PER26 - 2023

SIDE-CHANNEL ATTACKS

Antoine Le Calloch[1] supervisé par Yves Roudier [2]

[1] Polytech Nice Sophia - Université Côte d'Azur. 06410 Biot
[2] Laboratoire d'Informatique, Signaux et Systèmes de Sophia-Antipolis - UNSA - CNRS
antoine.le-calloch@etu.univ-cotedazur.fr
yves.roudier@i3s.unice.fr

Abstract.

L'émergence de l'IoT et des systèmes embarqués soulève de nouvelles préoccupations face à l'apparition de plusieurs vulnérabilités. Parmi elles, les attaques par canaux secondaires, ou Side Channel Attacks (SCAs) représentent une menace croissante. Ces cyberattaques tirent parti des signaux physiques émis par les systèmes informatiques durant leur fonctionnement, tels que la consommation d'énergie, les émissions électromagnétiques, le temps de calcul, les sons, et même les frappes sur un clavier. Elles utilisent les actions physiques et électriques qui se produisent lors de l'exécution des opérations informatiques pour récupérer des indices sur les données secrètes traitées par le système. Elles portent dans la majorité des cas sur l'extraction d'informations relevant de la cryptographie, comme les clés de chiffrement ou des vecteurs d'initialisation.

Ce projet vise à développer un ensemble de démonstrateurs pour illustrer concrètement ces types de SCAs dans un cadre pédagogique. Par démonstrateur, on entend un ensemble d'expérience pour montrer comment une attaque peut être réalisée et détectée. Il peut illustrer, par exemple, la manière dont les émissions électromagnétiques d'un ordinateur lors du traitement de l'information peuvent être capturées et analysées pour en extraire des données sensibles. En plus de la sensibilisation à ces menaces, le projet a pour ambition de fournir un inventaire de matériel de mesure à coût abordable.

En termes de documentation, on attend du projet qu'il fournisse une gamme de démonstrateurs, chacun étant accompagné d'un guide pratique détaillé pour leur mise en place et utilisation. Toutes ces ressources et informations contribueront à renforcer la sécurité des systèmes d'informations face à ce type d'attaque, mais également à évaluer la maturité de la technologie dans ce domaine.

Dans ce document, nous allons dans un premier temps effectuer une analyse de l'existant en présentant les différents types d'attaque par canaux auxiliaires. Pour chaque type d'attaque, nous expliquerons rapidement quelles sont les failles ciblées et comment les exploiter. Dans un deuxième temps nous présenterons plus en détails les 2 attaques sur lesquelles nous nous concentrerons et pour lesquelles nous essaierons de réaliser un exemple concret de l'attaque.

Keywords: IoT (Internet des objets), Attaques par canaux secondaires (SCAs), Sécurité informatique, Vulnérabilités, Cryptographie

1 Analyse de l'existant

1.1 Attaque par injection de faute

Les attaques par injection de faute consistent à créer des erreurs volontairement dans le système cible. Ces événements peuvent alors provoquer des comportements inhabituels dont nous pourront extraire des informations. Cette attaque peut être couplée à d'autres types d'attaques comme l'analyse de la consommation ou les attaques temporelles. Cette attaque peut être semi-invasive, voire non invasive.

Dans le cas d'une attaque semi-invasive l'attaquant doit avoir un accès au silicium. Il doit donc ouvrir le boîtier. Par la suite au moyen d'une lumière focalisée, il pourra injecter des erreurs dans le système. La nécessité pour l'attaquant d'agir sur le système physique limite alors le porté des attaques ce qui en diminue la sévérité. Mais dans le cas d'une attaque non invasive, ce n'est pas le cas.

L'attaque non-invasive ne nécessite pas cette intervention physique, ce qui augmente très largement la portée des attaques. Pour les réaliser, il est alors possible d'agir sur l'horloge ou l'alimentation d'un système à l'aide d'impulsions électriques, on parle alors de Voltage ou clock glitching

Voltage, clock glitching.
Le clock glitching est l'une des méthodes d'injection de fautes les plus anciennes et les plus largement utilisées. Cela nécessite l'insertion de courts glitchs dans l'horloge de l'appareil, ce qui provoque des violations de synchronisation à l'intérieur de l'appareil. Le résultat peut être un périphérique sautant une instruction, modifiant des données ou exécutant une instruction incorrecte.

