GIOVANNI CAMURATI

SECURITY THREATS EMERGING FROM THE INTERACTION BETWEEN DIGITAL ACTIVITY AND RADIO TRANSCEIVERS

Sorbonne Université
EURECOM
Ecole Doctorale ED130
Informatique, Télécommunications et Electronique

SECURITY THREATS EMERGING FROM THE INTERACTION BETWEEN DIGITAL ACTIVITY AND RADIO TRANSCEIVERS

Menaces de sécurité à la frontière entre le bruit électromagnétique et les émetteurs-récepteurs radio

GIOVANNI CAMURATI

Thèse de doctorat de Informatique
Dirigée par Aurélien Francillon
Co-encadrée par Ludovic Apvrille

Présentée et soutenue publiquement le 08/12/2020
Devant un jury composé de :

Prof. Dr. Srđan Čapkun (rapporteur)	ETH Zurich
Dr. Markus Kuhn (rapporteur)	University of Cambridge
Prof. Dr. Aurélien Francillon (directeur de thèse)	EURECOM
Prof. Dr. Ludovic Apvrille (co-encadrant de thèse)	Télécom Paris
Dre. Rabéa Ameur-Boulifa	Télécom Paris
José Lopes Esteves	Agence Nationale de la Sécurité des Systèmes d'Information
Prof. Dr. Raymond Knopp	EURECOM
Prof. Dre. Wenyuan Xu (invité)	Zhejiang University

For Linxia

ABSTRACT

Modern connected devices need both computing and communication capabilities. For example, smartphones carry a multi-core processor, memory, and several radio transceivers on the same platform. Simpler embedded systems often use a mixed-signal chip that contains both a microcontroller and a transceiver. The physical proximity between digital blocks, which are strong sources of electromagnetic noise, and radio transceivers, which are sensitive to such noise, can cause functional and performance problems. Indeed, there exist many noise coupling paths between components on the same platform or silicon die.

In this thesis we explore the security issues that arise from the interaction between digital and radio blocks, and we propose two novel attacks. With *Screaming Channels*, we demonstrate that radio transmitters on mixed-signal chips might broadcast some information about the digital activity of the device, making side channel attacks possible from a large distance. With *Noise-SDR*, we show that attackers can shape arbitrary radio signals from the electromagnetic noise triggered by software execution, to interact with radio receivers, possibly on the same platform.

RÉSUMÉ

Les ordiphones et les objets connectés utilisent des radio pour communiquer avec d'autres appareils électroniques. Ces radio sont placées à côté du processeur et des autres composants numériques. Par exemple, dans les ordiphones un processeur, une mémoire et plusieurs émetteurs-récepteurs radio se trouvent sur la même plateforme. Les systèmes embarquées, plus simples, utilisent souvent des puces à signaux mixtes contenant à la fois un microcontrôleur et un émetteur-récepteur. La proximité physique entre les blocs numériques, qui produisent un bruit électromagnétique très fort, et les émetteurs-récepteurs radio, qui sont sensibles à ce bruit, peut causer des problèmes de fonctionnement et de performance. En effet, il existe de nombreux chemins de couplage entre les composants sur le même système.

Dans cette thèse, nous explorons les problèmes de sécurité qui naissent de l'interaction entre composants numériques et systèmes radio, et nous proposons deux nouvelles attaques. Avec *Screaming Channels*, nous démontrons que les émetteurs radio sur des puces à signaux mixtes peuvent diffuser des informations sur l'activité numérique de l'appareil. Cela permet de mener des attaques par canaux auxiliaires à grande distance. Avec *Noise-SDR*, nous montrons que il est possible de générer des signaux radio arbitraires à partir du bruit électromagnétique déclenché par un logiciel sans privilèges, pour interagir avec des récepteurs radio, éventuellement sur la même plateforme.

PUBLICATIONS

This thesis is based on the research work that led to the publication of two papers about *Screaming Channels* [1, 3], and the submission of a paper about *Noise-SDR* [2]. When appropriate, part of the material conceived for these articles has been adapted and reused in this work. Two other publications, about dynamic firmware analysis [4] and system-on-chip security [5], were the result of a collaboration with other doctoral students, and are not part of this thesis. However, they were important to gain the general understanding of embedded systems security that is a prerequisite to the exploration of novel attacks.

[1] Giovanni Camurati, Aurélien Francillon, and François-Xavier Standaert. "Understanding Screaming Channels: From a Detailed Analysis to Improved Attacks." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2020.3 (2020), pp. 358–401. DOI: 10.13154/tches.v2020.i3.358-401. URL: https://doi.org/10.13154/tches.v2020.i3.358-401.

[2] Giovanni Camurati and Aurélien Francillon. "Noise-SDR: Shaping Arbitrary Radio Signals out of Noise on Modern Smartphones." 2020. Under submission.

[3] Giovanni Camurati, Sebastian Poeplau, Marius Muench, Tom Hayes, and Aurélien Francillon. "Screaming Channels: When Electromagnetic Side Channels Meet Radio Transceivers." In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*. Ed. by David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang. ACM, 2018, pp. 163–177. DOI: 10.1145/3243734.3243802. URL: https://doi.org/10.1145/3243734.3243802.

[4] Nassim Corteggiani, Giovanni Camurati, and Aurélien Francillon. "Inception: System-Wide Security Testing of Real-World Embedded Systems Software." In: *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*. Ed. by William Enck and Adrienne Porter Felt. USENIX Association, 2018, pp. 309–326. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/corteggiani.

[5] Nassim Corteggiani, Giovanni Camurati, Marius Muench, Sebastian Poeplau, and Aurelien Francillon. "SoC Security Evaluation: Reflections on Methodology and Tooling." In: *IEEE Design Test* (2020). DOI: 10.1109/MDAT.2020.3013827.

ACKNOWLEDGMENTS

Dear reader, here I am, reflecting on the results of my research, and sharing them with you. I am deeply fascinated by the topics that I have explored, and I will do my best to spark your interest in them. I hope I will be up to the task, and I thank you for taking the time to read about my work. Now, without waiting any longer, I would like to express my deepest gratitude to those who have been at my side.

Dear Linxia, I dedicate this thesis to you. These few lines are not enough to thank you for being always there with your love. There to share success and failure, excitement and despair. There to encourage me and to inspire me with your example.

Dear family and my dear brother Felice, you helped me become the person I am now. Thank you for your support, and for lighting up the curiosity which brought me to love research.

Dear friends from Turin, Sophia-Antipolis, and all over the World, I always think about each and every of you with great affection. In particular, thank you Paolo, Alberto, Emanuele, Stefano, and Camillo for our long and deep remote discussions during these last years.

Dear Aurélien, thank you for being the best supervisor I could imagine. There are many reasons for which I would like to thank you, like your endless support, your teachings, your ideas, and the freedom and responsibility that you gave me. But above all, thank you for always caring about my well-being and success as a person.

Dear Ludovic, thank you for your precious advice throughout these years and for your support when I was writing this thesis.

Dear François-Xavier, thank you for all what you taught me, and for welcoming me in your research group during my visits in Belgium. They were among the best experiences of these years.

Dear friends from the S3 group, the LabSoc, and EURECOM in general, thank you for being great colleagues and great friends. With you I have always felt part of a team and part of a family. Nassim, Sebastian, Marius, and Tom, a special thanks goes to you for going together through the challenging and rewarding process of publishing a scientific paper. Thank you Giulia, Matteo, and all students for your great work on many projects. Dear friends from Arm, thank you for the beautiful days of my internship, I have learned much from you, and spent a really good time.

Dear teachers and professors, from Nursery School, to Valsalice High School, Politecnico di Torino, EURECOM, and Télécom ParisTech, I would not be here without your teachings and encouragement.

Dear reviewers, attendees to my presentations, and anyone who listened or read about my work, thank you for your invaluable feedback.

Last but not least, I would like to thank all the institutions that made my doctoral studies possible by providing a great environment, enjoyable facilities, instruments, and financial support. My gratitude goes to EURECOM, the SeCiF project within the French-German Academy for the Industry of the future, the DAPCODS/IOTics ANR 2016 project (ANR-16-CE25-0015), the COST action CRYPTACUS, and Google for the Faculty Award assigned to Aurélien Francillon. I also thank the R2Lab at Inria for their support with measurements in their anechoic chamber.

CONTENTS

V CONCLUDING REFLECTIONS

VI APPENDIX

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

CEMA	Correlation-based Electromagnetic Analysis
POIs	Points Of Interest
SSA	Singular Spectrum Analysis
DTW	Dynamic Time Warping
SPA	Simple Power Analysis
DPA	Differential Power Analysis
SNR	Signal-to-Noise Ratio
SoC	System on a Chip

CMOS	Complementary Metal Oxide Semiconductor
DAC	Digital-to-Analog Converter
ADC	Analog-to-Digital Converter
VCO	Voltage Controlled Oscillator
SiP	System in Package (or System in a Package)
EM	Electromagnetic
LNA	Low Noise Amplifier
HDMI	High Definition Multimedia Interface
VGA	Video Graphics Array
FPGA	Field Programmable Gate Array
CRT	Cathode-Ray Tube
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LAN	Local Area Network
WLAN	Wireless Local Area Network
LPWAN	Low-Power Wide Area Network
RF	Radio Frequency
IF	Intermediate Frequency
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
RFI	Radio-Frequency Interference
SDR	Software Defined Radio
ISM	Industrial, Scientific and Medical
DRFC	Direct-to-RF Converter
CRC	Cyclic Redundancy Check
LO	Local Oscillator
CNR	Carrier-to-Noise Ratio
ISM	Industrial, Scientific and Medical
PA	Power Amplifier

SMPA	Switch-Mode Power Amplifier
GFSK	Gaussian Frequency Shift Keying
BFSK	Binary Frequency Shift Keying
AM	Amplitude Modulation
FSK	Frequency Shift Keying
AFSK	Audio Frequency Shift Keying
OOK	On Off Keying
ASK	Amplitude Shift Keying
BASK	Binary Amplitude Shift Keying
PSK	Phase Shift Keying
BPSK	Binary Phase Shift Keying
OFDM	Orthogonal Frequency Division Multiplexing
BFSK	Binary Frequency Shift Keying
MFSK	M-ary Frequency Shift Keying
DTMF	Dual-Tone Multi-Frequency
QPSK	Quadrature Phase Shift Keying
QAM	Quadrature Amplitude Modulation
AM	Amplitude Modulation
FM	Frequency Modulation
NBFM	Narrow-Band Frequency Modulation
PWM	Pulse Width Modulation
ΔΣ	Delta-Sigma
RF-PWM	Radio-Frequency Pulse-Width Modulation
BOC	Binary Offset Carrier
USB	Upper Side Band
LPI	Low Probability of Interception
RTTY	Radio Teletype
SSTV	Slow Scan Tele Vision
DRM	Digital Radio Mondiale

SSB	Single Side Band
NFC	Near Field Communication
RFID	Radio Frequency Identification
WSPR	Weak Signal Propagation Reporter
ESB	Enhanced ShockBurst
FHSS	Frequency Hopping Spread Spectrum
DSSS	Direct Sequence Spread Spectrum
CSS	Chirp Spread Spectrum
FFT	Fast Fourier Transform
LS	Least Square
LMS	Least Mean Square
STA	Spectral Temporal Averaging
LC	Linear Interpolation with the Comb Pilots
CSI	Channel State Information
EVM	Error Vector Magnitude
CDF	Cumulative Density Function
CCDF	Complementary Cumulative Density Function
PAR	Peak-to-Average Power Ratio
WPA	WiFi Protected Access
BLE	Bluetooth Low Energy
LESC	Low Energy Secure Connections
ECDH	Elliptic-Curve Diffie-Hellman
GPS	Global Positioning System
GNSS	Global Navigation Satellite System
PLL	Phase-Locked Loop
GSM	Global System for Mobile Communications
NTSC	National Television System Committee
PAL	Phase Alternating Line
DVB-T	Digital Video Broadcasting – Terrestrial

IoT	Internet of Things
SDK	Software Development Kit
DMA	Direct Memory Access
GPIO	General Purpose Input Output
SOS	Sum-of-Squares
UTC	Universal Coordinated Time
UART	Universal Asynchronous Receiver Transmitter
ECB	Electronic Code Book
CCM	Counter with Cipher Block Chaining Message Authentication Code
AES	Advanced Encryption Standard
RFRA	Radio Frequency Retroflector Attack
NSA	National Security Agency
ANT	Advanced Network Technology
EmSec	Emission Security
FCC	Federal Communications Commission
ANFR	Agence Nationale des Fréquences

Part I

NOVEL ATTACK OPPORTUNITIES

Modern connected devices have both computation and communication capabilities. The interaction between digital blocks, which produce strong electromagnetic noise, and radio receivers and transmitters, which are sensitive to such noise, is a largely unexplored source of security problems.

1 INTRODUCTION

Radio communications, first demonstrated towards the end of the 19th century, now pervade our daily life, and their security is paramount.

1.1 PERVASIVENESS OF WIRELESS COMMUNICATIONS

One of the first 'wireless' communication networks was the French optical telegraph, invented by Claude Chappe, and deployed in France towards the end of the 18th century. In this system, messages propagated through a chain of towers spaced by several kilometers. Each symbol was displayed on one tower and copied on the next by human operators equipped with mechanical semaphores and telescopes. The optical telegraph, operated by the revolutionary government, was fast and independent from the postal service, still influenced by the aristocracy [104]. Interestingly, it was also subject to some of the first examples of attacks to a modern information system, real [232], and fictional [104]. Figure 1.1 shows a tower with its ingenious mechanical semaphore.

Towards the end of the 19th century, many scientists and practitioners started experimenting with electromagnetism and wireless communications [85]. Curiously, the early radios used a detector of electromagnetic waves that was later found to behave like a memristor, an electronic component discovered several decades later [83]. Over the 20th century radio communications developed in unprecedented ways, becoming one of the central technologies in the modern information society. To cite a few examples, radio communications played a fundamental role in the safety of transport systems, in the development of mass media with audio and video broadcasts, in the widespread use of personal mobile communication devices, and in the availability of internet connectivity in remote areas.

Over the last decades, we assisted to an even larger use of radio devices, which now pervade many aspects of daily life. When talking about the modern development of information technologies, we often mention the large number of transistors and processors that persons carry, often unknowingly, in their devices. Interestingly, the same holds for radio transceivers. A prominent example are modern smartphones. Going beyond the initial purpose of communicating with other phones, smartphones can also receive satellite signals for

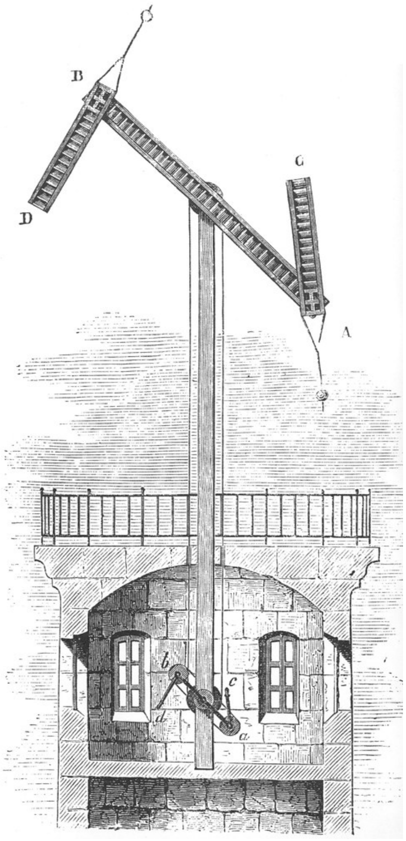


Figure 1.1: A tower of the optical telegraph ("1792-02" by ITU Pictures is licensed with CC BY 2.0).

precise positioning, receive FM radio broadcasts, query and emulate cards and tags for electronic payments, connect to wireless networks to access the internet, and control smart devices in their proximity. Often, a smartphone acts as gateway between smart devices using low-power short-range communications and the internet.

