fuite ont déjà été utilisés par le passé. De plus, avec la démocratisation des périphériques SDR, il est envisageable d'imaginer une évolution de ces techniques de fuite. En garantissant la couverture exhaustive d'un environnement à l'aide de multiples sondes, notre solution serait tout à fait en mesure de détecter l'apparition de communications radios anormales correspondant à ces canaux de fuites, puis de diagnostiquer approximativement ces émissions.

Il est également envisageable d'intégrer notre solution dans une approche de détection d'intrusions plus globale. En effet, un certain nombre d'attaques non couvertes par notre approche seraient détectables à l'aide de métriques provenant des couches protocolaires supérieures à celles de la couche physique, ou d'une analyse plus fine de cette dernière. Par exemple, notre solution n'étant pas en mesure d'inspecter les différents champs des messages échangés, contenant potentiellement des informations sur la provenance ou la charge utile d'une attaque, il nous est difficile d'identifier précisément la nature d'une attaque, mais également de détecter celles dont les caractéristiques physiques sont similaires à celles du trafic légitime. Ainsi, une approche multi-couches intégrant notre solution permettrait de couvrir efficacement à la fois des protocoles non traditionnels (de type propriétaires ou canaux de fuite) ainsi que les protocoles standardisés largement couverts par les approches existantes. Par exemple, une attaque ne reposant pas sur des modifications de comportements radios, mais émettant des communications normales intégrant une charge utile malveillante serait impossible à repérer par notre approche. En fournissant des moyens d'analyse du contenu des échanges, ces attaques pourraient être détectées.

En conclusion, l'approche générique de détection d'anomalies réseaux proposée nous semble être une solution intéressante pouvant être adaptée à de multiples cas d'applications présents et futurs, non seulement dans le domaine de l'Internet des Objets, mais plus généralement dans la détection d'anomalies radios.

3. `https://en.wikipedia.org/wiki/The_Thing_(listening_device)`

Annexe A

Notations

Table A.1 – Paramètres utilisés au sein du manuscrit avec leurs valeurs expérimentales

Paramètre	Description	Valeur expérimentale		Section
		Partie III	Partie IV	
$\{b\}$	Bande de fréquences	400-500 MHz 800-900 MHz 2.4-2.5 GHz		3.3.1
M	Nombre de bandes	3		3.3.1
w	Nombre d'échantillons de la FFT	8192		3.3.1
bw	*Bin width*	0.2 MHz	0.1 MHz	3.3.1
N	Nombre de balayages consécutifs	100	16	3.3.1
R	Fenêtre temporelle des spectrogrammes	10		4.3.1.2
B_l	Seuil bas du niveau de bruit		-50 dBm	6.3.2
B_u	Seuil haut du niveau de bruit		0 dBm	6.3.2
(no notation)	Taille entrées auto-encodeur		16×1000	6.3.1
(no notation)	Fenêtre apprentissage		80%	6.3.3
(no notation)	Fenêtre de test		20%	6.3.3
δ_f	Bande anomalie		0.2 MHz	6.5.4
k	Nombre de voisins		6	6.5.4

Table A.2 – Notations utilisées dans le manuscrit

Notation	Description	Section
p	Indice de sonde	3.3
S_i	Vecteur d'un balayage sur la fréquence F_i	3.3.1
K_i	Valeurs d'un vecteur S_i	3.3.1
V	Nombre de valeurs par seconde	3.3.1
T	Temps en secondes entre deux balayage	3.3.1
$S_{p,b}$	Spectrogramme	3.3.2
L	Fonction de coût	4.3.1.2
$Erreur(i)$	Erreur calculée pour chaque attribut i	4.3.2
$\mathbf{w}$	Spectrogramme traité	6.3.2
$RMSE(\mathbf{w}, \hat{\mathbf{w}})$	Fonction de coût pour l'apprentissage de l'auto-encodeur	6.3.2
$d_4(a, b)$	Distance norme-4	6.3.3
$RE(\mathbf{w}, \hat{\mathbf{w}})$	Erreur de reconstruction	6.3.3
τ	Seuil de détection	6.4.1
Sc	Seuil d'erreur cumulative	6.4.1
t_s	Début d'une anomalie	6.4.1
t_e	Fin d'une anomalie	6.4.1
$D_b(p)$	Union des intervalles de détection par sonde p sur la bande b	6.4.1
$weight(f)$	Poids de la fréquence f	6.4.2
$\hat{f}$	Fréquence estimée d'une anomalie	6.4.2
D_b	Union des intervalles de détection pour toutes les sondes	6.5.2
$MedianPower(p)$	Puissance médiane pour la sonde p	6.5.4
A_b	Union des intervalles temporels d'attaques	7.4.2
$A_{b,i}$	Union des intervalles temporels pour l'attaque i sur la bande b	7.4.2
P_b	Précision par bande b	7.4.2
R_i	Rappel par bande b	7.4.2