Le voltage glitching, quant à lui, agit sur la tension de l'appareil pendant l'exécution, provoquant le saut d'instructions, l'utilisation et le stockage de données incorrectes ou la mutation d'instructions. Une des attaques connues est l'attaque BellCoRe [1]. Cette attaque consiste à injecter une faute au moment de la signature de l'algorithme de signature RSA. Le calcul de la signature dans ce type d'algorithme fonctionne ainsi :

- N = pxq, p et q deux grands entiers premiers, e clé publique, d clé privée
- Signature s d'un message m : s = md mod N
- s signature sans faute :
- s = a(sp)× q + b(sq)× p
- s' signature avec une faute dans le calcul de sp devenue sp'
- s'= a(sp')× q + b(sq)× p
- Différence entre s et s': s - s'= a(sp)× q - a(sp')× q
- s - s' est un multiple de q donc q = pgcd(s-s',N), on retrouve ensuite aisément p puis d

1.2 Cryptanalyse acoustique

La cryptanalyse acoustique consiste à analyser les sons émis par du matériel effectuant des opérations cryptographiques comme un crypto-processeur. Avec du matériel adéquat, il est alors possible de récupérer des informations précieuses. Ce type d'attaque date de l'époque des machines cryptographiques, après la Seconde Guerre mondiale, avec l'analyse des sons émis par les touches ou les rotors. Plus récemment, une équipe de chercheurs de l'université de Tel-Aviv [2] ont montré qu'une telle analyse était possible sur des processeurs.

Ultrasons.
Les condensateurs présents dans les ordinateurs sont sujets à des déplacements physiques liés aux champs magnétiques qui leur sont appliqués. Ce déplacement produit des ultrasons d'une certaine fréquence liée aux données qui sont traitées. De ce fait, il est possible à l'aide d'un microphone d'écouter ces sons, de les traiter et d'en extraire des informations telles que des clés d'encryptage ou des mots de passe.

L'équipe de chercheur est parvenue à déchiffrer des données sur plusieurs ordinateurs en analysant les signaux sonores produits par les composants électroniques au niveau du régulateur de tension du processeur. Les tests ont été menés sur l'implémentation RSA 4096 bits de GnuPG. Lors de chaque opération de décryptage, les composants électroniques (condensateurs et bobines) situés dans le régulateur de tension du processeur émettent différents signaux sonores. Situés dans la partie haute du spectre (10 à 150 kHz), ceux-ci sont traduisibles en code assembleur (langage de bas niveau des processeurs). Il suffit donc de les enregistrer au préalable, sachant que l'environnement sonore autour de la machine ne complique que modérément les mesures, car la plupart des bruits (disque dur, ventilateur…) sont concentrés dans la partie basse du spectre.

Le microphone utilisé pour l'expérience est de type parabolique et se révèle efficace jusqu'à une distance de 4 mètres. Il est toutefois possible d'utiliser un simple smartphone, à condition de le placer à quelques dizaines de centimètres de la machine cible, la plage de capture audio étant alors généralement plafonnée à 20 kHz. Sur la plupart des machines cible utilisées, il a été possible de déterminer l'état du processeur (veille ou fonctionnement) et très régulièrement, de déchiffrer toutes les clés RSA après avoir déterminé leur signature acoustique. La faille a alors pris le nom de CVE-2013-4576. Les versions de GnuPG 1.x après 1.4.16 corrigent une partie des vulnérabilités en rendant moins prévisibles les opérations de décryptage.

1.3 Analyse d'émanations électromagnétiques

L'analyse d'émanations électromagnétiques consiste à étudier le rayonnement électromagnétique d'un appareil cible ou d'un circuit électronique pour comprendre son fonctionnement et obtenir des informations secrètes. Cette technique peut être utilisée pour retrouver la clé secrète présente dans un circuit de chiffrement. En effet, les portes logiques, constituant un circuit électronique, dégagent des émissions électromagnétiques lors de chacune de leurs commutations. Or, dans un circuit, toutes les portes logiques ne commutent pas à chaque cycle d'horloge et donc les émanations électromagnétiques seront proportionnelles au nombre de portes logiques commutant. Grâce à l'utilisation de certains algorithmes d'analyse mathématique (Distance de Hamming...), il est possible de retrouver la clé de chiffrement en se basant sur une analyse des variations des émanations électromagnétiques du circuit.

Far-field side-channel attacks.
Cette solution permet une application réelle des attaques par analyse électromagnétique. Le problème du type d'attaque vu précédemment étant la nécessité d'être proche du système cible. Cela peut impliquer d'ôter le boîtier de la cible, actions pouvant être détectée par des protections contre les manipulations. La solution à cela est la récupération du rayonnement électromagnétique que le microcontrôleur a involontairement généré et transmis via l'antenne durant son processus d'envoi de données. On peut ajouter à cela la détection de lancement d'encryptage, décryptage basé sur une analyse de la puissance utilisée. Il est alors possible de récupérer des données sensibles sans nécessairement établir de connexion physique avec l'appareil cible.

L'envoi involontaire des signaux généré par le microcontrôleur est causé par les fabricants de puces qui utilisent des composants microélectroniques toujours plus petits et moins chers. Ils ont adopté une approche de circuits à signaux mixtes, dans laquelle les circuits analogiques et numériques résident sur la même puce de silicium, à proximité physique étroite. Cela entraîne alors de mauvaise séparation des aspects numériques et analogiques et certaines opérations du processeur peuvent fuir vers l'émetteur radio. Plusieurs attaques peuvent ainsi être menées avec un éloignement de plus de 10 mètres. Ce type d'attaque a été étudié, réalisé et documenté par un groupe d'étudiants de Eurecom sous le nom de Screaming Channels [3].