Besides the civil, military, and commercial applications that we have described, radio amateurs have been operating radio communications for leisure over the last century. The article 'How to Construct an Efficient Wireless Telegraphy Apparatus at Small Cost' that appeared in *Scientific American* in 1902 introduced amateur ('ham') radio in the USA [53]. In the past 30 years, the advent of Software Defined Radio (SDR) has democratized radio reception and transmission for leisure, research, and attacks. In an SDR, most of the layers of a radio communication protocol are implemented in software. Only the minimal components necessary to receive or transmit the physical radio signal are implemented in hardware. Active communities of enthusiasts have designed receivers and transmitters for most existing protocols (not only those of amateur radio), and made them available as open-source software. Cheap SDR transceivers are now available on the market as easy-to-use off-the-shelf devices. For example, an inexperienced user with a small budget can easily listen to satellite transmissions, impersonate a cellular network, or transmit rogue satellite positioning signals.

Although experimenting with radio communications is nowadays very accessible, transmissions and reception are still strongly regulated. Radio communications travel through the air, occupying a certain frequency band. The frequency spectrum is therefore a very precious resource that operators in a given geographical region have to share. Countries have dedicated agencies that ensure the correct and fair use of radio waves. For, example, the Federal Communications Commission (FCC) plays this role in the United States of America, and the Agence Nationale des Fréquences (ANFR) does the same in France. Radio operators generally need a licence and obey strict rules. Electronic devices must be tested for compliance before entering the market.[1] For example, the electromagnetic noise they produce at various frequencies must stay below fixed limits defined to ensure the safe coexistence with other devices. Countries also assign and sell radio frequencies to commercial operators. Specific frequencies are available for free to licensed radio amateurs. Violations are investigated and sanctioned severely. For example, in 2017 a person was charged for using an illegal device that accidentally disrupted the operations of an airport in France.[2]

1.2 INTEGRATION CHALLENGES

The recent democratization of wireless communications was possible thanks to huge advances in the integration of analog and radio frequency components into single chips. Similarly to digital electronics, radio systems that were once made of discrete components are now integrated into a single block of silicon, becoming smaller, cheaper, and more energy efficient. Additionally, digital and radio blocks are often placed on the same ‘mixed-signal’ silicon die. Multi-standard multi-frequency transceivers are combined with digital processors and peripherals, providing the communication and computation capabilities required by modern connected devices in one place. This is very convenient for embedded systems and Internet of Things (IoT) devices. In more complex systems, such as smartphones, multiple digital and radio chips are at their turn integrated on the same electronic platform.

Unfortunately, the close physical proximity of digital and radio blocks requires to solve challenging functional and performance issues. On the one hand, digital blocks are strong sources of electromagnetic and radio frequency noise. Intuitively, the sharp current spikes corre-

[1] Devices complying with the FCC regulation have an identifier called FCCID. Information about a device with a given identifier is available at `https://fccid.io/`.

[2] News available, for example, at `https://www.connexionfrance.com/French-news/Forgotten-GPS-jammer-costs-motorist-2-000`.

sponding to transitions in the logic value of binary signals generate electromagnetic fields. In addition, several nonlinearities in the silicon components cause the intermodulation of different noise components. Unlike thermal noise, which can be regarded as a random physical process, digital noise has distinctive characteristics. For example, it presents narrow-band spikes at the frequency of operation of the digital blocks. On the other hand, radio circuits usually deal with thermal noise and other random noise sources that are due to the inherent nature of electronic components. Traditionally, digital and radio circuits were developed separately. The former without much concern about their impact on the radio spectrum. The latter without much consideration for digital noise. This has entailed the problematic coexistence of noisy digital blocks and noise-sensitive radio transceivers on the same system [255], where many noise coupling paths exist between the two. This happens both for mixed-signal chips [5, 27] and for platforms [153, 154, 179, 239, 255]. Many of the integration problems have been successfully understood and partially solved, and these architectures are nowadays the norm. However, as the complexity of the systems increases, design remains a challenge. In particular, modeling and simulating the electromagnetic emissions and interactions in a complex system is hard and computationally expensive. Solving the equations is often unfeasible and numerical approximate solutions remain complex. Testing is expensive and can happen only at a late stage of the development. Therefore, unexpected Electromagnetic Interference (EMI) and Radio-Frequency Interference (RFI) interactions might still appear in final products [153]. Often, these problems cannot be completely solved, but they can be ignored as long as they do not severely impact the functionality and performance of the product in normal conditions, and as long as regulations are met. For example, spurious emissions in a given band can be tolerated if they are below the limits set by the FCC.

1.3 SECURITY CHALLENGES

On the one hand, if we see the digital blocks as a generic noise source, the problems arising from the integration with radio transceivers are merely functional. On the other hand, if we consider that the digital blocks process sensitive information, we can suppose that unexpected interactions with radio blocks might lead to dangerous information flows.

1.3.1 Unexpected Side Channels

Connected devices must secure their wireless communications, for example, from attacks against confidentiality or integrity of the data.

To this purpose, they run cryptographic algorithms that protect a given layer. For example, Bluetooth Low Energy (BLE) uses public key cryptography to pair two devices, and symmetric cryptography to encrypt the link layer. Alternatively, cryptography is used at the application layer, for example, for authentication in Google Eddystone beacons [98].

It is well known that the implementation of a cryptographic algorithm might leak sensitive information about its internal computations through the observation of its execution on digital hardware. Side channel attacks exploit this data-dependent leakage to recover a cryptographic secret, such as the key of a symmetric cipher. There exist many types of quantities that reveal information about execution, for example, the execution time [142], power consumption [143], and electromagnetic emissions [7]. The latter emanate directly from a component, or they modulate another strong signal in the system, such as the clock [7].

Timing side channel attacks are sometimes possible remotely or from a large distance. For example, they could be measured while interacting with a WiFi device over-the-air [280]. On the contrary, power side channels are typically only possible with physical access, and electromagnetic side channels in close proximity. Only a few attacks were shown at larger distance, for example at 1 m from the target in a controlled environment [219]. Power and electromagnetic side channel attacks against a wireless node might break the cipher that protects the communication protocol [20, 195, 225]. In this case attackers might interact with the device over-the-air, but they still need physical proximity.

In some specific cases, the presence of a radio front-end close to a digital block that executes a cryptographic algorithm creates novel types of side channels. Radio Frequency Identification (RFID) tags are designed to reflect a modulated version of the carrier sent by and external reader (backscattering). This can be achieved by modulating the impedance of the antenna. In passive tags, the digital blocks are powered by the energy of the external carrier. As a result, the power consumption of the digital blocks modulates the impedance of the antenna, modulating the backscattered wave, and leaking side channel information. This effect is called parasitic backscattering [204–206, 245]. A similar effect, called parasitic load modulation, happens in passive Near Field Communication (NFC) devices [146]. In both cases successful side channel attacks against a cryptographic algorithm have been shown up to a distance of 1 m. These attacks require an external carrier (possibly produced by a legitimate reader). They can be mitigated by decoupling the digital part from the antenna.

Finally, the interactions of the user with a smartphone's screen modify the Channel State Information (CSI) of WiFi [152], so that an attacker with access to a malicious access point is able to retrieve, for example, a pin entered by the user at a distance of up to 1.6 m.

1.3.2 Unexpected Covert Channels

It is well known that an attacker can trigger and modulate electromagnetic leakages from hardware components to exfiltrate data from a machine without network connectivity. First proposed in 1998 under the name of Soft-TEMPEST [11, 149], these attacks have been demonstrated with many types of leakages over the years, but always using very simple modulation schemes. On devices with radio front-ends, Soft-TEMPEST leakages might interact with radio signals through multiple cascaded effects, leading to Second-Order Soft-TEMPEST attacks [55, 69], recently proposed in 2018. These cascaded effects could be unintentional, or they could be introduced with hardware trojans. An example of Second-Order Soft-TEMPEST attack is a polyglot modulation of WiFi packets in which covert information is modulated on top of intended data [69].

A similar covert channel on top of WiFi [236], or WiFi jamming [235], become possible by modifying the baseband firmware of a Broadcom WiFi transceiver present on many smartphones, using the Nexmon project [233, 234]. Indeed, these chips are implemented as an SDR, and custom firmware can transmit arbitrary signals on the WiFi channels.

The impedance of a WiFi card depends on its state (e.g., on/off). By turning on and off the WiFi card, an attacker can modulate the reflection of an incoming carrier or stream of packets (backscattering), thus becoming able to exfiltrate data [295, 296]. This is yet another example in which the unexpected interaction between software and the radio front-end introduces a security issue.

1.4 NOVEL ATTACK OPPORTUNITIES

In this thesis, we observe that the security problems that might arise from the the unexpected interactions between the activity of digital blocks and radio transceivers on the same device is still far from being exhaustively explored. In particular, we identify two novel attack opportunities on commodity devices:

- We conjecture that the activity of the digital blocks might accidentally modulate the radio signal transmitted by a radio transmitter. In this case, amplified side channel leakages would be broadcast, making side channel attacks possible from a large distance. An example of devices that might have this problem are mixed-signal chips where the radio transmitter is close to digital blocks on the same silicon die.

- We conjecture that unprivileged software could shape arbitrary radio signals from the electromagnetic leakages produced by its execution, for example, on a modern smartphone. This would effectively turn the smartphone in a generic SDR, opening novel attack opportunities, including the interaction with other receivers on the same platform, or nearby.

1.5 CONTRIBUTIONS

In summary, the main contribution of this thesis consists in two novel attacks on modern connected devices. In particular:

- We present *Screaming Channels*, a novel side channel vector that makes Electromagnetic (EM) side channel attacks possible at large distance.
  - We conjecture the presence of the channel and we observe promising effects on some chips.
  - We develop a complete attack and we conduct an in-depth analysis on a commercial BLE device.
  - We demonstrate several challenging attacks, at large distance, and against a real system.
- We present *Noise-SDR*, a method to turn unprivileged code running on Arm smartphones into a generic SDR.
  - We demonstrate how to shape noise into arbitrary radio signals leveraging a 1-bit coding scheme borrowed from the theory of fully-digital radios.
  - We show a practical implementation with DRAM accesses on modern Arm smartphones, and we also explore other noise sources that can be controlled from JavaScript.
  - We evaluate the transmission of a large number of protocols, leveraging them for applications such as device tracking and leakage detection.
  - We show that *Noise-SDR* can be used to interact with other radio receivers on the same smartphone, for example, to inject signals.

Developing such attacks required a considerable amount of experimental work ranging from firmware programming to radio measurements. To facilitate future research and guarantee the replicability of our results we have open sourced our code, data, and detailed instructions [311] for *Screaming Channels*. We will do the same for *Noise-SDR* after publication.

Our work on *Screaming Channels* led to two academic publications [35, 38], several talks [31–34, 37], coverage in the news (e.g., Le Monde [275], The Register [48]), the 3rd place at the Cyber Security Awareness Week (CSAW) Europe 2018, and a Google Bughunter Program Honorable Mention. We are aware that other researchers in industry and academia have successfully reproduced our results, and even proposed improvements using deep learning [290, 291]. Other attacks on mixed-signal chips have recently appeared in literature [94], confirming the importance of studying the unintentional coupling between blocks in complex systems from a security point of view.

Part II

BACKGROUND AND STATE OF THE ART

The work presented in this thesis builds on top of both side channel attack and analysis, and radio communications. We briefly recall those concepts that are necessary to understand our results, and we refer to specialized literature for more details. Then we discuss the state of the art.

2 BACKGROUND

2.1 BASICS OF RADIO COMMUNICATIONS

In this section we briefly recall the basics of radio communications that are necessary to understand this thesis.

2.1.1 Intuitive Principle

Intuitively, radio communications are based on the propagation of EM waves, generated by a transmitter, and picked up by a receiver, as shown in Figure 2.1. At the transmitter, an electrical current flowing in a conductor (the antenna) generates an EM field. EM waves propagate 'over the air' until the receiver. Here, they induce again a current in the antenna. The geometric shape of the antenna determines its resonant frequency, at which transmission and reception work best, and many other parameters about radiation.

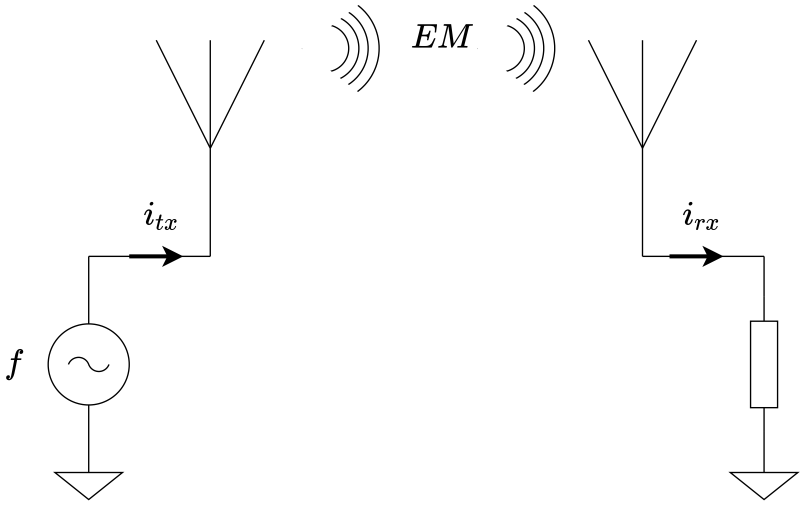


Figure 2.1: Intuitive principle of radio communications. A current at frequency f in the transmitting antenna generates EM waves that propagate to the receiving antenna, where they induce a current.

2.1.2 Electromagnetic propagation

The behavior of the EM field radiated by an antenna changes with distance. As the distance increases, the field quickly moves from being dominantly reactive (reactive near field) to dominantly radiating (radiating near field). Starting from a sufficiently large distance, the field approximately propagates with plane waves (far field). Given an antenna with maximum dimension D, and a field with wavelength λ, the far field approximation is valid at distance d if the following conditions are met:

$$\begin{aligned} d > 2D^2/\lambda \\ d \gg D \\ d \gg \lambda \end{aligned} \tag{2.1}$$

In the far field, at best, the radiated power decreases with the square law of the distance:

$$P_{rx}/P_{tx} \sim G_{tx}G_{rx}(\lambda/2\pi d)^2 \tag{2.2}$$

where G_{tx} and G_{rx} are the gains of the transmitting and receiving antennas, respectively. They highly depend on their shape. Higher wavelengths (lower frequencies) propagate better. Propagation through a different medium might have different properties. Sometimes in our experiments we use a coaxial cable. In this case the decrease is exponential $e^{-\gamma d}$, where γ depend on the frequency and on the type of cable.