Bibliographie

[Ahmed 2016] Ejaz Ahmed, Ibrar Yaqoob, Abdullah Gani, Muhammad Imran et Mohsen Guizani. *Internet of Things based Smart Environments : State-of-the-art, Taxonomy, and Open Research Challenges.* IEEE Wireless Communications, vol. 23, 10 2016. (Cité en pages 9 et 10.)

[Al-Fuqaha 2015] Ala I. Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari et Moussa Ayyash. *Internet of Things : A Survey on Enabling Technologies, Protocols, and Applications.* IEEE Communications Surveys and Tutorials, vol. 17, no. 4, pages 2347–2376, 2015. (Cité en page 122.)

[Amphoux 1988] Pascal Amphoux. *L'intelligence de l'habitat.* Dans Domotique 88, Paris, France, janvier 1988. (Cité en page 9.)

[Amphoux 1990] Pascal Amphoux. *Domotique domestique.* Culture technique, 1990. (Cité en page 9.)

[Antonakakis 2017] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas et Yi Zhou. *Understanding the Mirai Botnet.* Dans 26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017., pages 1093–1110, 2017. (Cité en pages 26, 28 et 30.)

[Avizienis 2004] Algirdas Avizienis, J.-C Laprie, Brian Randell et Carl Landwehr. *Basic Concepts and Taxonomy of Dependable and Secure Computing.* Dependable and Secure Computing, IEEE Transactions on, vol. 1, pages 11 – 33, 02 2004. (Cité en page 21.)

[Bachy 2019] Yann Bachy, Vincent Nicomette, Mohamed Kaâniche et Eric Alata. *Smart-TV security : risk analysis and experiments on Smart-TV communication channels.* Dans J. Computer Virology and Hacking Techniques, volume 15, pages 61–76, 2019. (Cité en pages 31 et 128.)

[Bahl 2000] Paramvir Bahl et Venkata N. Padmanabhan. *RADAR : An In-Building RF-Based User Location and Tracking System.* Dans Proceedings IEEE INFOCOM 2000, The Conference on Computer Communications, Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, Reaching the Promised Land of Communications, Tel Aviv, Israel, March 26-30, 2000, pages 775–784, 2000. (Cité en page 119.)

[Bellardo 2003] John Bellardo et Stefan Savage. *802.11 Denial-of-service Attacks : Real Vulnerabilities and Practical Solutions.* Dans Proceedings of the 12th Conference on USENIX Security Symposium - Volume 12, SSYM'03, pages 2–2, Berkeley, CA, USA, 2003. USENIX Association. (Cité en page 93.)

[Ben 2017] Seri Ben. *BlueBorne Technical White Paper-1.Pdf*, 2017. (Cité en page 29.)

[Blumenthal 2007] Jan Blumenthal, Ralf Grossmann, Frank Golatowski et Dirk Timmermann. *Weighted centroid localization in zigbee-based sensor networks.* Dans 2007 IEEE international symposium on intelligent signal processing, pages 1–6. IEEE, 2007. (Cité en page 139.)

[Borowiec 2016] Steven Borowiec. *AlphaGo seals 4-1 victory over Go grandmaster Lee Sedol.* The Guardian, vol. 15, 2016. (Cité en page 40.)