2 Expériences réalisées

2.1 Attaque par analyse d'émanations électromagnétiques

Présentation.
Dans cette première expérience, le but était de reproduire un type d'attaques par canal secondaire à grande distance contre les puces à signaux mixtes utilisées dans les appareils connectés modernes. L'objectif étant de se baser sur le papier académique Screaming Channels [3]

L'activité numérique bruyante d'une architecture à signaux mixtes, combinant logique numérique et analogique sur une même puce se propage facilement aux composants radios sensibles au bruit. Cette propagation implique ainsi des fuites d'informations émises par la radio et potentiellement récupérable par une entité extérieure éloignée. C'est une importante découverte, car précédemment, pour toutes les écoutes de signaux, il était nécessaire d'être très proche du système jusqu'à en démonter les protections. Le système présenté ici évite ainsi toute interaction physique avec l'appareil, il a pris le nom de « canaux hurlants ».

Réalisation.
Pour la réalisation de cette expérience, nous devions disposer d'un certain nombre d'équipement hardware et software. Une configuration minimale pour réaliser l'expérience décrite dans le papier serait la suivante :

- Le système cible : PCA10040 development board
- L'antenne radio: HackRF one
- Le système hôte : Un ordinateur permettant l'analyse des données de l'antenne

Le système cible PCA10040 ne faisait pas partie du matériel en notre disposition durant les semaines de réalisation de ces attaques. Nous sommes donc partis dans un premier temps sur un système cible plus simple, un Arduino Mega 2560. Une analyse d'émanations électromagnétiques sur un Arduino Uno a été réalisé par trois chercheurs à l'université de Dublin [4], nous nous sommes donc appuyés sur ces recherches pour notre expérience.

Pour ce qu'il en est du système permettant la récupération de ces émanations nous avons utilisé une radio logicielle. Une radio logicielle (SDR), une technologie émergente très prisée des chercheurs est des hackeurs, est un système permettant de détecter des ondes radio sur une large bande du spectre de fréquence. Le système est capable de manipuler les signaux de radiofréquence de manière entièrement numérique, éliminant ainsi la nécessité de conversions répétées entre le domaine numérique et analogique. Cette approche permet au logiciel un contrôle direct sur les opérations des composants matériels d'un dispositif, permettant des ajustements précis sans intervention manuelle.

Dans notre cas le système utilisé est le même que celui du papier Screaming Channels [3], un HackRF One. C'est un récepteur-émetteur SDR couvrant les fréquences de 1MHz à 6GHz. Les émanations électromagnétiques d'un système en fonctionnement apparaissent dans une plage de fréquence similaire à celle de l'horloge régissant le tous. La carte Arduino Mega 2560 est une carte de développement dédiée à la construction

d'applications étendues. La carte est basée sur le microcontrôleur ATmega2560 qui fonctionne à une fréquence de 16 MHz. Les fréquences couvertes par le HackRF One descendant jusqu'à 1MHz on aura donc bien accès à la plage de fréquences laissant apparaitre les émanations électromagnétiques de la carte.

Avec ce matériel nous avons réalisé la première expérience. Dans un premier temps nous avons brancher l'Arduino Mega et le HackRF One à notre ordinateur. Nous avons lancer SDRSharp, un logiciel de réception radio logicielle (SDR) populaire et open-source pour les systèmes d'exploitation Windows. Il permet aux utilisateurs de recevoir et de décoder une large gamme de signaux radio en utilisant des périphériques de réception SDR compatibles tels que le HackRF One. Puis nous avons pointer l'antenne du HackRF One vers la carte Arduino comme illustré sur la **Fig. 1**. Les informations captées par le HackRF One et affichées sur SDRSharp nous ont permis de détecter un signal à 16MHz correspondant à l'horloge de l'Arduino visible sur la **Fig. 2**. L'hypothèse fut ensuite confirmée par l'observation d'une perte du signal à la suite d'une coupure de l'alimentation de la carte visible sur la **Fig. 3**.

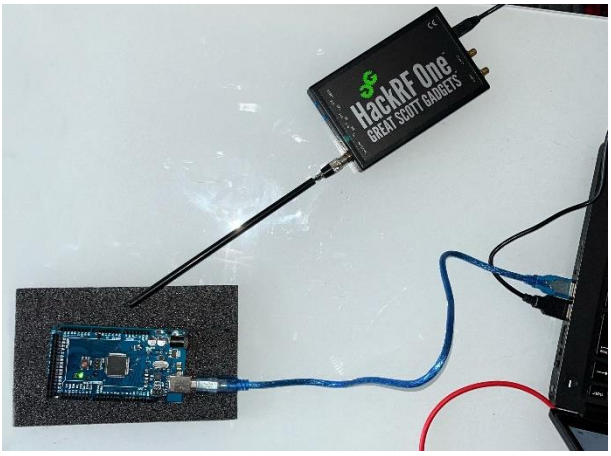

Fig. 1. Configuration expérimentale illustrant le HackRF One pointant vers l'Arduino Mega 2560 afin de capter les émanations électromagnétiques émises par la carte.