2.1.3 Signals Used in Radio Communications

Assuming a basic familiarity with signal theory, we briefly describe how radio signals transmit information.

2.1.3.1 *Intuitive Principle*

Radio communications usually work by transmitting an EM signal at frequency f_c, called *carrier* frequency. The radio signal is composed by a sinusoidal carrier whose parameters are modulated with the data to transmit. Figure 2.2 shows an intuitive example of amplitude, frequency, and phase modulation. Modulation can be analog (the parameters can take any value), or digital (the parameters can take only discrete values in a finite set of symbols). In the frequency domain, the radio signal occupies a limited bandwidth $2B$ around the carrier frequency, from $f_c - B$ to $f_c + B$. The carrier frequency is the one at which the signal propagates over the air. This signal is often referred to as *passband*. On the contrary, the signal that modulates the carrier occupies the band from $-B$ to B, and it is called *baseband* signal. The process of bringing a signal from baseband to passband is called *up-conversion*, and the converse is called *down-conversion*.

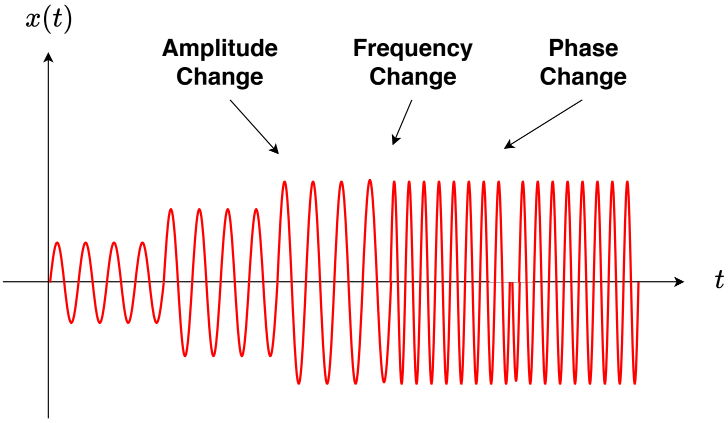


Figure 2.2: Carrier modulation.

2.1.3.2 *Analytic Signal*

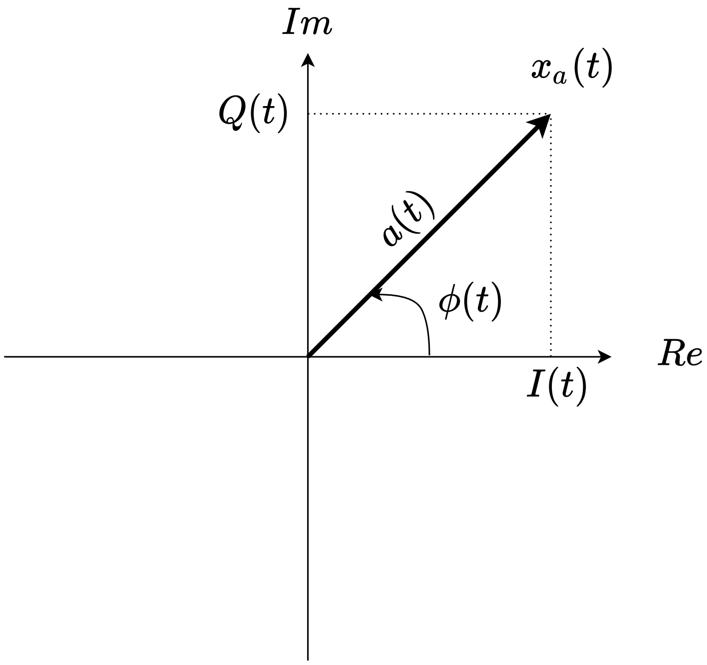


Figure 2.3: Complex signal notation.

It is very convenient to represent signals in the complex domain. One of the reasons is that trigonometric operations involving sinusoidal signals become trivial and intuitive.

As shown in Figure 2.3, an *analytic signal* is a vector in the complex plane. With a *polar* notation, it is defined as

$$x_a(t) = a(t)e^{i\phi(t)} \tag{2.3}$$

where $a(t)$ is its *instantaneous amplitude* and $\phi(t)$ is its *instantaneous phase*. Alternatively, with a *quadrature* notation, it is defined as

$$\begin{aligned} x_a(t) &= I(t) + iQ(t) \\ I(t) &= a(t)\cos(\phi(t)) \\ Q(t) &= a(t)\sin(\phi(t)) \end{aligned} \tag{2.4}$$

where $I(t)$ is the *in-phase* component and $Q(t)$ is the *quadrature* component. The name *quadrature* comes from the fact that I and Q are separated by an angle of $\pi/2$ in the complex plane.

A generic real signal can be now conveniently represented as the projection on the real axis of the corresponding analytic signal:

$$x(t) = \mathcal{R}e\{x_a(t)\} = \mathcal{R}e\{a(t)e^{i\phi(t)}\} = a(t)\cos(\phi(t)) \tag{2.5}$$

The 'speed' in Hz at which the analytic signal rotates on the plane is called *instantaneous frequency*, and it can be computed from the derivative of the instantaneous phase as:

$$f(t) = \frac{1}{2\pi}\frac{\partial\phi}{\partial t}(t) \tag{2.6}$$

2.1.3.3 *Modulated Carrier*

A generic radio signal such as the one in Figure 2.2 is written as:

$$x_{rf}(t) = \mathcal{R}e\{a(t)e^{i\theta(t)}e^{i2\pi f_c t}\} = a(t)\cos(2\pi f_c t + \theta(t)) \tag{2.7}$$

where $a(t)e^{i\theta(t)}$ is the analytic representation of the baseband signal and $e^{i2\pi f_c t}$ is the analytic representation of the carrier. We observe that the baseband signal is up-converted to passband by multiplication with a radio frequency carrier. The bandwidth of the baseband signal must be much smaller than the carrier frequency.

The baseband signal controls the instantaneous amplitude $a(t)$ and phase $\theta(t)$ of the carrier. It also controls its instantaneous frequency:

$$f(t) = f_c + \frac{1}{2\pi}\frac{\partial\theta}{\partial t}(t) \tag{2.8}$$

2.1.4 Modulation Types and Protocols

We briefly recall the main modulation schemes used in this thesis, listed in Table 2.1. Simple modulation schemes control the amplitude (AM, OOK, ASK), the frequency (FM, FSK, DTMF, BFSK, MFSK, GFSK), or the phase (PSK, BPSK, 4-QAM) of the carrier. AM and FM are analog modulations, whereas the others are digital. OFDM systems use several orthogonal subcarriers, each modulated with one of the simpler modulation schemes, increasing the data rate. To increase the resistance to noise, spread spectrum schemes spread the original signal over a wide band of frequencies. The DSSS method consists in

Table 2.1: Modulation types.

Acronym
Amplitude Modulation (AM)
Frequency Modulation (FM))
On Off Keying (OOK)
Amplitude Shift Keying (ASK)
Frequency Shift Keying (FSK)
Binary Frequency Shift Keying (BFSK)
M-ary Frequency Shift Keying (MFSK)
Dual-Tone Multi-Frequency (DTMF)
Gaussian Frequency Shift Keying (GFSK)
Audio Frequency Shift Keying (AFSK)
Phase Shift Keying (PSK)
Binary Phase Shift Keying (BPSK)
Quadrature Amplitude Modulation (QAM)
Direct Sequence Spread Spectrum (DSSS)
Chirp Spread Spectrum (CSS)
Orthogonal Frequency Division Multiplexing (OFDM)
Upper Side Band (USB)

spreading a BFSK signal by multiplying it with a faster pseudo-random sequence with distinctive properties. Instead, CSS uses a chirp, that is, a signal that continuously sweeps over a range of frequencies. The radio signal spectrum occupies the band from $f_c - B$ to $f_c + B$, where B is the bandwidth of the baseband signal. Some of this information is redundant because the spectrum is symmetric around f_c for amplitude-modulated signals. Single Side Band (SSB) transmits only half of the spectrum of the original signal, suppressing the other half. For example, USB transmits only the frequencies from f_c to $f_c + B$.

Apart from the modulation type, radio communications are characterized by protocol layers described in standard specifications. For example, digital protocols transmit data in packets and support various types of network configurations. In this thesis we work with several existing radio protocols, listed in Table 2.2 together with a reference to their documentation. Throughout the thesis we will provide more details when necessary.

2.1.5 Diversity

2.1.5.1 *Motivation*

In a simple radio transmission a signal is sent once over a noisy channel. Besides thermal noise, the channel might be affected by other problems. For example, obstacles could create deep fade conditions in which signals do not arrive at destination, or produce multiple reflec-

Table 2.2: Protocols.

Acronym	Reference
Voice AM	[247]
Voice Narrow-Band Frequency Modulation (NBFM)	[249]
PSK31	[251]
Radio Teletype (RTTY)45.45	[252]
MFSK128	[75]
Olivia 64/2000	[250]
2xPSK500	[75]
Slow Scan Tele Vision (SSTV)	[253]
Ham Digital Radio Mondiale (DRM)	[248]
FT4	[268]
LoRa	[78, 226, 267]
GLONASS	[44]
BLE	[228]
ANT/ANT+	[2]
Enhanced ShockBurst (ESB)™	[241]
Gazel™	[242]
WiFi	[132]

tions that arrive through different paths at different times. Another example is an interfering signal that disrupts a certain frequency band for some time.

To solve these problems, diversity schemes combine several copies of the same signal received through multiple independent channels. Intuitively, these methods use redundancy to maximize the signal at reception. The resulting signal increase is called diversity gain.

2.1.5.2 *Diversity Types and Combination Methods*

There exist several ways to achieve diversity. Time diversity consists in sending the same signal multiple times over the same channel. Frequency diversity consists in sending the same signal at multiple frequencies. In spatial diversity, the receiver uses multiple antennas in different positions to receive different copies of the same signal arriving from different paths, as shown in Figure 2.4.

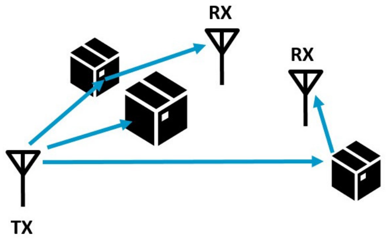


Figure 2.4: Spatial diversity.

Once multiple copies of the same signal arrive at the receiver, they can be combined in different ways [26]. First, the signals must be synchronized in time, then the receiver can choose the best one (selection), or add them using the same weight for each copy (equal gain), or assigning a weight based on the signal quality (maximal ratio).

2.2 MAIN RADIO ARCHITECTURES

2.2.1 Basic Components

In modern radios the baseband signal is generated in the digital domain, in software or in hardware. Then, it is converted into an analog signal using a multi-bit Digital-to-Analog Converter (DAC), whose symbol is shown in Figure 2.5.

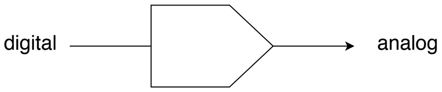


Figure 2.5: Symbol of a multi-bit DAC.

The carrier sine wave is generated using a frequency synthesizer. Usually, this is made by a Voltage Controlled Oscillator (VCO) and a Phase-Locked Loop (PLL). The VCO is an oscillator whose frequency can be controlled with the input voltage. The PLL is a closed-loop control system that adjusts the frequency and phase of the oscillator. For simplicity, we will usually depict a sinusoidal wave generator as in Figure 2.6.

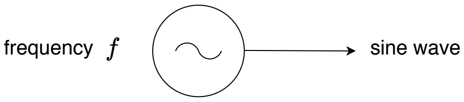


Figure 2.6: Symbol of a sine wave generator.

The baseband signal is multiplied with the carrier using a mixer, shown in Figure 2.7. The multiplication of two sinusoidal waves at frequencies f_1 and f_2 results in the generation of two new components at frequency $f_1 \pm f_2$. A mixer is then useful to translate signals in frequency. In particular, it is used to up-convert the baseband signal to the carrier frequency.

At various stages of transmission and reception, several types of filters are used to filter undesired frequency components. Finally, a Power Amplifier (PA) is used to increase the power of the signal before feeding it to an antenna. The power amplifier, shown in Figure 2.8 draws the power from the power supply, and it is characterized by its gain $G = P_{out}/P_{in}$.

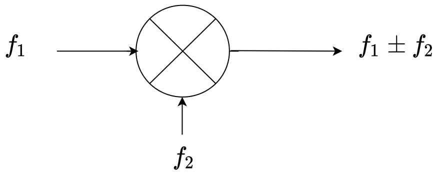


Figure 2.7: Symbol of a mixer.

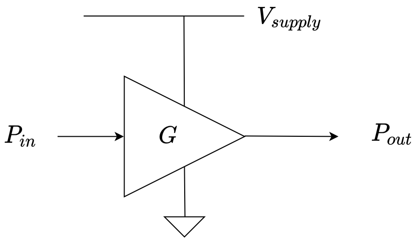


Figure 2.8: Symbol of a PA.

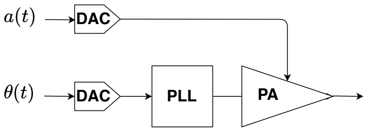


Figure 2.9: Polar modulator.

2.2.2 Classic Radios

2.2.2.1 *Polar vs. Quadrature*

The *polar* architecture in Figure 2.9 follows closely the polar notation. A frequency synthesizer generates the carrier, whose frequency and phase can be controlled. The signal is then fed to a power amplifier, whose gain is controlled in order to modulate the amplitude of the output signal.

The *quadrature* architecture in Figure 2.10 follows closely the quadrature notation. A frequency synthesizer generates both the in-phase and quadrature components of the carrier. They are mixed with the respective components of the baseband signal. Then, they are added and sent to the power amplifier. Filters help removing undesired frequency components and respecting the specifications for emissions in the spectrum (the spectral mask) [16].

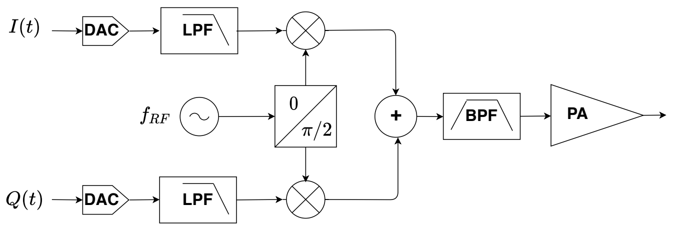


Figure 2.10: Direct conversion transmitter.

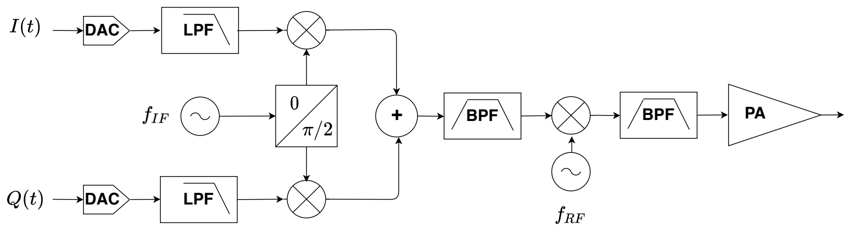


Figure 2.11: Superheterodyne transmitter.