[Cauquil 2016] Damien Cauquil. *Btlejuice : The bluetooth smart mitm framework.* DEF CON, vol. 24, pages 4–7, 2016. (Cité en pages 29 et 93.)

[Cayre 2019] Romain Cayre, Jonathan Roux, Eric Alata et Vincent Nicomette. *Mirage : un framework offensif pour l'audit du Bluetooth Low Energy.* SSTIC 2019, 2019. (Cité en pages 29, 94 et 128.)

[Cerf 1989] Vinton G. Cerf et Bob Kahn. *Requirements for Internet Hosts – Communication Layers*, 1989. (Cité en page 14.)

[Chollet 2015] François Chollet*et al.* *Keras.* `https://keras.io`, 2015. (Cité en page 125.)

[Choo 1997] Chun Wei Choo. - *IT2000 : Singapore's Vision of an Intelligent Island.* Dans Peter Droege, editeur, Intelligent Environments, pages 49 – 65. North-Holland, Amsterdam, 1997. (Cité en page 10.)

[Cimpanu 2017] Catalin Cimpanu. *New KRACK Attack Breaks WPA2 WiFi Protocol.* URL : https ://www. bleepingcomputer. com/news/security/new-krack-attack-breaks-wpa2-wifi-protocol, 2017. (Cité en page 30.)

[Cordeiro 2010] Carlos Cordeiro, Dmitry Akhmetov et Minyoung Park. *Ieee 802.11Ad : Introduction and Performance Evaluation of the First Multi-gbps Wifi Technology.* Dans Proceedings of the 2010 ACM International Workshop on mmWave Communications : From Circuits to Networks, mmCom '10, pages 3–8, New York, NY, USA, 2010. ACM. (Cité en page 17.)

[Damien 2018] Aliénor Damien, Marc Fumey, Eric Alata, Mohamed Kaâniche et Vincent Nicomette. *Anomaly based Intrusion Detection for an Avionic Embedded System.* Dans Aerospace Systems and Technology Conference (ASTC-2018), Londres, United Kingdom, novembre 2018. (Cité en page 34.)

[Deswarte 1991] Yves Deswarte, Laurent Blain et Jean charles Fabre. *Intrusion Tolerance in Distributed Computing Systems.* Dans In Proceedings of the IEEE Symposium on Research in Security and Privacy, pages 110–121, 1991. (Cité en page 25.)

[Dhanjani 2015] Nitesh Dhanjani. Abusing the internet of things : blackouts, freakouts, and stakeouts. O'Reilly Media, Inc., 2015. (Cité en page 30.)

[Fouladi 2013] Behrang Fouladi et Sahand Ghanoun. *Security Evaluation of the Z-Wave Wireless Protocol.* Black hat USA, 2013. (Cité en page 31.)

[Francillon 2011] Aurélien Francillon, Boris Danev et Srdjan Capkun. *Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars.* Dans Proceedings of the Network and Distributed System Security Symposium,

NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011, 2011. (Cité en pages 30 et 34.)

[Fuhrer 2006] Patrik Fuhrer et Dominique Guinard. *Building a smart hospital using RFID technologies.* ECEH, vol. 91, pages 131–142, 2006. (Cité en page 11.)

[Gadgets 2017] Great Scott Gadgets. *Hackrf one*, 2017. (Cité en page 45.)

[Ganzha 2017] Maria Ganzha, Marcin Paprzycki, Wiesław Pawłowski, Paweł Szmeja et Katarzyna Wasielewska. *Semantic interoperability in the Internet of Things : An overview from the INTER-IoT perspective.* Journal of Network and Computer Applications, vol. 81, pages 111 – 124, 2017. (Cité en page 11.)

[Guddeti 2019] Yeswanth Guddeti, Raghav Subbaraman, Moein Khazraee, Aaron Schulman et Dinesh Bharadia. *SweepSense : Sensing 5 GHz in 5 Milliseconds with Low-cost Radios.* Dans 16th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2019, Boston, MA, February 26-28, 2019., pages 317–330, 2019. (Cité en pages 100 et 130.)