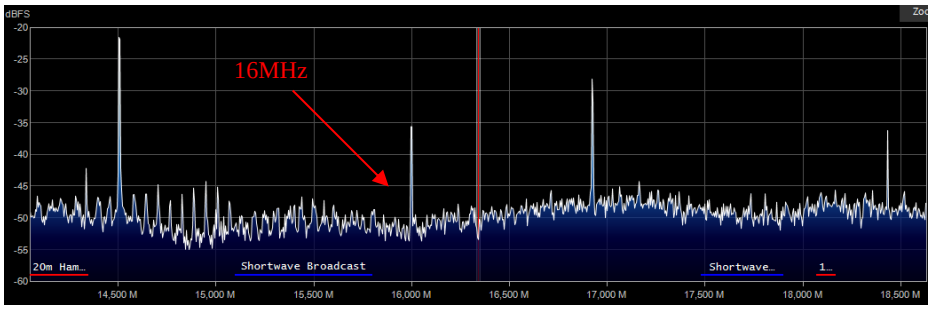

Fig. 2. Spectre de fréquence des émissions EM capté par un HackRF One pointant sur un Arduino Mega 2560 en fonctionnement.

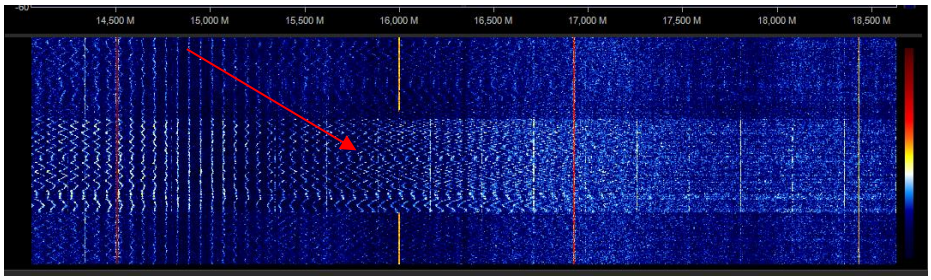

Fig. 3. Spectrogramme des émissions EM capté par un HackRF One pointant sur un Arduino MEGA 2560 qui a été débranché un cours instant.

Or ce signal n'évolue pas dans le temps même à la suite d'un téléversement de code dans la carte depuis l'ordinateur. Si ce signal reste constant qu'importe les actions du processeur, il ne nous permet pas de récupérer les informations sensibles comme le type de code exécuté ou les informations stockées. Nous avons donc recherché dans les fréquences voisines, des émissions affectées par le téléversement. De nombreuses zones de ce type furent trouvées surtout aux alentours des harmoniques de l'horloge. Les harmoniques d'une clock correspondent aux fréquences multiples entières de la fréquence fondamentale de l'horloge, dans notre cas 16MHz, 32MHz, 64MHz... Elles s'expliquent par certaines propriétés des composants électroniques à l'origine des horloges (résonance, commutation …). Nous avons sélectionné une de ces zones, illustrée sur la **Fig. 4**.

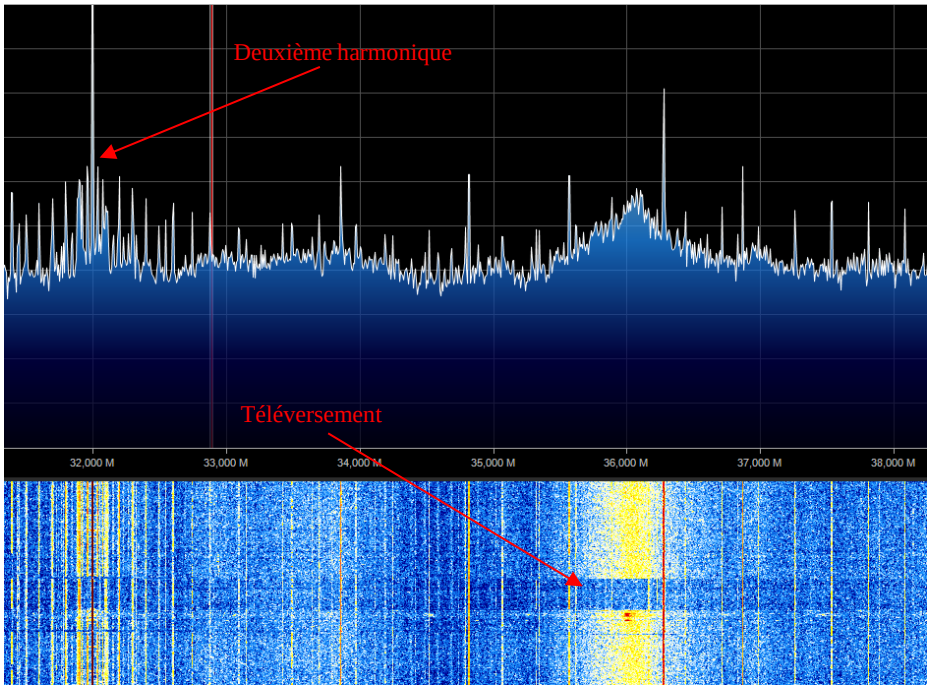

Fig. 4. Spectre de fréquence et spectrogramme permettant de visualiser la deuxième harmonique de l'horloge à 32MHz, et, les émanations de la carte ce coupant au moment du téléversement.