2.2.2.2 *Direct vs. Superheterodyne*

In the *direct conversion* architecture in Figure 2.10 (and Figure 2.9), the baseband signal is directly converted to the desired carrier frequency by one mixing stage. In contrast, the *superheterodyne* transmitter in Figure 2.11 uses two separate stages. The Intermediate Frequency (IF) stage up-converts the signal to an intermediate carrier frequency. Then, the Radio Frequency (RF) stage up-converts the signal to the final desired carrier frequency.

These architectures all reach the same goal, but each of them has different trade-offs when implemented with actual hardware components and different technologies. Discussing these aspects in more detail is beyond the scope of this thesis. Refer to [16] for a comparison in the specific case of radios for Wireless Local Area Network (WLAN) applications.

2.2.3 Fully-Digital Radios

2.2.3.1 *Motivation and Main Strategies*

The architectures presented in the previous section use analog and radio-frequency hardware components, which must be integrated into a single chip, possibly with other digital circuits. In contrast, fully-digital radios try to minimize the number and complexity of the analog electronic components required to generate a radio signal.

One of the main strategies consists in leveraging 1-bit coding techniques to generate an arbitrary signal from a square wave. This is

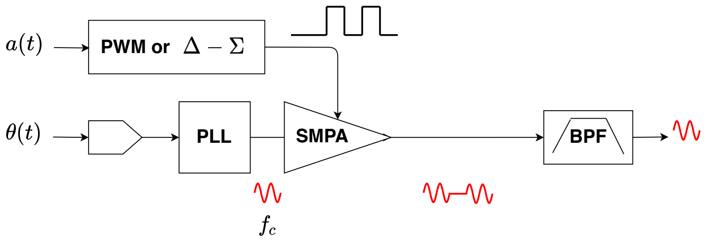


Figure 2.12: Fully-digital radio based on (baseband) PWM or $\Delta\Sigma$. In this case the baseband component of the square wave is used to modulate the amplitude of a phase/frequency-modulated sinusoidal carrier.

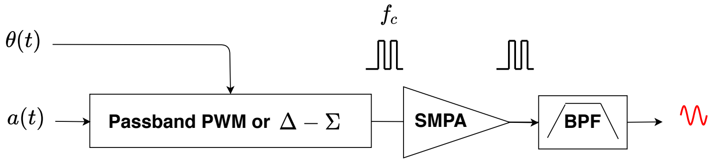


Figure 2.13: Fully-digital radio based on RF-PWM or *passband*-$\Delta\Sigma$. In this case the amplitude/frequency/phase-modulated sinusoidal carrier is directly generated as one component of a square wave.

possible by modulating one of the frequency components of a square signal, and filtering the other ones as noise. The filtering stage is done after amplification, and the power amplifier works with square signals. This makes it possible to use a highly efficient Switch-Mode Power Amplifier (SMPA) instead of a less efficient linear power amplifier.

Two main 1-bit coding techniques are (baseband) Pulse Width Modulation (PWM) and (baseband) Delta-Sigma ($\Delta\Sigma$). In this case, the baseband component of a square wave is used to modulate the amplitude of a frequency/phase-modulated carrier. Alternatively, some schemes use the Radio-Frequency Pulse-Width Modulation (RF-PWM) (*passband*-PWM) and *passband*-$\Delta\Sigma$ techniques. In this case, the modulated radio signal is generated directly from one of the passband components of a square wave. Figure 2.12 and Figure 2.13 depict the baseband and passband case, respectively. In the following we will describe the PWM and RF-PWM techniques in more detail, with focus on RF-PWM.

We refer the reader to [193, 202] for a detailed background about fully-digital radios, and to [105, 150, 193, 199, 217, 287] for the theory and examples of practical implementations of RF-PWM.

2.2.3.2 *Background: A Modulated Square Wave*

A generic square wave is characterized by three parameters: the fundamental frequency at which the pulses repeat, the phase of the pulses, and the pulse width. The ratio between pulse width and pulse period is commonly referred to as duty-cycle.

A square wave that can take values 0 or 1, with duty-cycle $\delta(t)$, fundamental frequency f_0 and phase $\theta(t)$ can be written as the following sum of sinusoidal components:

$$\begin{aligned} \mathrm{offset} &= \delta(t) \\ \mathrm{fundamental} &= \frac{2}{\pi}\sin(\pi\delta(t))\cos(2\pi f_0 t + \theta(t)) \\ \mathrm{harmonics} &= \sum_{k=1}^{k=+\infty} \frac{2}{k\pi}\sin(k\pi\delta(t))\cos(2k\pi f_0 + k\theta(t)) \end{aligned} \tag{2.9}$$

2.2.3.3 *Baseband PWM*

The classic PWM technique consists in modulating the duty-cycle to generate an arbitrary baseband signal (the offset), while filtering all the other components (the fundamental and the harmonics). It is very common in many applications. For example, it is an effective method to control the intensity of LED lights and the speed of electric motors.

In the baseband PWM radio architecture (Figure 2.12), PWM is used as 1-bit DAC to modulate the amplitude of the carrier by controlling the switch-mode power amplifier. The sinusoidal radio carrier is instead generated and modulated in frequency/amplitude in a conventional way. As a result, the signal at the output of the amplifier is:

$$[\delta(t) + \mathrm{fundamental} + \mathrm{harmonics}]\cos(2\pi f_c t + \theta(t)) \tag{2.10}$$

One or more filters cut all the undesired frequency components originating from the fundamental and the harmonics of the PWM square wave, leaving only the desired modulated carrier:

$$\delta(t)\cos(2\pi f_c t + \theta(t)) \tag{2.11}$$

A proper choice of f_0 and f_c compared to the bandwidth of $a(t)$ and $\theta(t)$ is necessary to ensure that filtering is possible.

2.2.3.4 *RF-PWM*

The RF-PWM uses the fundamental component of the PWM square wave directly as an RF carrier, hence the name RF-PWM. This is possible by generating a square wave with the following parameters:

$$\begin{aligned} f_0 &= f_c \\ \delta(t) &= \frac{\arcsin(a(t))}{\pi} \\ &\theta(t) \end{aligned} \tag{2.12}$$

The resulting decomposition is:

$$\begin{aligned} \mathrm{offset} &= \frac{\arcsin(a(t))}{\pi} \\ \mathrm{fundamental} &= \frac{2}{\pi} a(t)\cos(2\pi f_c t + \theta(t)) \\ \mathrm{harmonics} &= \sum_{k=2}^{k=+\infty} \frac{2}{k\pi} \sin(k\pi\delta(t))\cos(2\pi k f_c t + k\theta(t)) \end{aligned} \tag{2.13}$$

The fundamental component is directly in the form of a sinusoidal carrier at RF, modulated in amplitude and frequency/phase. A filter removes the offset and the harmonics, leaving only the desired modulated carrier:

$$\frac{2}{\pi} a(t)\cos(2\pi f_c t + \theta(t)) \tag{2.14}$$

Like for the baseband PWM case, a proper choice of f_c compared to the bandwidth of the modulating signal is necessary.

2.2.4 Backscattering and Load Modulation

The radio architectures that we have described until now actively generate a carrier signal. Some small devices with a very small battery cannot afford wasting energy to produce a strong carrier. In this case, a better solution consists in modulating the carrier generated by an external reader. The external carrier can even power completely passive devices without any internal power source. This method, depicted in Figure 2.14, is called backscattering.

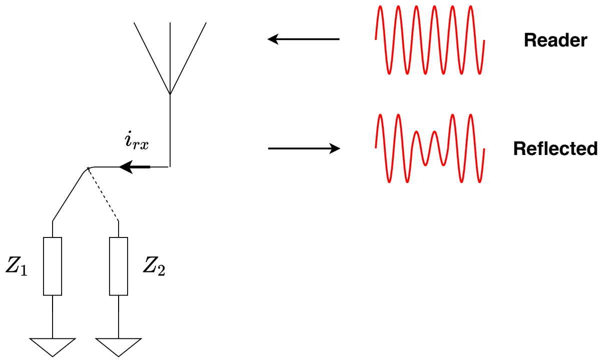


Figure 2.14: Backscattering.

When the incident wave from the reader reaches the antenna, its power is partially absorbed and partially reflected depending on the impedance to which the antenna is connected. By modulating this impedance, the device controls the reflection of a modulated signal. Backscattering is commonly used by RFID tags. Previous work has also explored the possibility for low-power devices to transmit data by backscattering ambient transmissions. To cite a few examples, some systems use TV and cellular transmissions [159], WiFi communications [138], and signals from FM stations [288]. NFC tags use a similar technique called load modulation. In this case, the antennas of the transmitter and of the receiver are close together and inductively coupled. The tag modulates the impedance (load) of its antenna (seen by the reader) to communicate data to the reader.

2.2.5 Software Defined Radios

A Software Defined Radio (SDR) is a type of radio in which hardware is minimal and flexible, whereas processing and configuration are mostly performed in software. In a typical SDR, the hardware does only the up-conversion (transmitter) or down-conversion (receiver). The DAC and the Analog-to-Digital Converter (ADC) convert the from the digital baseband signal to the analog domain, and vice-versa. Software configures the center frequency, bandwidth, gain, and other parameters of the radio front-end. All protocol layers, from the digital baseband to the application, are implemented in software. In some cases, a Field Programmable Gate Array (FPGA) is available to offload the most expensive computations.

Some examples of SDR transceivers are the *Great Scott Gadgets HackRF* [102], the *Ettus Research USRP N210* [70], and the *Ettus Research USRP B210* [71]. On the software side, GNURadio [82] is an open-source tool to develop reception and transmission chains. Many examples of transceivers exist and are available online, for example, for WiFi [307] and LoRa [267]. Other tools are available, for example: *Gqrx* [56] (generic reception), *Qsstv* [212] (SSTV and HamDRM), *FLDigi* [76] (most amateur radio protocols), *WSJT-X* [269] (FT4), *gnss-sdr* [310] (Global Navigation Satellite System (GNSS) reception).

Figure Figure 2.15 shows an example of SDR transceiver connected to a laptop running GNURadio.

2.3 SIDE CHANNEL ATTACKS THEORY

In this section we briefly recall how to analyze side channel leakages and how to perform key-recovery attacks.

2.3.1 Introduction

2.3.1.1 *Principle*

In many cases, the security guarantees of a cryptographic algorithm can be violated in its practical implementation. In particular, the

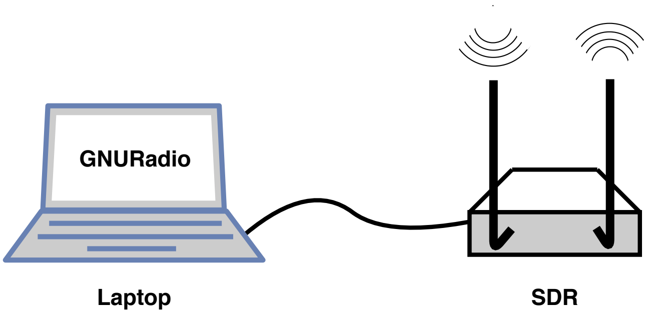


Figure 2.15: An SDR connected to a laptop running GNURadio.

execution of the algorithm often leaks some information about its internal computations through some measurable physical quantities. They include execution time [142], power consumption [143], and electromagnetic emissions [7], to cite a few. Multi-channel attacks introduced in [6] combine side channel traces coming from different types of leakages. Figure 2.16 shows an example of power and EM measurement on a smart card.

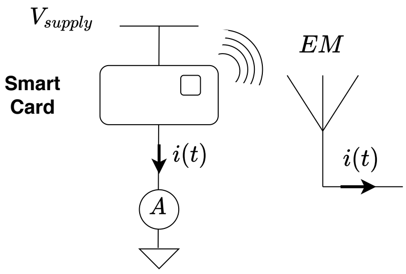


Figure 2.16: Power and EM side channels against a smart card.

Several attacks exist that leverage this data dependency and statistical analysis to recover the secret key. Usually, they require some assumptions, for example, that the attacker can observe a large number of encryptions with known plaintext. Often, the attack phase is preceded by a profiling step, during which the attacker builds a model of the leakage. The ability to attack a device using a profile built on a different instance is important in practical scenarios. Indeed, profiling often requires full control over the target, whereas attacks happen in more constrained scenarios.

2.3.1.2 *Advanced Encryption Standard (AES)*

Even though side channel attacks are not limited to a specific cryptographic algorithm, we focus on the AES [211] symmetric cipher, with a 128-bit key. It is characterized by a key schedule followed by ten rounds. At the first round, each plaintext byte p is xored with the corresponding key byte k, and given as input to an S_{box} function (Figure 2.18). The input $p \oplus k$ and output $S_{box}(p \oplus k)$ of the S_{box} are the usual targets for side channel attacks.

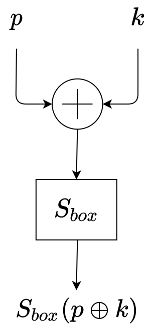


Figure 2.17: AES S_{box}.

2.3.1.3 *Important Notation and Terminology*

Divide and conquer attacks target a *leakage variable* y that is a function of one byte of the plaintext and one byte of they key. For example, a popular leakage variable for AES is the output of the S_{box} of the first round: $y = S_{box}(p \oplus k)$. The attacker can measure a physical quantity that is affected by y. The actual (unknown) *leakage function* is $l(y)$. The attacker models this leakage with a *leakage model* $\mathtt{model}(y)$. An example of simple leakage model is assuming that the leakage follows the Hamming Weight of the leakage variable.

2.3.2 Side Channel Analysis

2.3.2.1 *Detection of Informative Points*

The input of side channel analysis is a set of time traces measured while encrypting with random plaintexts and keys. By comparing these traces, an attacker can detect some points in time for which the traces corresponding to different values of the leakage variable show a significant statistical difference. This difference can be seen as the side-channel signal. The informative points are called Points Of Interest (POIs). There exist several methods to detect a leakage and the corresponding POIs. In this thesis we focus on:

- **fixed vs. fixed t-test** [65]. The Welch's t-test [292] can be used to check whether the expected values of two random variables

X and Y show a significant statistical difference, and it can be naturally applied to study whether or not a difference in the plaintext and key produces a difference in the measured traces. It is defined as:

$$(\bar{X}-\bar{Y})/\sqrt{\frac{var(X)}{N_x}+\frac{var(Y)}{N_y}} \tag{2.15}$$

where N is the sample size. Many options exist for the choice of the two sets to compare (e.g., [15, 91, 167]), for example, a set of traces with random plaintext and random key against a set of traces with fixed key. Better results can be achieved with the fixed vs. fixed test proposed in [65], which compares two sets with fixed plaintext and key, chosen to maximize the expected difference between the two sets.