[Ho 2016] Grant Ho, Derek Leung, Pratyush Mishra, Ashkan Hosseini, Dawn Song et David A. Wagner. *Smart Locks : Lessons for Securing Commodity Internet of Things Devices.* Dans Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2016, Xi'an, China, May 30 - June 3, 2016, pages 461–472, 2016. (Cité en page 52.)

[Hodo 2016] Elike Hodo, Xavier Bellekens, Andrew Hamilton, Pierre-Louis Dubouilh, Ephraim Iorkyase, Christos Tachtatzis et Robert Atkinson. *Threat analysis of IoT networks Using Artificial Neural Network Intrusion Detection System.* Dans ISNCC, 2016. (Cité en page 62.)

[Jasek 2016] Slawomir Jasek. *Gattacking Bluetooth Smart Devices.* Dans Black Hat USA Conference, 2016. (Cité en page 29.)

[Jiang 2017] Jie Jiang, Riccardo Pozza, Kristrún Gunnarsdóttir, Nigel Gilbert et Klaus Moessner. *Using Sensors to Study Home Activities.* Journal of Sensor and Actuator Networks, 2017. (Cité en page 63.)

[Kamkar 2015] Samy Kamkar. *Drive it like you hacked it : new attacks and tools to wirelessly steal cars.* Presentation at DEFCON, vol. 23, 2015. (Cité en pages vii, 30, 34, 65, 94 et 128.)

[Kingma 2015] Diederik P. Kingma et Jimmy Ba. *Adam : A Method for Stochastic Optimization.* Dans 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. (Cité en page 112.)

[Laprie 1996] Jean-Claude Laprie, Jean Arlat, Jean-Paul Blanquart, Alain Costes, Yves Crouzet, Yves Deswarte, Jean-Charles Fabre, Hubert Guillermain, Mohamed Kaâniche, Karama Kanoun, Corinne Mazet, David Powell, Christophe Rabéjac et Pascale Thévenod. Guide de la sûreté de fonctionnement. Cépaduès, Toulouse (France), 1996. OCLC : 35123685. (Cité en page 21.)

[Latvakoski 2014] Juhani Latvakoski, Antti Iivari, Paul Vitic, Bashar Jubeh, Mahdi Ben Alaya, Thierry Monteil, Yoann Lopez, Guillermo Talavera, Javier Gonzalez, Niclas Granqvist, Mounir Kellil, Herve Ganem et Teemu Väisänen. *A Survey on M2M Service Networks.* Computers, vol. 3, 11 2014. (Cité en page 11.)

[Li 2018] Shancang Li, Li Da Xu et Shanshan Zhao. *5G Internet of Things : A survey.* Journal of Industrial Information Integration, vol. 10, pages 1 – 9, 2018. (Cité en page 17.)

[London 2018] Ian London. *Encoding Cyclical Continuous Features - 24-Hour Time*, 2018. (Cité en page 76.)

[Luvison 2009] Bertrand Luvison, Thierry Chateau, Patrick Sayd, Quoc-Cuong Pham et Jean-Thierry Lapresté. *An Unsupervised Learning based Approach for Unexpected Event Detection.* Dans International Conference on Computer Vision Theory and Applications, 2009. (Cité en page 79.)

[Martin 2010] Eladio Martin, Oriol Vinyals, Gerald Friedland et Ruzena Bajcsy. *Precise indoor localization using smart phones.* Dans Proceedings of the 18th International Conference on Multimedia 2010, Firenze, Italy, October 25-29, 2010, pages 787–790, 2010. (Cité en page 117.)

[Masahisa 1968] Miyagi Masahisa. *Frequency-shift-keying phase-modulation code transmission system*, mai 21 1968. US Patent 3,384,822. (Cité en page 44.)

[Meidan 2017] Yair Meidan, Michael Bohadana, Asaf Shabtai, Martín Ochoa, Nils Ole Tippenhauer, Juan David Guarnizo et Yuval Elovici. *Detection of Unauthorized IoT Devices Using Machine Learning Techniques.* CoRR, vol. abs/1709.04647, 2017. (Cité en page 42.)