Mais après plusieurs tests avec différentes opérations arithmétiques effectués sur la carte nous n'avons pas réussi à observer de différences significatives dans le signal reçu. Après de nombreuses recherches nous avons finalement compris que ce bruit que l'on reçoit ne venait en réalité pas du processeur mais bien de la connectique entre la carte et l'ordinateur. Elles étaient en fait le résultat des échanges de la carte vers l'ordinateur des « Serial.println » présent dans le code. En effet dès que nous fermions le serial monitor de l'IDE Arduino les émanations électromagnétiques disparaissaient aussi tôt. Il nous fallait donc trouver un signal capté par le hackRF venant de la carte, et réagissant au code s'exécutant sur le processeur. Pour cela nous avons créé un code effectuant 5s d'un calcul arithmétique, puis, 5s de delay. À la suite d'une analyse répétée et minutieuse du spectre de fréquences, un signal de faible intensité mais perceptible avait finis par capter notre attention.

En effet, à la fréquence de 13,889MHz, donc dans les alentours du première harmonique, un signal de faible amplitude semblé disparaitre au téléversement du code et au moment du delay. Après un zoom et de nombreux ajustements on a pu visualiser le signal de manière beaucoup plus clair. La prochaine étape consistait à tenter de distinguer plusieurs opérations arithmétiques entre elles. Nous avons donc mis au point un code exécutant les quatre principales opérations arithmétiques qui sont l'addition, la soustraction, la multiplication et la division. Chaque opération est exécutée en boucle pendant 4s et séparé de 0,1s entre elles. Chaque fois que les 4 opérations s'exécutent un delay de 4s s'ensuit. Le code utilisé pour cette nouvelle étape de l'expérience est le suivant :

```
unsigned long calcul;
unsigned long startTime;
startTime = millis(); // Enregistre le temps actuel
while (millis() - startTime < 4000) { // Boucle tant que le temps écoulé est inférieur à 4s
    calcul = millis() * 2; // Multiplication
}

delay(200); // Delay de 0,2s
startTime = millis();
while (millis() - startTime < 4000) {
    calcul = millis() / 2; // Division
}

delay(200); // Delay de 0,2s
startTime = millis();
while (millis() - startTime < 4000) {
    calcul = millis() + 2; // Addition
}

delay(200); // Delay de 0,2s
startTime = millis();
while (millis() - startTime < 4000) {
    calcul = millis() - 2; // Soustraction
 }
delay(4000); // Delay de 4s
```

A l'exécution du code suivant sur la carte Arduino, nous avons alors pu y voir avec clarté les 4 paternes appartenant à chaque opération arithmétique et nous avons même pu les différencier, comme démontré sur la **Fig. 5**. En effet l'opération de soustraction émet beaucoup moins d'ondes que les autres opérations, alors que l'addition en émet plus. Quant aux deux autres opérations, il est plus difficile de les différencier même si la multiplication semble émettre légèrement moins d'ondes que la division. En réalisant plusieurs fois cette même expérience on peut effectuer le même consta.