- k**-fold** ρ**-test** [65]. The ρ-test first splits the trace set into a profiling set and a test set. The profiling set is used to estimate a (possibly nonlinear) model $\mathtt{model}(y)$ of the leakage, for each possible value of the leakage variable y. This can be done by classifying the traces in groups (classes) L_y according to the value of y, and then estimating the average trace for each class:

  $$\hat{m}_Y(y)=\bar{L}_y \tag{2.16}$$

  Then, the Pearson's correlation coefficient is used to estimate the linear correlation between the traces in the test set and the corresponding model:

  $$\hat{r}=\hat{\rho}(L(y),\hat{m}_Y(y)) \tag{2.17}$$

  To reduce the bias, the results obtained from k different partitions of the initial set into profiling and attack sets are averaged together. With N traces, $\hat{r}_z=\frac{1}{2\sqrt{N-3}}\ln(\frac{1+\hat{r}}{1-\hat{r}})$ approximately follows a normal distribution $\mathcal{N}(0,1)$ and can be used to to measure the confidence that data-dependent information is leaking.

2.3.2.2 *Studying Nonlinearity*

Linear regression was proposed in [231] to fit the leakage function. Naturally, this approach works well when fitting a linear leakage with a linear model. By choosing a higher-order polynomial as model, it can also fit more complex (non-linear) leakages. As highlighted in [222], the non-linear distortion of the leakage model (compared to the classic Hamming Weight model) can be studied by analyzing the results of linear regression.

2.3.3 Side Channel Attacks

2.3.3.1 *Profiled Attacks*

In this thesis we focus on profiled attacks. In this case, the attack is preceded by a profiling stage, ideally performed on a different device instance. Profiling is done using a set of traces with known random plaintexts and keys, whereas attack is done using a (smaller) set of traces with known random plaintext and unknown key.

- **Profiled Correlation Attacks** [65]. Profiled correlation attacks are very similar to the ρ-test. With the same method, the attacker estimates the model $\mathtt{model}(y)$ of the leakage for each value of the leakage variable y, using a profiling set. Then, for each byte of the key, the attacker measures the correlation between the attack traces and the values indicated by the precomputed profile for a given key guess:

  $$\hat{r}(p, k_g) = \hat{\rho}(L(y(p, k_g)), \hat{m}_Y(y(p, k_g))) \tag{2.18}$$

  With enough traces, the right guess will show the best correlation.

- **Template Attacks** [46]. Using the profiling traces, the attacker estimates the average and covariance of the leakage for each of the values of y. Assuming a Gaussian distribution, the attacker can compute the conditional probability density function of L given Y (i.e., $\mathrm{pdf}(L = l|Y = y)$) for any of the values of y. Since it estimates also the covariance, this model can capture second order statistical relations. To reduce complexity, the attacker can first reduce each trace to the POIs only. In the attack phase, the attacker is given a trace l with known plaintext p and unknown key $\tilde{k}$. For each possible guess k_g of the key, the attacker computes the guessed leakage variable y_g. The conditional probability density function $\mathrm{pdf}(L = l|Y = y_g)$ indicated by the profile can be used as a discriminant for the guess. When multiple attack traces l_i are available, the attacker can combine the results (maximum likelihood) as:

  $$\prod_i \mathrm{pdf}(L = l_i|K = k_g, P = p_i) \tag{2.19}$$

  The correct key guess will correspond to the discriminants with the maximum value. The product can be replaced with a sum of logarithms to avoid numerical problems. To improve accuracy and reduce numerical problems, assuming that the covariance is the same for each value of y, the attacker can compute the pooled covariance matrix as the average of the individual matrices [19, 50, 197, 198].

2.3.3.2 *Leakage Order*

By construction, template attacks can capture a second order relation between leakage model and leakage, whereas profiled correlation attacks can only capture a linear relation. If template attacks do not perform better than profiled correlation attacks, then we can conclude that, at least for the given sample size, we are facing a first order leakage.

2.3.3.3 *Profile Reuse*

In a realistic scenario the attacker wants to reuse a profile built on a device different than the victim. For profiled correlation attacks, the linear correlation $r(P_2, P_1)$ of a profile P_2 with a profile P_1 (built in different conditions) gives the attacker a quantitative measure of the loss introduced by using P_2 instead of P_1. Indeed, the correlation of the attack traces L_a with P_2 can be computed from the correlation with P_1 as:

$$r(P_2, L_a) = r(P_2, P_1) r(P_1, L_a) \tag{2.20}$$

As a rule of thumb, the number N of attack traces required for an attack to succeed is:

$$N \propto c/\rho^2(P_2, L_a) \tag{2.21}$$

where c is a constant [260].

Previous work has studied and evaluated several strategies to make profile reuse effective, even in the presence of inter-device variations, with focus on template attacks. Often, even a simple normalization of the traces (e.g., offset removal, variance normalization) can significantly facilitate the cross-device template tacks [51, 67, 124, 178]. Another strategy consists in estimating a template using multiple devices. Although these templates are in general more robust, they are less effective when applied to a specific device [51, 124, 222]. It is also possible to use a portion of the attack traces to build a profile on the fly, without resorting to another device [222]. Finally, effective cross-device attacks with deep learning have been recently proposed in literature [40, 62, 95].

2.3.3.4 *Rank Estimation and Key Enumeration*

Typical divide and conquer attacks assign a probability (or score) to each possible value of each sub-key k. Starting from this output, key-ranking algorithms (e.g., [18, 93, 166, 283, 298]) estimate how many bits of the key were not recovered, and key-enumeration algorithms (e.g., [22, 166, 208, 284]) bruteforce those bits to recover the key, which can be faster than acquiring more traces. For example, [208] is based on the idea that a histogram describing which keys fall in a given

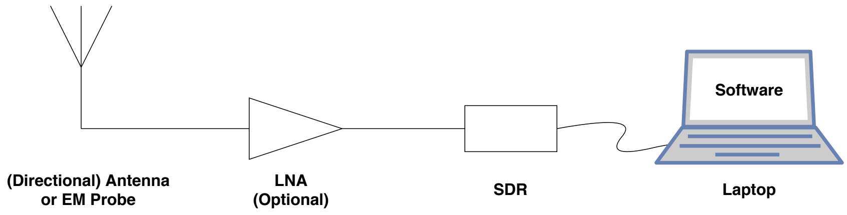


Figure 2.18: Reception chain.

probability range can be built by convoluting the histograms relative to each sub-key, directly constructed from the probability lists returned by the attack. This histogram is then used to count how many keys precede the right one (key ranking), or to try keys starting from the most likely until the good one is found (key enumeration).

2.4 RECEPTION SETUP

Our experimental setup consists of an SDR-based reception chain, shown in Figure 5.9, to receive both radio signals and side channel traces

The first block in the reception chain is the antenna. For short distances, we use standard WiFi antennas that have a good response in the Industrial, Scientific and Medical (ISM) band at 2.4 GHz. For larger distances, we use a *TP-LINK TL-ANT2424B* [266] directional WiFi antenna. Directional antennas are designed to maximize the power received from a given direction. This is convenient when the transmitter is in line of sight at a large distance, because the signal from the transmitter is maximized and the noise from other sources coming from other directions is minimized. For reception in close proximity to the emitter we use a loop probe designed for near-field measurement: a *NAE-HPROBE-15* [186], or a custom loop probe inspired from [54].

When external amplification is necessary, the second block in the chain is an Low Noise Amplifier (LNA). A LNA is an amplifier designed to amplify the input signal while adding very little noise. Depending on the desired frequency band and gain we use a *Minicircuits ZEL-1724LN* [175], a *ZKL-1R5* [176], a *ZX60-272LN* [177], or a *TEKBOX TBWA2* [264].

The last block in the hardware chain is an SDR. We use several widely available SDRs: a *Great Scott Gadgets HackRF* [102], an *Ettus Research USRP N210* [70], and an *Ettus Research USRP B210* [71]. The latter has two separate reception chains that are useful for spatial diversity.

On the software side, we use both custom code [311] based on *GNURadio* [82] and other SDR open source tools, such as *Gqrx* [56], *Qsstv* [212], *FLDigi* [76], *WSJT-X* [269], and *gnss-sdr* [310]. *GNURadio* is a framework to develop signal processing chains, whereas the other tools are dedicated to specific radio protocols.

3 STATE OF THE ART

3.1 WIRELESS SECURITY

Like any other communication channel, wireless communications should be secured using well designed protocols, good cryptographic schemes, and correct implementations.

3.1.1 Security Considerations

Many attack opportunities arise from the fact that the radio spectrum is a shared medium to which attackers have full access. This increases the attack surface of radio protocols at any layer. Attacks are possible over the air, without physical proximity, and often in a stealthy way.

Some examples of attack primitives due to the shared nature of the spectrum are: sniffing (collecting transmitted packets), jamming (disrupting communications by emitting an interfering signal), spoofing and injection (transmitting rogue signals or packets), relay (relaying a transmission over a different and often longer channel), and replay (retransmitting recorded packets). Often, these primitives are combined to achieve more complex results. For example, spoofing and injection of rogue signals into a victim receiver might be facilitated by jamming of the legitimate transmitter.

In the past, many attacks were thought to be impractical because receiving and transmitting certain radio signals required complex and expensive equipment, as well as knowledge and expertise to operate it. However, as highlighted in Chapter 2, the proliferation of WLANs and SDRs has invalidated this assumption. For example, WiFi and BLE transceivers are present in most consumer laptops and smartphones, and relatively inexpensive SDRs are able to receive and transmit most protocols, from Global System for Mobile Communications (GSM) to Global Positioning System (GPS). Besides more accessible hardware, open source tools are largely available for analysis and attack.

Covering all existing attacks against wireless communications is beyond the scope of this thesis. In the following, we will discuss some interesting examples.

3.1.2 Attacks

At the physical layer, a robust design and the detection of interference conditions helps protecting against jamming, but little can be done against an attacker that disrupts the channel. Jamming might be used as a defense, for example, to prevent offensive drones from reaching a sensitive target. Interestingly, in this case the jamming signal might itself reveal the position of the target, helping the drone to reach it [271].

For some protocols, the characteristics of the physical layer complicate sniffing from an external attacker. For example, BLE uses a form of spread spectrum called Frequency Hopping Spread Spectrum (FHSS). In this case the channel used by two communicating devices changes over time following a pseudo-random sequence. To collect the transmitted packets, an attacker must learn the sequence (e.g., by listening to the packets that establish the connection) and must have hardware able to quickly switch channel or to collect all channels at the same time. Both dedicated hardware [103] and SDR-based tools [141] exist to facilitate this task. In the case of Low Probability of Interception (LPI) signals, the physical layer is explicitly designed to prevent detection, complicating (but not always preventing) sniffing and jamming. Examples of simple methods are the use of FHSS, CSS, and DSSS with secret parameters [3].

In some applications, such as wireless car keys, the physical proximity between receiver and transmitter is used to verify the presence of the legitimate user. However, in many cases relay attacks between a car and its legitimate user at large distance are possible and practical [80]. Using ultra-wide band transmissions helps building distance measurement that is secure from relay attacks [254].

Many simple systems, such as wireless clickers, are often not protected against replay attacks, making it easy to record valid packets and retransmitting them later in time. The use of a rolling code, which is a counter synchronized between transmitter and receiver, helps providing a proof of freshness. Nevertheless, a rolling code alone is easy to bypass. Using a tool such as [207], attackers are able to reverse engineer the protocol and modify the value of the counter before replaying a message. The Rolling Jam attack uses a combination of jamming and replay to record a packet with the rolling code equal to the one following the last valid reception [134].

Stronger security guarantees are possible using cryptography, for example, for authentication and encryption. For example, access to personal WiFi networks is protected with a shared password using WiFi Protected Access (WPA), and the link layer is encrypted. However, bruteforce/dictionary attacks against the password are possible if the attacker captures the packets of the 4-way handshake of the authentication. To force such a condition, the attacker sends de-authentication packets to a legitimate device. Link layer encryption is also used by many other protocols, such as Bluetooth and BLE.

In some cases, protocol specifications contain vulnerabilities that create dangerous corner cases in which normal protections are not effective. For example, in the Bluetooth protocol the entropy of the keys can be negotiated, but negotiation is neither integrity protected nor encrypted. As a result, an attacker is able to force the use of a key with 1 byte of entropy, which is easy to bruteforce [12].

Sometimes, attackers are able to force the use of an older vulnerable protocol. This is the case for mobile communications that still support GSM. In recent mobile communication protocols both mobile phone and network have to prove their identity. In contrast, a mobile phone is not able to check the identity of a GSM network. Using moderately inexpensive SDRs and open source software [63], attackers can easily create a rogue GSM base station. If the signal is strong enough, the victim mobile phone is likely to connect to it.

The lack of authentication is a problem also for many civilian GNSS systems, as it makes position spoofing possible, with potentially catastrophic consequences. GPS spoofing has received particular attention in literature, for example, [274]. Open source tools to generate rogue baseband signals are available [130, 131, 289]. These signals can then be transmitted using an SDR. Some have investigated the use of simple hardware for transmission [59, 165], not caring about signal quality and spectrum pollution.

Some attacks target a vulnerability in the software stack that can be triggered over the air. For example, in [188] some GPS receivers do not perform a basic check on the denominator of a division during the calculation of the position. As a result, attackers are able to craft a spoofing signal that triggers a division by zero and serious crashes of the device. Several cases of remote code execution over the air have been demonstrated, too. For example, [244] shows how to exploit a memory corruption in the Android Bluetooth stack, which was likely never executed during testing and typical usage. Hence, the importance of developing effective analysis and testing techniques [127, 128, 227].

An attacker with control on the plaintext message sent with a wireless protocol (at the data link layer) might be able to craft full packets injected in a receiver (at the network layer) [97], or even craft packets injected in different types of receivers (with a different physical layer) [25]. In the first case, the attacker hides a full network layer packet into the payload of another packet. At the receiver, if the outer packet is discarded because of an error (e.g., bit error in the beginning of the frame because of interference), then the inner payload will be parsed as a valid packet. In the second case, the attacker additionally exploits the fact that the same modulated signal could be demodulated into valid bits by different receivers using different modulation schemes. To mention a simple example, an FSK signal could be interpreted as two OOK signals at two different frequencies. An OOK receiver tuned on one of the two frequencies used by an FSK transmitter would receive the same bits as the intended FSK receiver.

Finally, the spectrum is shared also by wireless transceivers on the same chip. For this reason, the WiFi and Bluetooth transceivers of some Broadcom chips coordinate their operation communicating between each other, making cross attacks possible (e.g., jamming, key stroke inference, turning Bluetooth vulnerabilities into WiFi vulnerabilities) [52, 227].

3.2 COMPROMISING EMANATIONS

Classic wireless security focuses on the analysis of intended communication channels over the air. Emission Security (EmSec) additionally studies the 'side' channels that arise from the unintentional interaction of computing devices with the physical world, resulting in the emission of physical signals that carry sensitive information.

3.2.1 Principle

Communication and computing devices are implemented with physical components. Many attacks that compromise their security arise from the interaction with the physical world. Sound, vibrations, light, magnetic and electromagnetic fields, and temperature are some examples of physical quantities that depend on the data processed by a device or that can alter its operation. EM emissions are particularly important in the context of modern electronic systems. Those at radio frequency are particularly interesting for their interaction with radio communication systems.

In the simplest case, attackers merely listen to the undesired emissions of a victim device to recover secret data, or to identify its location. Alternatively, there are several methods the attacker might use to facilitate the process. At the physical level, a hardware trojan could introduce or strengthen a leakage. The victim might be illuminated by an external carrier and reflect a copy modulated with sensitive information. The carrier could originate from an ambient transmission or from the attacker. Finally, an attacker could intentionally trigger and modulate leakages from software running on the victim, to exfiltrate data without using any network interface.