[Miettinen 2017] Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N. Asokan, Ahmad-Reza Sadeghi et Sasu Tarkoma. *IoT SENTINEL : Automated Device-Type Identification for Security Enforcement in IoT.* 37th IEEE International Conference on Distributed Computing Systems, ICDCS 2017, Atlanta, GA, USA, June 5-8, 2017, pages 2177–2184, 2017. (Cité en page 41.)

[Newlin 2016] Marc Newlin. *MouseJack, KeySniffer and Beyond : Keystroke Sniffing and Injection Vulnerabilities in 2.4GHz Wireless Mice and Keyboards.* Dans DEFCON, 2016. (Cité en pages 30 et 52.)

[Ng 2011] Andrew Ng*et al. Sparse autoencoder*, 2011. (Cité en page 64.)

[Nordic 2007] Nordic. *nRF24L01 Specification - ESB*, 2007. (Cité en page 30.)

[O'Shea 2016] Timothy J. O'Shea, T. Charles Clancy et Robert W. McGwier. *Recurrent Neural Radio Anomaly Detection.* CoRR, vol. abs/1611.00301, 2016. (Cité en page 46.)

[OWASP 2017] OWASP. *OWASP Internet of Things Project - OWASP*, 2017. (Cité en page 28.)

[Perry 1955] James W Perry, Kent Allen et Madeline M Berry. *Machine literature searching x. machine language ; factors underlying its design and development.* American Documentation (pre-1986), vol. 6, no. 4, page 242, 1955. (Cité en pages 85 et 126.)

[Pocero 2017] Lidia Pocero, Dimitrios Amaxilatis, Georgios Mylonas et Ioannis Chatzigiannakis. *Open source IoT meter devices for smart and energy-efficient school buildings.* HardwareX, vol. 1, pages 54 – 67, 2017. (Cité en page 11.)

[Powell 2001] David Powell, Robert Stroud (editors, Sadie Creese (qinetiq, Yves Deswarte (laas cnrs, Klause Kursawe (ibm Zrl, Jean claude Laprie (laas cnrs, David Powell (laas cnrs et James Riordan (ibm Zrl. *Malicious- and accidental-fault tolerance for internet applications : Conceptual model and architecture.* Conceptual model and architecture, 2001. (Cité en page 23.)

[Rajendran 2018] Sreeraj Rajendran, Wannes Meert, Vincent Lenders et Sofie Pollin. *SAIFE : Unsupervised Wireless Spectrum Anomaly Detection with Interpretable Features.* Dans 2018 IEEE International Symposium on Dynamic Spectrum Access Networks, DySPAN 2018, Seoul, Korea (South), October 22-25, 2018, pages 1–9, 2018. (Cité en page 46.)

[Raza 2013] Shahid Raza, Linus Wallgren et Thiemo Voigt. *SVELTE : Real-time intrusion detection in the Internet of Things.* Ad Hoc Networks, vol. 11, no. 8, pages 2661–2674, 2013. (Cité en page 42.)

[Roberts 1970] Lawrence Roberts et Barry D. Wessler. *Computer Network Development to Achieve Resource Sharing.* AFIPS Proceedings, vol. 36, pages 543–549, 01 1970. (Cité en page 13.)

[Ronen 2016] Eyal Ronen et Adi Shamir. *Extended Functionality Attacks on IoT Devices : The Case of Smart Lights.* Dans IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016, pages 3–12, 2016. (Cité en pages 28 et 52.)

[Ronen 2018] Eyal Ronen, Adi Shamir, Achi-Or Weingarten et Colin O'Flynn. *IoT Goes Nuclear : Creating a Zigbee Chain Reaction.* IEEE Security & Privacy, vol. 16, no. 1, pages 54–62, 2018. (Cité en pages 24, 29, 42, 52 et 128.)

[Ross 2016] Ron Ross, Michael McEvilley et Janet Carrier Oren. *Systems Security Engineering : Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems*, 2016. (Cité en page 35.)