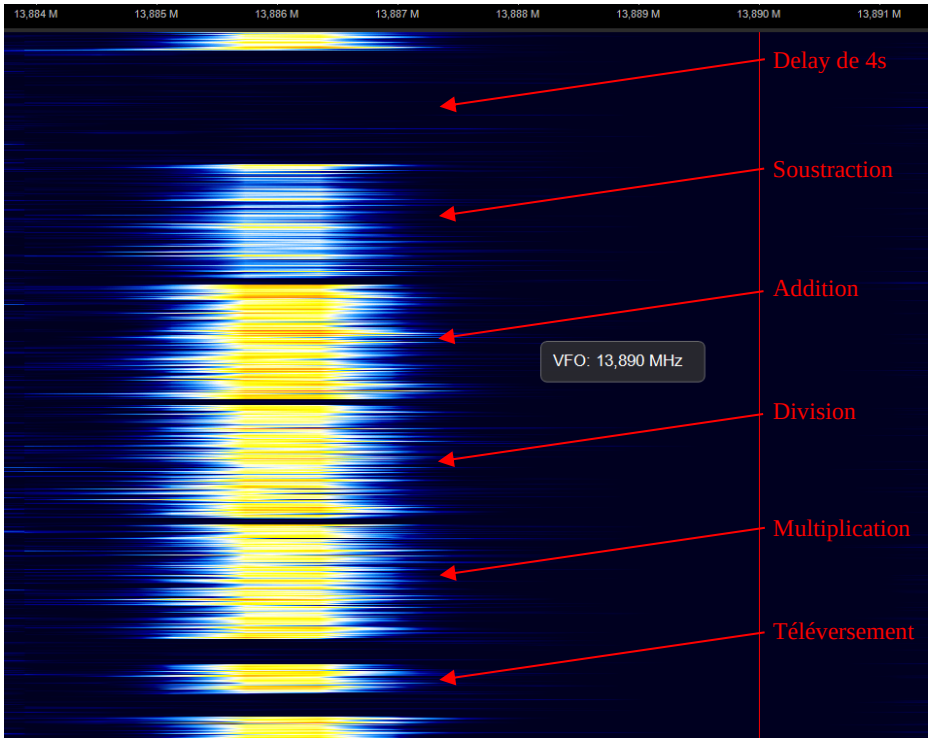

Fig. 5. Spectrogramme des émissions EM capté autour de 13,889MHz par un HackRF One pointant sur un Arduino MEGA 2560 pendant l'exécution du code donnée plus haut.

Conclusion.

Cette information en apparence anodine révèle en réalité une faille de sécurité significative dans le système présenté. En effet, cette expérience nous a permis d'accéder à des informations, auxquelles nous ne sommes pas censées avoir accès, de manière facile et fiable. Bien que nous n'ayons pas eu l'opportunité d'approfondir suffisamment l'expérience pour mettre en pratique une attaque réelle dans son ensemble, les étapes réalisées ont déjà mis en évidence un problème de sécurité préoccupant.

2.2 Attaque par cryptanalyse acoustique

Présentation.
Dans cette seconde expérience, nous allons tenter de reproduire un type d'attaque par canal secondaire utilisant de la cryptanalyse acoustique. Pour réaliser cette attaque nous nous baserons sur le papier académique des trois chercheurs de l'université de Tel-Aviv [2].

Dans ce papier, il est décrit une nouvelle attaque d'extraction de clé de cryptanalyse acoustique, applicable à l'implémentation actuelle de RSA par GnuPG. L'attaque peut extraire l'intégralité des clés de décryptage RSA de 4 096 bits d'ordinateurs portables (de différents modèles), en une heure, en utilisant le son généré par l'ordinateur lors du décryptage de certains textes chiffrés choisis. De notre côté, nous allons nous concentrer sur l'utilisation d'un téléphone portable comme outils d'écoute, cette expérience n'ayant pas été grandement documentée dans le papier. De cette manière, l'attaque deviendrait facile à mettre en œuvre et correspondrait mieux à une attaque réalisable en milieu réel.

Réalisation.
Pour la réalisation de cette expérience, nous devions disposer d'un certain nombre d'équipement hardware et software. Une configuration minimale pour réaliser l'expérience décrite dans le papier serait la suivante :

- Le système cible : Un ordinateur avec un GnuPG d'une version inférieure à 1.4.16
- L'antenne radio: Un téléphone portable équipé d'un micro fonctionnel
- Le système hôte : Un ordinateur permettant l'analyse des données enregistrées par le téléphone

L'expérience effectuée dans la suite de ce papier nécessite comme vu plus haut GnuPG d'une version inférieure à 1.4.16. GnuPG, une abréviation de GNU Privacy Guard, est un logiciel de chiffrement open-source largement utilisé pour protéger la confidentialité des données et des communications. Il fournit des fonctionnalités de chiffrement et de signature numérique, permettant aux utilisateurs de crypter leurs messages et fichiers afin qu'ils ne puissent être lus que par les destinataires autorisés, ainsi que de signer numériquement des documents pour garantir leur authenticité et leur intégrité. L'attaque que nous avons étudiée utilise une fail, dans la façon dont ce logiciel utilise les composants électriques, qui créée des fuites de données. En effet, les versions inférieures à 1.4.16 permettent une analyse et une récupération fiable des clés de signature RSA. Même les version .14 et .15 qui importaient une mitigation des canaux latéraux (ajout de bruit aléatoire…) dans GnuPG a en fait aidé l'attaque, car elle a amplifié l'effet de fuite des informations cryptographiques. Nous avons donc installé la version Windows de GnuPG 1.4.13 pour la réalisation de notre expérience.

Avec ce matériel nous avons réalisé l'expérience suivante. Nous avons créé un fichier volumineux d'une taille de 2G.

fsutil file createnew fichier_a_encrypter.txt 2147483648

Puis nous avons généré une paire de clés du type RSA et RSA de longueur 4096 bits avec comme passe phrase "per"

gpg --gen-key

Nous avons ensuite lancé un enregistrement sonore sur l'iPhone 13 mini. Nous l'avons placé à proximité des ventilateurs de l'ordinateur DELL Latitude 5400 comme sur la **Fig. 6**. Nous avons alors lancé le chiffrement du fichier volumineux créé précédemment.

gpg --encrypt --recipient "Antoine Le Calloch" --output fichier_encrypter.asc --armor fichier_a_encrypter.txt