Emission security has always been a important concern for military and intelligence applications. Only towards the end of the 20th century it started to be investigated in depth in public literature. We refer the reader to [10] for a brief history of EmSec and a review of EM compromising emanations and other EM attacks. In the following we discuss the main applications and interesting examples. In this field, terminology and discovery time are not always clear and unambiguous, especially when public literature tried to interpret partial information present in declassified military documents, or when similar results where discovered in different contexts. We focus on highlighting the important features of the attacks, without aiming at a rigorous classification, and we follow the common usage of terminology.

3.2.2 Applications

3.2.2.1 *Eavesdropping Plaintext at a Distance*

Although computation and communication systems encrypt data to protect the confidentiality of secret information, the emission of physical leakages might reveal the plaintext, for example when it is typed on a keyboard, displayed on a screen, or spoken or played in form of audio. Thanks to these emissions, an external attacker without close access to the devices is able to retrieve secret information. For example, historically one of the main targets where the communications inside foreign embassies. These attacks are often called with the name TEMPEST, the code word used by the National Security Agency (NSA) in some declassified documents [182, 265].

Some of the first examples of TEMPEST attacks in public literature are those that retrieve the content of the video displayed on a screen by listening to the EM emissions at radio frequency from monitors, cables, and connectors. Attacks against monitors are also called Van Eck phreaking from the name of the author of one of first attacks on Cathode-Ray Tube (CRT) displays [66]. Refer to [148] for a historical review on this topic and attacks on more recent flat-panel displays. Recently, TEMPEST attacks have been demonstrated against the screen of mobile devices (tablets) [126]. On the one hand, countermeasures such as shielding and TEMPEST resistant fonts [148, 149] help protecting from these attacks. On the other hand, the advent of SDR receivers and open-source TEMPEST software [163, 224] has simplified the attack setup. Moreover, commodity devices might still present strong leakages, for example from cheap High Definition Multimedia Interface (HDMI) connectors. TEMPEST attacks can reach large distances, for example, an attack at 10 m was recently demonstrated [64]. Images on Liquid Crystal Display (LCD) screens can also be reconstructed through the sound emitted by their electronics components [89], which constitutes a serious threat during internet calls that transmit audio remotely.

A non-exhaustive list of other attacks includes optical emanations from CRT screens and Light Emitting Diode (LED) indicators [147, 160, 161], reflection of optical signals on surfaces [14], visible light bulbs vibrations modulated by sound [185], WiFi CSI modulated by the position of the user's finger on the screen of a mobile phone when entering a pin [152]. The latter [152] is particularly interesting because it arises from the interaction of the user with a wireless transceiver.

In general, TEMPEST attacks passively listen to the natural emissions of a victim device. An alternative is the Radio Frequency Retroflector Attack (RFRA) attack. In this case, the attacker adds a malicious hardware component to the communication interface to monitor, for example, a video cable or a digital communication line. This component acts as a backscattering antenna that reflects a modulated version of an external carrier. The NSA Advanced Network Technology (ANT) catalog [181] includes a continuous wave radar to illuminate the target and receive the reflection, and several retroflectors for different cases. Retroflectors have been studied in public literature as well [286]. The study reports successful attacks up to 10 m, inserting a transistor in a coaxial cable or a USB cable, and using an SDR for transmission and reception. 'The Thing' [272] is another curious example of attack device, which was used for spying the conversations inside an embassy. It consists in a resonant cavity microphone that reflects an external RF carrier, modulated with the audio signal that makes the cavity vibrate.

Experiments in the Electromagnetic Compatibility (EMC)/RFI [79, 145] and security [28] communities suggest that a communication line between two logic gates acts as a backscattering antenna that reflects an external carrier, intermodulated with the data transmitted on the line. This suggests that adding hardware retroflectors might not always be necessary. Similarly, the unintentional modulation of an external carrier with sensitive data is mentioned under the name of NONSTOP in some declassified documents [8] and discussed by public literature [11, 139].

3.2.2.2 *Exfiltrating Data at a Distance*

In a different scenario than TEMPEST, an attacker with access to a device might want to exfiltrate sensitive data without using any network interface. This might be because the device is air-gapped (disconnected from public networks and placed in secured facilities), because unprivileged software does not have access to communication interfaces, or to avoid interception by network analysis tools. In this context, compromising emanations controlled by the attacker are useful to build a wireless covert channel.

In 1998, the idea of modulating TEMPEST leakages from screens to exfiltrate data from a victim was proposed in [11, 149], under the name of Soft-TEMPEST. Here, the ability to control TEMPEST leakages was also proposed as a defense mechanism, in which a TEMPEST-aware video driver would chose fonts that minimize the TEMPEST signal. The name Soft-TEMPEST comes from the fact that the compromising emanations are controlled from software, also called TEMPEST virus [149].

Over the years, literature has explored a large number of covert channels based on software-controlled emanations of several physical quantities [42, 108–118, 120–123, 125, 170, 243, 246, 261, 303]. We refer the reader to [41, 43] for a general survey of covert channels.

Like TEMPEST, Soft-TEMPEST attacks also generally work at a distance from the devices, without physical access, and through walls. Magnetic leakages [112, 117, 170] are interesting because they can bypass EM shields designed for RF signals. However, in general modulation schemes are very simple (e.g., OOK, FSK) because the control that software has on the leakages is very limited. In some cases these covert channels do not use unintended emissions but explicit output interfaces (e.g., audio [42], reverse-engineered WiFi chips [234]). In this case more modulation options become available, but the attack also requires higher privileges and it cannot be considered an unintended leakage anymore.

As highlighted in Chapter 1 Second-Order Soft-TEMPEST attacks proposed in [55, 69] leverage cascaded interactions with other signals in the system to exfiltrate data, for example, as a polyglot modulation on top of WiFi.

Backscattering of an external carrier is a good option also for data exfiltration, for example leveraging the impedance of a WiFi front-end, which depends on the card status [295, 296].

3.2.2.3 *Detecting Electronic Devices at a Distance*

In general, radio transmitters can be naturally detected, located, and fingerprinted using their intended emissions (e.g.,[60, 285]). Instead, radio receivers and other electronic devices do not emit any intended signal. However, their unintended leakages may reveal their position. This is very useful as a defense against passive attackers sniffing some communication or electronic devices used to control explosives, to cite a few examples.

Often, a radio receiver leaks an RF signal coming from the Local Oscillator (LO). For example, the LO leakage from WiFi cards in monitor mode can be detected in many commodity devices [45] at up to 5 m of distance. Another interesting active approach consists in illuminating a receiver with a chirp signal that falls in the reception channel, and then looking for the re-emission of a down-converted version of this chirp [259]. This methods was shown to work at several tens of meters using two commercial receivers.

A more detailed discussion of this topic is beyond the scope of this thesis.

3.2.2.4 *Side Channel Attacks Against Cryptography in Proximity*

As we have explained in Chapter 2, side channel attacks are a specific type of attacks against the implementation of cryptographic algorithms. Compromising emanations from cryptographic software or hardware might act as side channel vectors. The signals exploited by side channel attacks are usually weaker than those previously described for TEMPEST and Soft-TEMPEST. As a result, physical side channels are usually considered as attacks in close proximity that require physical access to the device. However, a few exceptions exist that can reach distances in the order of a few meters.

Power side channels [143] that measure the current consumed by Complementary Metal Oxide Semiconductor (CMOS) logic generally require physical access to insert a current probe. Acoustic side channel attacks, which measure the sound produced by electronic components, were demonstrated against public key cryptography (RSA) at up to 4 m [86]. We refer the reader to [86] for a review of other acoustic attacks.

In the early 2000's, public literature began to investigate electromagnetic emissions as a side channel vector [7, 84, 213]. In particular, [7] distinguishes between direct emanations from electronic components, and the unintended modulation of other carrier sources, such as the harmonics of the clock signal, due to unintentional coupling between blocks.

EM attacks usually require placing a near-field probe close to the victim, but some previous work has also demonstrated attacks at larger distance [87, 88, 174, 219]. In particular, [219] shows successful attacks against AES running on an Arm Cortex-M3 processor, at 30 cm in a laboratory environment, and at 1 m in an anechoic chamber.

Some previous work has also studied the effect of distance systematically, coming to interesting conclusions. A study of correlation attacks against AES in [174] shows that the leakage model is more and more distorted as distance increases up to 0.5 m. The degradation of the signal with distance is discussed in [7] for 2048-bit exponentiation modulated onto a harmonic of the clock. In this case the far-field leakage is visible up to 12 m. Discussing the leakage at 4.5 m, the paper mentions the possibility to run template attacks. The near-field and far-field leakages of a software pattern are analyzed in [300]. Far-field emissions follow a quadratic loss and are visible at up to 3 m. The software pattern consists of two different instructions repeated in a loop. Therefore this is not a direct study of the feasibility of Correlation-based Electromagnetic Analysis (CEMA) attacks against symmetric cryptography.

As discussed in Chapter 1, active attacks leveraging parasitic load modulation in NFC tags and parasitic load modulation in RFID tags have been demonstrated at a distance of up to 1 m.

Some previous work, inspired by NONSTOP, demonstrated some active attacks in which an external RF is injected in the victim circuit. In [28], a carrier is injected in a smart card with a current transformer on the ground, and it is modulated by a loop or by the DES module. In [164], a carrier injected in the power supply by conduction modulates the emissions of a clock generator in frequency, and up-converts a data-dependent timing leak from code execution on a secure microcontroller.

Timing side channels do not necessarily require physical access. For example, timing side channels over the air against a WiFi access point have been demonstrated in [280]. Other types of side channel attacks that can be performed remotely go beyond the scope of this thesis. In general, they require access to a remote system, where victim and attacker share some resources, breaking their logical isolation. They include two main categories. First, microarchitectural attacks (e.g.,[106, 276]) that exploit timing measurements of the shared resources used by the victim (e.g., the cache). Second, attacks in which access to a remote 'sensor' lets the attacker measure the physical emissions (e.g., power consumption) of the victim on the same remote device. They usually require access to custom logic on an FPGA [90, 92, 218, 230, 306], but they were also demonstrated on a System on a Chip (SoC) using delay lines [101] or an ADC [94, 196].

3.2.2.5 *Fingerprinting, Malware and Trojan Detection in Proximity*

The use of EM compromising emanations has also been proposed for applications such as fingerprinting USB keys [133], and detecting malware in embedded systems (e.g.[140]). Active side channels based on the backscattering of an external carrier have been proposed to detect dormant hardware trojans [187], and to build an RFID tag with an FPGA [47]. A study of backscattering from an FPGA illuminated with a 300 GHz signal at a distance of 28 cm appears in [4]. A more detailed review of these applications goes beyond the scope of this thesis.

3.3 SUMMARY

In this chapter we have discussed relevant attacks in classic wireless security (intended wireless transmissions) and in EmSec (unintended emissions).

We have seen that side channel attacks can break the cryptographic protections that defend wireless protocols. However, differently from most of the other wireless and EmSec attacks that can be performed over the air from a large distance, side channel attacks usually require physical proximity. This limits their applicability and threat to communication devices. Interestingly, some unexpected side and covert channels arise from the presence of radio front-ends in proximity to other digital blocks.

We have also seen that compromising emanations can be modulated in order to transmit data. However, modulation schemes and applications are quite limited. Indeed software has little control on the leakages.

Part III

SCREAMING CHANNELS

We conjecture that the radio transmitter on a mixed-signal chip might accidentally leak information about the activity of the digital blocks on the same device, making side channel attacks possible from a large distance. Since the radio intentionally emits a strong signal, we call this novel side channel vector '*Screaming Channels*', in contrast to the weak 'whisper' of conventional electromagnetic leakages. After having demonstrated the existence of *Screaming Channels* on some devices, we focus on a Bluetooth Low Energy chip for an in-depth analysis and challenging attacks.

4 DISCOVERY

In this chapter we describe our hypothesis on the existence of different types of *Screaming Channels*, some examples that we observed empirically, and two preliminary attacks that confirm our intuition.

4.1 HYPOTHESIS

4.1.1 The Idea

4.1.1.1 *Preliminaries*

Modern devices need both computation and wireless communication capabilities. In general, digital computation blocks on an electronic platform are strong sources of deterministic noise, that can be characterized starting from the properties of the digital signals. For example, the noise produced by a clock signal consists of several narrow-band peaks at certain harmonics, which can be predicted from the frequency, duty-cycle, and other parameters of the clock square wave [255]. Digital noise is therefore fundamentally different from the random noise due to the inherent nature of physical components (e.g. thermal noise), which characterizes analog designs [5]. Traditionally, radio systems were designed considering only this Gaussian noise, whereas digital systems were designed without any consideration of the impact of their digital noise on other radio blocks, setting the ground for the problematic coexistence of computation and communication in modern systems [255]. In mixed-signal chips, the problem is exacerbated by the close physical proximity between digital and radio components on the same silicon die, which results in many noise coupling paths between the two. For example, noise can flow through the substrate or the power supply. Figure 4.1 shows an example of mixed-signal chip. Previous work (e.g., [5, 27]) has studied this problem and possible solutions from a functional point of view. We refer the reader to specialized literature for more details about mixed-signal chip design, which are outside the scope of this thesis.

4.1.1.2 *Intuition*

Digital noise is related to the logical activity of the device. A classic example is the power consumption of a CMOS inverter, which depends on the logic transitions, as shown in Figure 4.2. The two CMOS transistors act as switches that connect the output to the power supply (high)

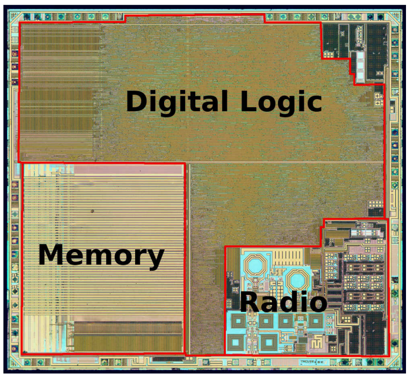


Figure 4.1: Example of mixed-signal chip. Digital and radio blocks are clearly visible on the same silicon die. "nRF51822 - Bluetooth LE SoC : weekend die-shot" CC BY 3.0 by Zeptobars [302], modified with annotations.

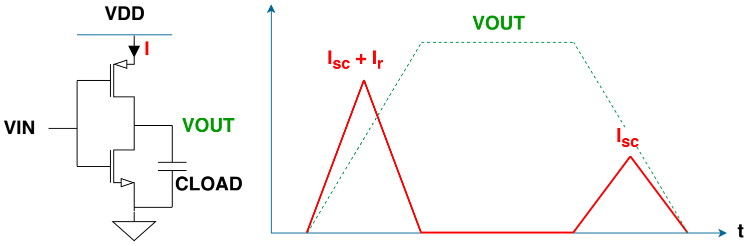


Figure 4.2: Digital noise is related to the logic activity of the gates. For example, the current sourced from the power supply to a CMOS inverter depends on the direction of the transition of the logic value, whereas no current flows if the logic value is constant.

or the ground (low). When switching from low to high, the power supply sources a spike of current I_r to charge the parasitic capacitance of the output. In contrast, when switching from low to high, the parasitic capacitance discharges to ground. In both cases, for a short instant during the transition both transistors conduct, and a short circuit current spike I_{sc} flows from the power supply to ground. As seen from the power supply, a rising edge requires $I_r + I_{sc}$, whereas a falling edge requires I_{sc}, and no current is consumed when no transition occurs.