[Roux 2017] Jonathan Roux, Eric Alata, Guillaume Auriol, Vincent Nicomette et Mohamed Kaâniche. *Toward an Intrusion Detection Approach for IoT based on Radio Communications Profiling.* Dans 13th European Dependable Computing Conference, page 4p., Geneva, Switzerland, septembre 2017. (Cité en page 3.)

[Roux 2018] Jonathan Roux, Eric Alata, Guillaume Auriol, Mohamed Kaâniche, Vincent Nicomette et Romain Cayre. *RadIoT : Radio Communications Intrusion Detection for IoT - A Protocol Independent Approach.* Dans 17th

IEEE International Symposium on Network Computing and Applications, NCA 2018, Cambridge, MA, USA, November 1-3, 2018, pages 1–8, 2018. (Cité en page 3.)

[Russell 2016] S. Russell et P. Norvig. Artificial intelligence : A modern approach. Always learning. Pearson, 2016. (Cité en page 39.)

[Ryan 2013] Mike Ryan et others. *Bluetooth With Low Energy Comes Low Security.* Dans Usenix, 2013. (Cité en page 29.)

[Siby 2017] Sandra Siby, Rajib Ranjan Maiti et Nils Ole Tippenhauer. *IoTScanner : Detecting and Classifying Privacy Threats in IoT Neighborhoods.* CoRR, vol. abs/1701.05007, 2017. (Cité en page 42.)

[Snoonian 2003] D. Snoonian. *Smart buildings.* IEEE Spectrum, vol. 40, no. 8, pages 18–23, Aug 2003. (Cité en page 10.)

[Sung 2016] Yunsick Sung. *Intelligent Security IT System for Detecting Intruders Based on Received Signal Strength Indicators.* Entropy, vol. 18, no. 10, page 366, 2016. (Cité en pages 41 et 62.)

[Swetina 2014] Jorg Swetina, Guang Lu, Philip Jacobs, Francois Ennesser et Jaeseung Song. *Toward a standardized common M2M service layer platform : Introduction to oneM2M.* IEEE Wireless Communications, vol. 21, no. 3, pages 20 – 26, 2014. (Cité en page 11.)

[Tuttlebee 2003] Walter HW Tuttlebee. Software defined radio : enabling technologies. John Wiley & Sons, 2003. (Cité en page 44.)

[U.S. Department of Homeland Security 2016] U.S. Department of Homeland Security. *Strategic Principles for Securing the Internet of Things (IoT)*, 2016. (Cité en page 35.)

[Youssef 2007] Moustafa Youssef, Matthew Mah et Ashok K. Agrawala. *Challenges : device-free passive localization for wireless environments.* Dans Proceedings of the 13th Annual International Conference on Mobile Computing and Networking, MOBICOM 2007, Montréal, Québec, Canada, September 9-14, 2007, pages 222–229, 2007. (Cité en page 117.)

[Zillner 2015] Tobias Zillner et S. Strobl. *ZigBee Exploited : The Good the Bad and the Ugly.* Dans BlackHat USA, 2015. (Cité en page 29.)

[Zimmermann 1980] H. Zimmermann. *OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection.* IEEE Transactions on Communications, vol. 28, no. 4, pages 425–432, April 1980. (Cité en pages 13, 14 et 18.)

Abstract :

The massive deployment of connected objects, forming the Internet of Things (IoT), is now disrupting traditional network environments. These objects, previously connectivity-free, are now likely to introduce additional vulnerabilities into the environments that integrate them. The literature today paints an unflattering picture of the security of these objects, which are increasingly becoming prime targets for attackers who see them as new exploitable surfaces to penetrate previously secure environments. In addition, the wireless means of communication used by these objects are numerous, with very heterogeneous characteristics at all protocol levels. Particularly in terms of the frequencies used, which make it difficult to analyse and monitor the environments that are equipped with them. These issues, and in particular the strong heterogeneity of these numerous protocols, call into question the traditional solutions used to ensure the security of the exchanges carried out. However, the explosion in the number of these objects requires security architectures that are adapted to these new issues.