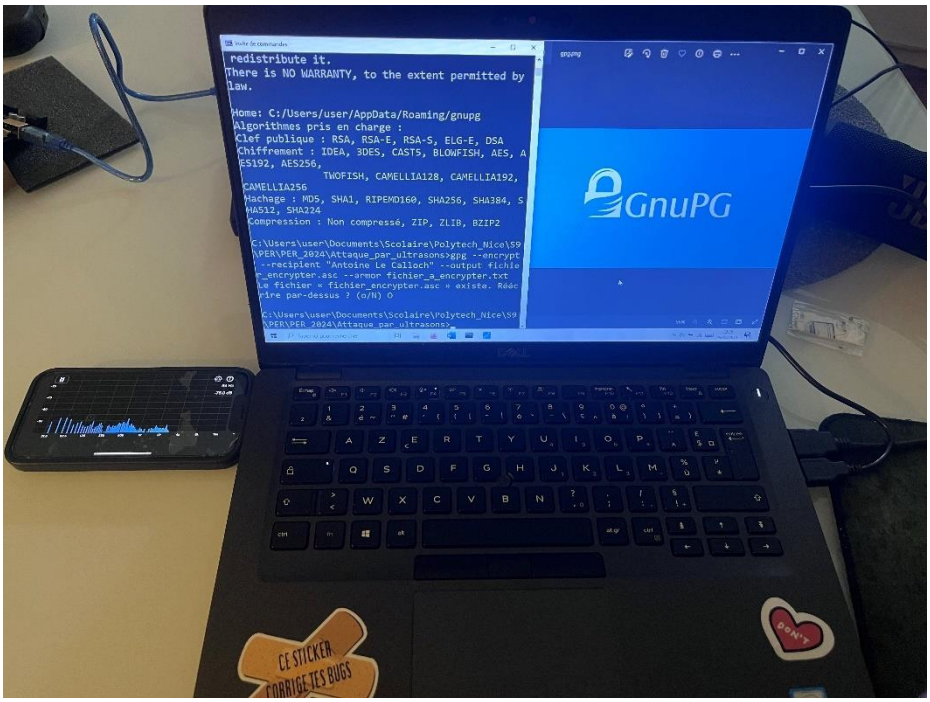

Fig. 6. Configuration expérimentale illustrant l'iPhone 13 mini entrain d'enregistrer, et, placé à proximité des ventilateurs du DELL Latitude 5400, avec GnuPG 1.4.13 qui exécute une opération de chiffrement en fond

Ensuite l'audio enregistré est envoyé sur l'ordinateur, il est ouvert avec Audacity et le spectrogramme est affiché. On peut alors distinguer sur la **Fig. 7** plusieurs sons enregistrés durant le chiffrement du fichier dont certains présents dans le domaine bas des ultrasons. Ces formes qui sont visibles seulement au moment du chiffrement du fichier sont la résultante des vibrations rapides de certains composants électroniques de l'ordinateur en train de réaliser la tâche de chiffrement. Malheureusement comme on peut le voir ici les signaux sont très difficiles à discerner, en effet le bruit ambiant étant très présent, même dans les endroits les plus calmes, et le micro du téléphone n'étant pas

forcément de très bonne qualité il a été très difficile de traiter ces signaux afin d'en sortir de réelles informations. De plus, la majorité des informations étant contenues dans des émanations à des fréquences plus aigües allant jusqu'à 350KHz, avec un micro de téléphone ne dépassant pas les 22KHz, il était impossible d'y accéder.

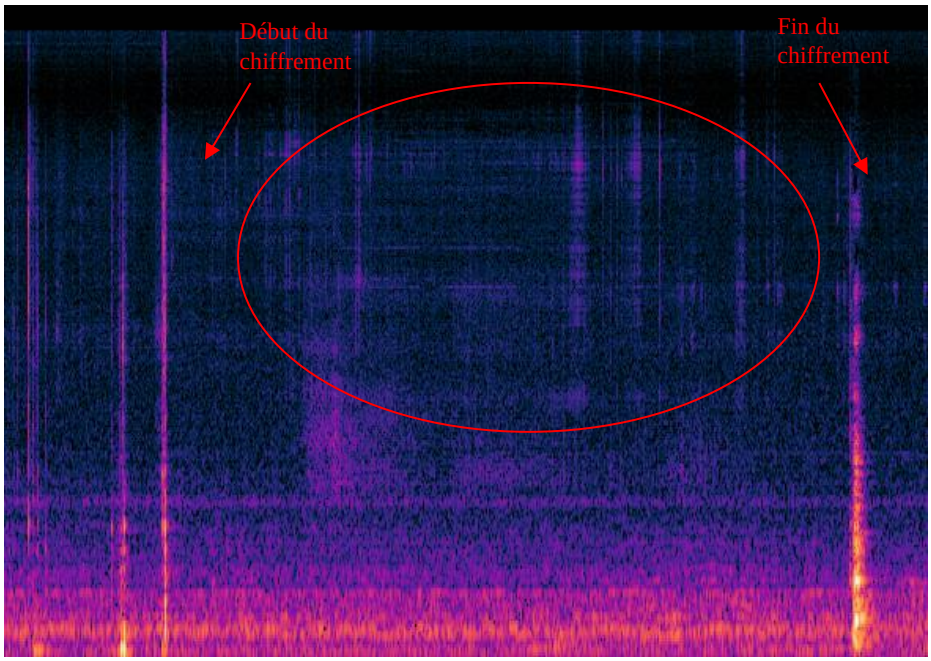

Fig. 7. Spectrogramme allant de 0Hz a 22KHz de l'enregistrement de l'iPhone durant le chiffrement du fichier lourd par GnuPG 1.4.13

Conclusion.