From a security point of view, this is a dangerous characteristic, because it makes digital noise a potential vector for leaks of sensitive information. TEMPEST and electromagnetic side channel attacks are prominent examples of this problem. We know that digital noise can affect a radio transmission, for example, increasing phase noise or producing spurious frequency components [5]. Possible examples of coupling paths between digital blocks and the components of a direct conversion transmitter are shown in Figure 4.3. These problems cannot be completely removed, and they are just reduced to the point that the protocol requirements are met and the regulations on emissions

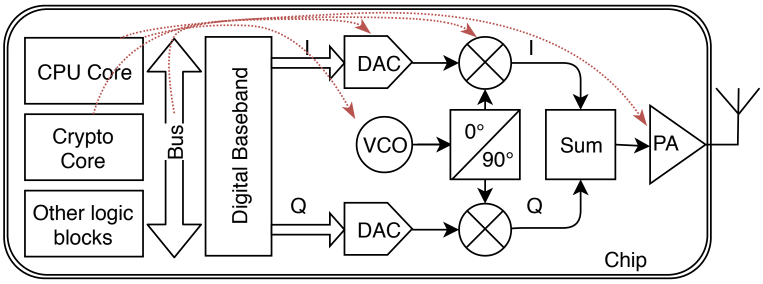


Figure 4.3: A mixed-signal chip with digital blocks close to a direct conversion radio transmitter. Lines in red depict possible noise coupling paths between the digital part and the main components of the radio (e.g., VCO, PA).

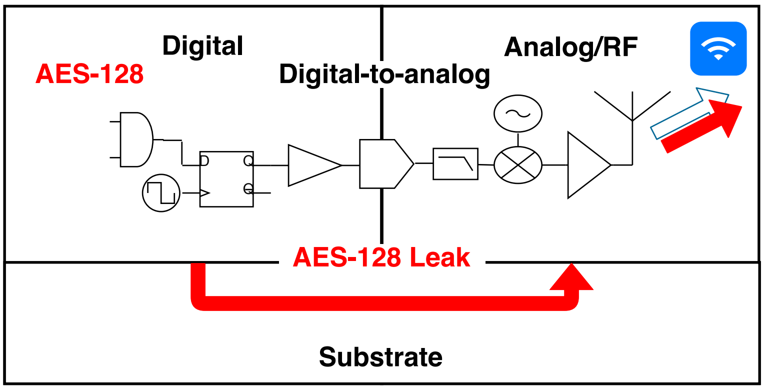


Figure 4.4: An example of *Screaming Channels*: AES running on a mixed-signal chip produces a leakage that accidentally flows from the digital blocks to the radio transmitter. Here, it is picked-up, up-converted, amplified, and broadcast at large distance together with the intended data.

are respected. We conjecture that the effect of digital noise on radio transmissions could be strong enough to constitute a side channel carrying sensitive information about the logic activity of the device, which could be passively measured over the air at large distance from the target. In other words, we conjecture that the radio transmitter in a mixed-signal chip might pick-up, up-convert, amplify, and broadcast a side channel leakage coming from the digital blocks on the same device, because of the unintended noise coupling between digital and radio components. Figure 4.4 depicts an example were the leakage from the execution of AES is broadcast, making side channel attacks possible at large distance.

4.1.2 Refinement

After having formulated the initial hypothesis, we refine our analysis of the problem, in order to plan the experiments for empirical validation.

4.1.2.1 *Definition of A Scenario*

The existence of *Screaming Channels* opens novel attack opportunities. From a system security point of view, we imagine the following scenario.

- A smart device (e.g., smart watch, smart lock, beacon) uses a mixed-signal chip to integrate computation and communication capabilities.
- Interactions over the wireless link are protected by cryptography at the application layer, independently from the underlying wireless protocol.
- Application layer cryptography is likely to run in software (e.g., for portability), and software usually produces stronger leakages than dedicated hardware blocks.
- Protections against side channel attacks are unlikely to be present. Indeed, conventional side channel attacks that require physical proximity are usually outside the threat model for these simple devices, not designed for security applications. Moreover, conventional side channel countermeasures might be too expensive considering strong cost constraints.
- Because of *Screaming Channels*, side channel attacks become possible from a large distance, without physical access. The attacker can passively measure side channel traces over the air. Possibly, the attacker can interact with the device over the air to trigger the execution of application layer cryptography.

For simplicity, we focus on this scenario. In particular, we decide to demonstrate *Screaming Channels* with correlation attacks against AES [211] encryptions with known plaintext. We will see that an attack against a real world application (Google Eddystone beacons authentication) falls in this case. Other scenarios include attacking the link layer encryption of a specific protocol, or public key cryptography used for paring devices.

4.1.2.2 *Definition of the Signal to Observe*

The definition of a scenario lets us define what signal we expect to find on the radio channel. Note that what we call signal from the attackers' point of view is noise for the radio designers. This is a fundamental step to plan the experimental validation of our intuition. Even though it is not necessarily the only option, it is a realistic case, representative of attacks against real systems. As we have seen in Chapter 2, a successful CEMA against AES requires collecting many AES traces, with different known plaintext and key for profiling, and unknown key for the attack. In this case, we define two different signals. The first one is the AES trace. If the execution of software with different instructions and loops modulates the radio transmission, then we are able to identify a single AES encryption from other operations, extract it as a time trace, and align it with other traces. Since we perform the attack over the air, We do not have any trigger signal to identify the beginning of a trace. Therefore, we need to be able to identify the AES signal itself from noise. Note that in our scenario encryptions are not likely to be synchronized with packets, so that we cannot use the beginning of a packet as a source of synchronization. This is the worst case for the attacker. The second type of signal is the data-dependent difference among traces at certain time instants (POIs). This is what is usually called side channel signal, and it is very small compared to the Gaussian noise that affects the measurement of the traces. If the AES algorithm and the underlying hardware are vulnerable, a statistical analysis of the traces will reveal the key. In summary, we plan to observe from a large distance on the radio channel:

- An effect of software execution strong enough to allow the extraction and alignment of AES traces corresponding to AES encryptions. A simplified condition consists in distinguishing simple software patterns, such as loops of different sizes.

- A data dependency strong enough to allow the extraction of the secret key with side channel attacks.

4.1.2.3 *Definition of a Simplified Channel*

Conventional EM traces leak from the device where they are measured with a near-field probe. In contrast, *screaming-channel* traces flow from the digital blocks to the radio transmitter, and then they are broadcast over the air together with the intended data. The coexistence of side channel traces and intended data creates at least two main problems:

- Side channel traces and intended data are modulated onto the same carrier signal. Trace extraction would require receiving the intended packets and then measuring the traces as an error on top of the ideal signals. In case leakage and intended data use orthogonal modulations, extraction becomes trivial as it is independent from packet reception.

- Side channel traces are visible only when a packet is being transmitted. Since packets are sent at discrete instants of time, not synchronous with the encryption traces, extraction becomes more difficult. Only some of the traces would fall in a packet and be visible. In addition, some protocols (e.g., BLE) use frequency hopping, so the frequency of the channel changes over time.

Because of the above considerations, we use a simplified setup for testing the existence of *Screaming Channels* on real hardware:

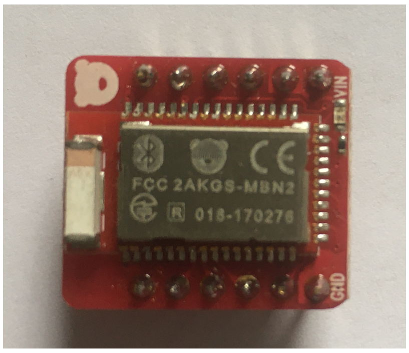


Figure 4.5: *BLE Nano v2.*

- Software runs AES encryptions or other simple patterns on the CPU of the target.
- The transmitter of the target is set to transmit a continuous unmodulated wave. This option is often present as a test mode for the hardware. Unfortunately, the firmware required to activate it is not always available.
- Attacker and target are possibly connected via cable, to reduce channel noise as much as possible. Alternatively, experiments are run in an anechoic chamber, for the same purpose.

4.2 EXPERIMENTAL VALIDATION

4.2.1 *Nordic Semiconductor nRF52832*

4.2.1.1 *The Chip*

The *Nordic Semiconductor nRF52832* [191] is a mixed-signal chip featuring an Arm Cortex-M4 processor, NFC, and a 2.4 GHz transceiver supporting BLE [228], ANT/ANT+ [2], Gazel™ [242], and Enhanced ShockBurst™ [241]. It also has other peripherals, including a dedicated block for hardware AES encryption. The *nRF52832* is used in a wide range of real-world products, including smart locks and BLE beacons, sold in large quantities. For this reason, we believe it is a relevant target for research. In addition, the excellent documentation, paired with an open source Software Development Kit (SDK) and several examples of applications, facilitates experimentation. In this thesis we focus on an off-the-shelf BLE dongle (*BLE Nano v2* [201, 221] in Figure 4.5), and two development boards (*PCA10040* [189], *Rigado BDM301* [223]).

4.2.1.2 *Firmware*

We program the chip with the *Nordic Semiconductor nRF5* [190] SDK. Starting from an example for radio testing, we develop our custom

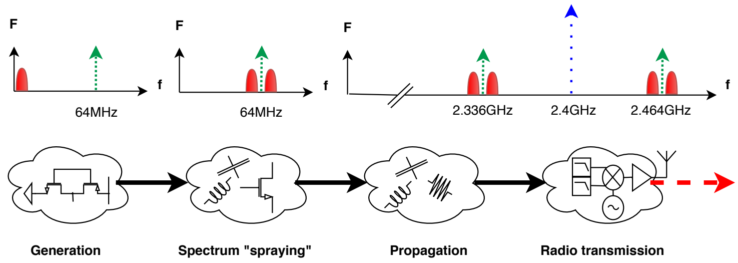


Figure 4.6: An example of *Screaming Channels* in practice on a *Nordic Semiconductor nRF52832* chip. Software running on the Arm Cortex-M4 processor modulates the harmonics of the 64 MHz clock. At their turn, these harmonics are intermodulated with the radio carrier at 2.4 GHz. As a result, AES traces are clearly visible on the radio channel at high power.

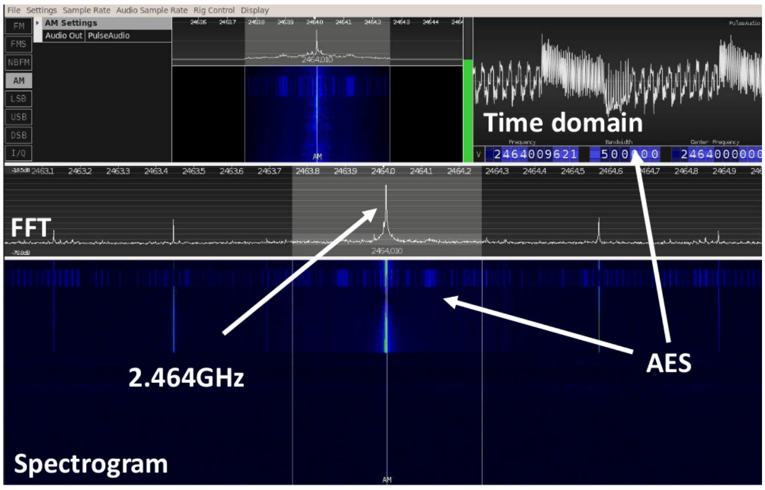


Figure 4.7: If we tune a radio receiver at 2.464 GHz we clearly see a frequency component modulated by the execution of AES. Modulation is visible in the time domain (AM demodulation), frequency domain (FFT) and time-frequency domain (spectrogram).

firmware for *Screaming Channels* experimentation. Our test program turns on and off a continuous wave on one of the BLE channels. In particular, for the initial tests we transmit at 2.4 GHz at a power of 4 dBm. In parallel, the firmware runs simple patterns, such as loop at different frequencies, chirps, and AES encryptions. We use the `tinyAES` and `mbedTLS` implementations of *AES-128*, or the dedicated hardware block.

4.2.1.3 *Software Modulation*

We observe that software execution modulates the radio transmission in two steps, as depicted in Figure 4.6. First, the leakage modulates the harmonics of the clock at multiples of 64 MHz. This is a common type of conventional side channel [7]. Then, many of these harmonics are intermodulated with the radio carrier at 2.4 GHz, and its harmonics.

Finally, the carrier is amplified and broadcast by the radio front-end. We can draw a parallel with the superheterodyne radio transmitter, in which the baseband signal is first modulated onto an IF carrier (the 64 MHz clock in this case) and then onto the RF carrier (the 2.4 GHz carrier in this case). As a result, software AES traces are visible on the radio channel at high power, as shown in Figure 4.7. This is a first experimental validation of our hypothesis on the existence of *Screaming Channels*. We confirm that the leakage at radio frequency is the product of intermodulation with the carrier by changing the channel frequency. Overall, we conclude that the original baseband leakage is visible at radio frequency at:

$$f_{RF} = p \cdot f_{carrier} \pm q \cdot f_{clock} \tag{4.1}$$

where p, q are positive integers. To be precise, we also observed the direct modulation of the radio carrier. However, this case is hard to measure with good accuracy, because small variations are superimposed on a strong signal that occupies most of the dynamic range of the receiver. The active suppression of the carrier in hardware would facilitate the measurement, but we leave this for future work.

4.2.1.4 *Hypothesis About The Coupling Path*

Without any access to the internal design of the chip, we can only conjecture which is the reason for the coupling between CPU and radio transmitter. After responsible disclosure and discussion with the manufacturer, we believe that a reasonable hypothesis is the coupling to the power amplifier via the power supply. This explanation matches our observation that the leakage is cross-modulated (in amplitude) on the clock and then on the carrier. Note that this is not a major problem for the normal functioning of BLE. Indeed, the GFSK modulation used by BLE is orthogonal to (and not much affected by) amplitude modulation. Conversely, *screaming-channel* traces would still appear in the same way if using BLE packets instead of a continuous wave.

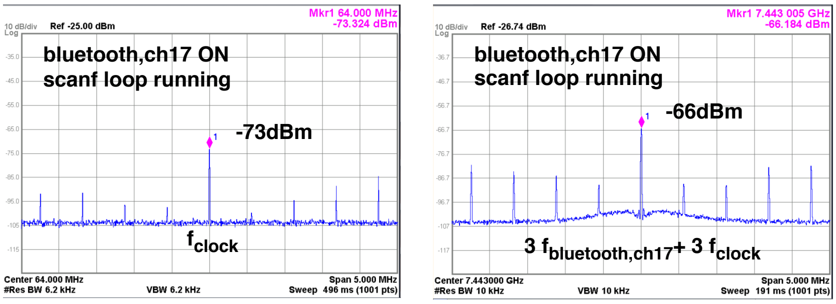
4.2.2 *Nordic Semiconductor nRF52840*

4.2.2.1 *The Chip*

The *Nordic Semiconductor nRF52840* [192] is a mixed-signal chip similar to the *nRF52832*. We run similar experiments on the development kit.