In this thesis, we are interested in monitoring and detecting anomalies that may occur in any wireless means of communication used in the IoT. We found a critical lack of solutions with the ability to analyze all exchanges, regardless of the protocol used. To answer this question, we propose a new security architecture based on the monitoring of physical radio signals, making it possible to free oneself from protocol knowledge and therefore to be generic. Its objective is to learn the model of legitimate radio behaviour in an environment using radio probes, then to identify deviations from this model, which may correspond to anomalies or attacks.

The description of this architecture is the first contribution of this thesis. We then studied the applicability of our solution in different contexts, each with its own characteristics. The first study, corresponding to our second contribution, consists in proposing an implementation and deployment of our approach in connected homes. The evaluation of the latter in the face of real attacks injected into radio space and its results show the relevance of our approach in these environments. Finally, the last contribution studies the adaptation and deployment of our generic solution to professional environments where the presence of expert users promotes the integration of advanced diagnostic information to identify the origins of an anomaly. The subsequent evaluation and the results associated with each of the diagnostic mechanisms implemented demonstrate the value of our approach in heterogeneous environments.

Keywords : IoT, Internet-of-Things, Security Architecture, Wireless Networks, Software-Radio, Machine Learning, Security

Résumé :

Le déploiement massif des objets connectés, formant l'Internet des Objets ou IoT, bouleverse aujourd'hui les environnements réseaux traditionnels. Ces objets, auparavant exempts de connectivité, sont désormais susceptibles d'introduire des vulnérabilités supplémentaires dans les environnements qui les intègrent. La littérature dresse aujourd'hui un portrait peu flatteur de la sécurité de ces objets, qui constituent de plus en plus des cibles de choix pour les attaquants, qui y voient de nouvelles surfaces exploitables pour s'introduire dans les environnements auparavant sécurisés. En outre, les moyens de communications non-filaires utilisés par ces objets sont nombreux, avec des caractéristiques très hétérogènes à tous les niveaux protocolaires, notamment en terme de fréquences utilisées, qui rendent complexes l'analyse et la surveillance des environnements qui s'en équipent. Ces problématiques, et notamment l'hétérogénéité forte de ces nombreux protocoles, remettent en question les solutions traditionnelles permettant d'assurer la sécurité des échanges effectués. Or, l'explosion du nombre de ces objets impose d'adapter ou de proposer des architectures de sécurité qui soient adaptées à ces nouvelles problématiques.

Dans cette thèse, nous nous intéressons à la surveillance et à la détection d'anomalies pouvant survenir sur les moyens de communications sans-fil utilisés dans l'IoT, quels qu'ils soient. Nous avons relevé un manque crucial de solutions ayant la capacité d'analyser tous les échanges, et ce, qu'importe le protocole utilisé. Pour y répondre, nous proposons une architecture de sécurité basée sur le *monitoring* des signaux radios physiques, permettant de s'affranchir de la connaissance des protocoles et donc d'être générique. Son objectif est d'apprendre le modèle des comportements radios légitimes d'un environnement à l'aide de sondes radios, puis d'identifier les déviations vis-à-vis de ce modèle, pouvant correspondre à des anomalies ou des attaques.

La description de cette architecture est la première contribution de cette thèse. Nous avons ensuite étudié l'applicabilité de notre solution dans différents contextes, chacun ayant ses caractéristiques propres. La première étude, correspondant à notre deuxième contribution, consiste à proposer une implémentation et un déploiement de notre approche dans les domiciles connectés. L'évaluation de celle-ci face à des attaques réelles injectées dans l'espace radio et ses résultats montrent la pertinence de notre approche dans ces environnements. Finalement, la dernière contribution étudie l'adaptation et le déploiement de notre solution générique à des environnements professionnels où la présence d'utilisateurs experts favorise l'intégration d'informations de diagnostics avancées permettant d'identifier les origines d'une anomalie. L'évaluation qui s'en suit et les résultats associés à chacun des mécanismes de diagnostics implémentés démontrent l'intérêt de notre approche dans des environnements hétérogènes.

Mots clés : IoT, Internet des Objets, Architecture de sécurité, Réseaux sans-fil, Radio logicielle, Apprentissage automatique, Sécurité