La deuxième expérience s'est révélée moins fructueuse que la première, en raison de divers facteurs qui ont affectés la fiabilité et la clarté des enregistrements. Le choix de se concentrer sur l'utilisation d'un téléphone comme dispositif d'enregistrement, dans le but de simuler au mieux une attaque réelle et de s'adapter aux ressources disponibles, a posé plusieurs défis. La bande passante limitée du micro ainsi que sa qualité moyenne ont constitué des obstacles initiaux. De plus, l'absence d'un environnement à faible bruit pour les enregistrements a entraîné une présence importante de bruits parasites, compliquant l'identification des émissions pertinentes. En outre, l'utilisation d'ordinateurs récents a complexifié l'expérience, car les sons émis par les composants étaient moins prononcés sur ces modèles récents. Toutes ces difficultés nous ont bloqué et freiné dans le traitement des résultats.

3 Discussion

Les expériences menées dans le cadre de cette étude ont mis en lumière plusieurs aspects importants concernant la sécurité des systèmes informatiques et la vulnérabilité potentielle aux attaques par canaux secondaires.

Dans le cas de l'attaque par analyse d'émanations électromagnétiques, notre expérience a démontré la faisabilité de cette méthode pour intercepter des informations sensibles émises par des dispositifs électroniques. En exploitant les émanations électromagnétiques émises par un Arduino Mega 2560 lors de l'exécution de différentes opérations arithmétiques, nous avons pu observer des variations dans le signal capté par un HackRF One. Ces variations nous ont permis de distinguer les différentes opérations, mettant en évidence une faille de sécurité potentielle liée aux émanations électromagnétiques des dispositifs.

Cependant, des limitations ont été rencontrées lors de cette expérience. Notamment, le manque de temps qui nous a empêché de mener à terme cette expérience en récupérant des informations sensibles telles que le type de code exécuté ou les données stockées. De plus, l'interférence provenant de la connectique entre la carte Arduino et l'ordinateur a compliqué l'analyse du signal, nécessitant des ajustements supplémentaires pour isoler les émissions liées au processeur. Les bruits ambiants ont aussi été un défi qui nous a énormément compliqué dans notre recherche des signaux importants.

En ce qui concerne l'attaque par cryptanalyse acoustique, notre expérience a rencontré des difficultés significatives. Malgré nos efforts pour enregistrer les émissions acoustiques générées par un ordinateur lors du chiffrement de données avec GnuPG, la qualité des enregistrements et la présence de bruits parasites ont rendu l'analyse des données difficile. De plus, la bande passante limitée du microphone du téléphone portable utilisé comme dispositif d'enregistrement a limité notre capacité à capturer les fréquences pertinentes.

4 Conclusions

Ces résultats soulignent l'importance de prendre en compte plusieurs facteurs lors de la conception de systèmes sécurisés. Les attaques par canaux secondaires, telles que l'analyse d'émanations électromagnétiques et la cryptanalyse acoustique, mettent en évidence les vulnérabilités potentielles des systèmes électroniques et informatiques. Ce type d’attaque est encore trop peu connu et donc les systèmes en sont rarement protégés. Avec l’émergence de l’IOT et des systèmes embarqués il est primordial de garder à l’esprit ce genre de faille qui peuvent avoir aussi de fort impact. Pourtant même s’il semble facile d’avoir des résultats avec ce genre d’attaque on a vu durant nos expériences qu’elles restent quand même très difficiles à concrétiser. En effet, entre réussir à capter des émanations venant d’un système cible et transformer ces émanations en données utilisables et sensibles il y a un pas énorme. Il faut donc bien se rendre compte que ce genre de pratiques reste très difficile à réaliser entièrement et réservées à des chercheurs passionnés et à des hackers rigoureux.

Références

[1] A. Sidorenko, J. v. d. Berg, R. Foekema, M. Grashuis et J. d. Vos, Bellcore attack in practice, 2012.

[2] D. Genkin, A. Shamir et E. Tromer, RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis, Tel Aviv: Tel Aviv University, 2013.

[3] G. Camurati, S. Poeplau, M. Muench, T. Hayes et A. Francillon, Screaming Channels - Novel side-channel vector in mixed-signal chips with radio transceivers, Sophia Antipolis: EURECOM, 2018.

[4] A. Sayakkara, N.-A. Le-Khac et M. Scanlon, A Survey of Electromagnetic Side-Channel Attacks and Discussion on their Case-Progressing Potential for Digital Forensics, Ireland: University College Dublin, 2019.