4.2.2.2 *Software Modulation*

Using the same firmware as for the *nRF52832*, we observed a very similar behavior. Figure 4.8 shows an example with $f_{carrier} = 2.417$ GHz, corresponding to BLE channel 17. The same leakage, corresponding to a loop, is visible at clock frequency $f_{clock} = 64$ MHz and at radio frequency $3f_{carrier} + 3f_{clock} = 7.443$ GHz. In both cases the leakage



(a) Leakage at 64 MHz.

(b) Leakage at 7.443 GHz.

Figure 4.8: Also on the *Nordic Semiconductor nRF52840* the leakage is visible on the harmonics of the clock (a), and on the harmonics of the clock intermodulated with the harmonics of the radio carrier (b).

changes when modifying the size of the loop, confirming that software modulates the radio transmission. Note that the power of the signal at 7.443 GHz is still higher than the conventional leakage at 64 MHz, despite being on the third harmonic of the radio channel and the third harmonic of the clock. More experiments with WiFi chips are discussed in Appendix a.1.

4.3 COMPLETE ATTACKS AGAINST THE *nrf52832*

From the previous analysis of several chips, we observe that the *nRF52832* is an ideal target for *screaming-channel* attacks. Indeed, AES traces are clearly visible on the radio channel. We now develop a few simple side channel attacks against different implementations of AES running on a *BLE Nano v2*, up to a distance of 10 m in an anechoic chamber. This completes the empirical validation of the existence of *Screaming Channels* that make side channel attacks possible at a large distance.

4.3.1 Trace Extraction

We run `tinyAES` or `mbedTLS` encryptions, compiled without optimizations, on a *BLE Nano v2*. We tune our radio receiver at $f_{RF} = f_{chan} + 2f_{clk} = 2.528$ GHz with a sampling rate of 5 MHz, and we use quadrature amplitude demodulation to recover a stream of AES traces corresponding to the encryptions running on the processor. Inspired by [308], we use a specific frequency component in the AES traces as a trigger to detect the beginning of each trace. With this method, physical access to a trigger pin on the device is not required. Once single encryption traces are extracted, we align them to a common reference in a fine-grained way using cross-correlation.

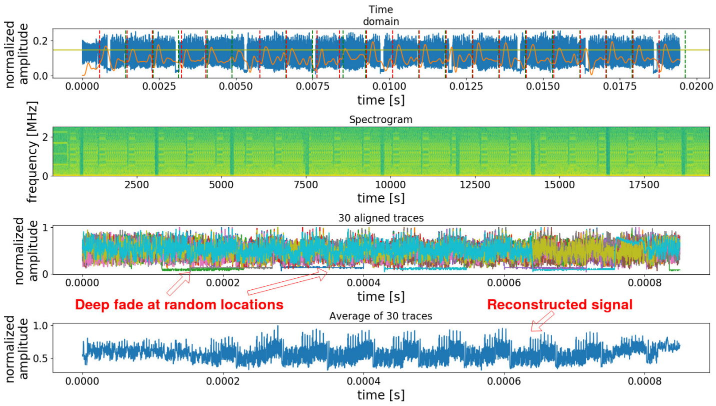


Figure 4.9: *Screaming Channels* trace extraction. AES traces are detected using a specific frequency component as trigger, and then aligned using cross-correlation. Multiple traces corresponding to the same encryption are averaged together as a form of time diversity.

4.3.2 Modulated Packets

BLE uses GFSK modulation, which is orthogonal to the amplitude modulated AES traces that we extract. Therefore, in order to make the attack more realistic, we can easily replace the continuous wave of the test mode with the transmission of discrete packets. Thanks to the orthogonality, the demodulation and extraction system works transparently. Packets are configured to have a random address and a random 254-byte payload, resulting in a duration of ≈2.1 ms, followed by a period of ≈120 µs during which the radio transmitter is off. `tinyAES` encryptions are ≈820 µs long, spaced by ≈48 µs. To remain close to a realistic scenario, encryptions are not synchronous with the packets. As a result, from the point of view of the leakage, the channel is intermittently off. The traces collected by the attacker might therefore contain gaps of ≈120 µs, corresponding to the periods when the radio is not on. This phenomenon, that we can interpret as deep fade, is a distinctive characteristic of *screaming-channel* leakages compared to conventional ones. We use a preprocessing technique consisting in averaging many traces corresponding to the same encryption. In this context, it can be interpreted as a form of time diversity to reconstruct a full trace despite random holes in the signal. The full extraction procedure is visible in Figure 4.9.

4.3.3 `tinyAES` at 10 m in an Anechoic Chamber

In an anechoic chamber [200, 214] we place the *nRF52832*, and our reception hardware, at a distance of 10 m, as shown in Figure 4.10. We use a *Ettus Research USRP N210* [70] SDR, with two *Minicircuits ZEL-1724LN* [175] amplifiers, and a *TP-LINK TL-ANT2424B* [266] directional

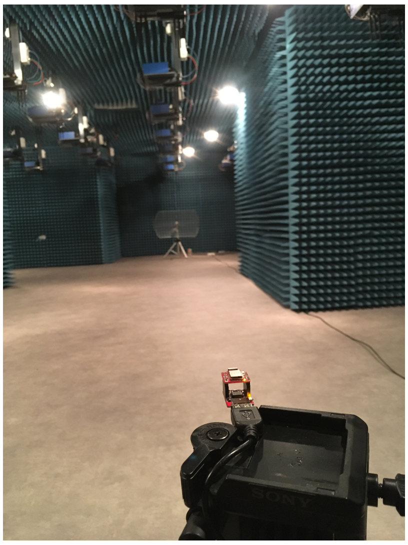


Figure 4.10: *Screaming-channel* attack at 10 m in an anechoic chamber.

WiFi antenna. We collect $70000 \cdot 500$ profiling traces and $1428 \cdot 500$ attack traces, where 500 is the number of traces that we average for each encryption with a given plaintext and key. Using a simple template attack, initially based on a modified version of the code from the ChipWhisperer project [194], we can fully recover the secret key.

4.3.4 mbedTLS at 1 m in a Home Environment

We also show an attack against mbedTLS, a widely used TLS implementation. Although mbedTLS cares about timing side channels that can be mounted remotely, power side channels that require physical access are outside the threat model.[1] However, with *Screaming Channels* mbedTLS can be attacked without requiring physical proximity. In a home environment, we place the *nRF52832*, and our reception hardware, at a distance of 1 m. For simplicity, we set the *nRF52832* to transmit a continuous wave, and we add a short preamble to facilitate the detection of mbedTLS traces. We succeed at full key recovery, using a correlation attack, initially based on a modified version of the code from the ChipWhisperer project [194]. The number of traces required for the attack is a bit less than $40000 \cdot 500$.

4.4 CONCLUSION

In this chapter we have demonstrated the existence of a novel side channel vector that might appear in mixed signal chips where the radio

[1] See for example this questions on the mbedTLS discussion forum and the response by the main mbedTLS developer: https://tls.mbed.org/discussions/crypto-and-ssl/aes-implementation-resistant-to-side-channel-analysis-attacks

transmitter accidentally broadcasts a leakage from the digital blocks. This makes side channel attacks possible from a large distance, posing a considerable threat.[2] We have refined our initial intuition, specifying a threat model, and a way to validate our idea with experiments on actual devices. Our experiments have shown that several chips have some *screaming-channel* effects. In particular, the *Nordic Semiconductor nRF52832* chip appeared as an ideal target to develop attacks and further research. We have demonstrated that attacks at large distance are possible, for example, at 10 m in an anechoic chamber. However, several questions about *Screaming Channels* are still open. In Chapter 5 we will conduct an in-depth systematic analysis of *screaming-channel* attacks against the *nRF52832* BLE chip. See Appendix a.1 for a discussion about *Screaming Channels* on WiFi transmissions.

[2] Interestingly, a TEMPEST document [182] declassified in 2000 mentions the modulation of an intended signal among the possible propagation methods of secret information, but details are redacted. We could speculate that these redacted lines reflect the interest for effects similar to *Screaming Channels*, where intended transmissions accidentally carry other sensitive signals.

5 ANALYSIS

In Chapter 4 we have demonstrated the existence of *Screaming Channels*, and some initial attacks. However, numerous questions are still open about the nature and properties of *screaming-channel* leakages compared to conventional side channels. A better understanding of the novel vector is necessary for both defensive and offensive research. In this chapter, taken from [35], we conduct a thorough investigation of the leakage channel, supported by extensive experimentation. We focus on the *Nordic Semiconductor nRF52832* chip, transmitting BLE packets and running `tinyAES` encryptions without optimizations. In this phase, we do not aim at mounting successful and realistic attacks, but at uncovering the characteristics of the leakage. Nevertheless, we often use attacks as an analysis tool.

5.1 OPEN QUESTIONS ABOUT AN UNEXPLORED CHANNEL

One of the prominent features of *Screaming Channels* is that attacks are possible at large distance, because the leakage is transmitted by a regular radio front-end. Although the advantages of this amplified transmission are intuitively clear, many aspects of the interaction between the leakage source and the radio channel are left unexplored.

5.1.1 Coexistence of Intended and Unintended Signals

The coexistence between the leakage signals and the signals for the intended radio communication is a first important factor to analyze. In the ideal case, the radio front-end continuously transmits a carrier, which is modulated by the leakage. Most of the experiments in Chapter 4, apart from the attack in the anechoic chamber, were conducted in these conditions. In reality, a modern radio protocol sends data in the form of digitally modulated packets at discrete time intervals. The (absence of) orthogonality between the intended digital modulation and the unintended analog modulation of the leakage has a huge impact on how the attacker can recover sensitive information from the radio signal. The radio channel is active intermittently, only when a packet is under transmission. The attacker has to pay great attention to the (absence of) synchronization between sensitive operations and radio packets. If the protocol uses FHSS, the packets regularly switch channel (frequency) following a pseudo-random pattern (possibly unknown to the attacker), further increasing the complexity of the reception.

5.1.2 From Digital Logic to the Radio Spectrum

Before being sent over the radio channel, the leakage generated by the digital logic has first to flow to the analog/RF part, modulate the carrier, and be amplified and radiated. On the contrary, conventional leakages emanate directly from the digital part. The additional steps taken by *screaming-channel* leakage involve going through filtering stages and nonlinear components. As a consequence, some of the original information might be lost in the process, countering the advantage of amplification and transmission. Similarly, the shape of the leakage traces might encounter considerable distortions compared to conventional ones. The attacker has to understand the impact of such distortions on the leakage model, and take it into account to maximize the information that can be extracted.

5.1.3 The Radio Link

Once upconverted and amplified, the leakage is radiated by the antenna and it propagates in space. On the one hand, we can imagine that the transmission properties of this leakage are rather similar to those of the intended packets. For example, being in the same frequency range, they should have the same type of attenuation and reflection. On the other hand, the leakage has some unique properties. First of all, it might not have the same modulation scheme as the packets, resulting in worse resilience to noise. Second, despite the powerful amplification stage, the leakage is considerably weaker than the packets, as the input signal is itself weak. Finally, the frequency and bandwidth of the leakage might differ from those of the packets, changing the requirements for reception, and potentially conflicting with other channels or protocols. It is important to model the radio channel of the leakage, and to understand its attenuation and the distortion it might introduce. The use of time, spatial, or frequency diversity is likely to have a great impact on *screaming-channel* leakages, since they travel on a long radio link in a possibly complex environment.

5.1.3.1 *Profile Reuse*

One of the main advantages of *screaming-channel* attacks is that they can be mounted from a considerable distance when compared to conventional EM side channels. In this case, the attacker is not likely to have access to the target device. At the same time, profiling the

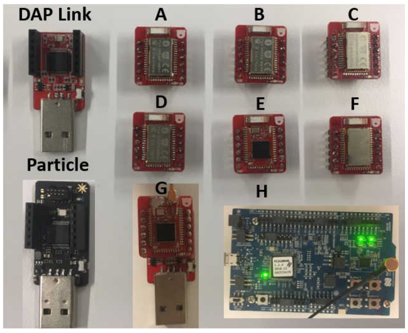


Figure 5.1: Several instances of the device under analysis/attack.

target device at large distance and in challenging conditions might be complex and require a large number of traces. It is therefore necessary to evaluate whether profiles built in certain conditions for a certain device can be reused in different conditions for another instance of the device. As a result, the attacker should be able to take into account the effect of the setup, of distance, of channel frequency, and of the target device. This is potentially more complex than simply trying to reuse profiles on different instances but in the same conditions. Although previous work exists on inter-device and inter-setup variations, the effect of distance on profile reuse for *Screaming Channels* was never studied before. If possible, reusing a profile taken in convenient conditions on a different instance in more challenging conditions would be a huge advantage for the attacker.

For these reasons, from now on we will study several devices based on the *nRF52832* chip, shown in Figure 5.1: several instances of the *BLE Nano v2* [221] with the *Particle Debugger* [201] (*devices A, B, C, D, E, F*, and with the *DAPLink Board* (*devices* F_{DAP}, *G*), and a *PCA10040* [189] evaluation board (*device H*). We have removed the shield of *device E* to place a conventional probe, and we have modified *device G* to replace the antenna with a direct cable connection to the radio. Instead, *device H* has an antenna that can be bypassed with a switch to connect to the radio output with a cable.

5.2 COEXISTENCE OF INTENDED AND UNINTENDED SIGNALS

In Chapter 4, we highlighted that the noise from the digital side of a mixed-signal chip can affect the radio transmitter in many ways on several devices. However, we focused on the specific case of a Bluetooth 5 [228] capable chip, the *Nordic Semiconductor nRF52832* [221]. In the following, we discuss this scenario in detail, while also reasoning about other similar cases.

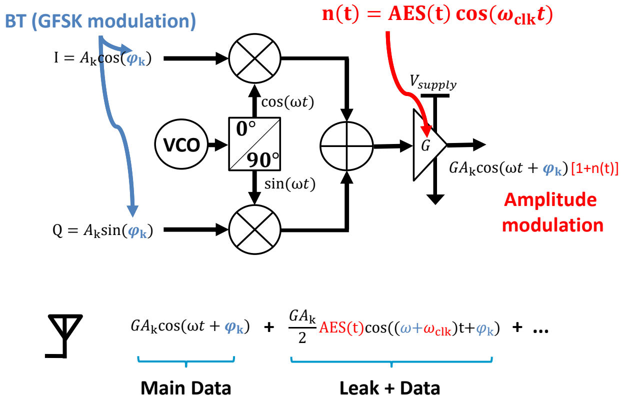

BT (GFSK modulation)
$n(t) = AES(t)\cos(\omega_{clk}t)$
$I = A_k\cos(\varphi_k)$
V_{supply}
$\cos(\omega t)$
VCO
0°
90°
G
$\sin(\omega t)$
$GA_k\cos(\omega t + \varphi_k)[1+n(t)]$
Amplitude modulation
$Q = A_k\sin(\varphi_k)$
$\underbrace{GA_k\cos(\omega t + \varphi_k)}_{\text{Main Data}} + \underbrace{\frac{GA_k}{2}AES(t)\cos((\omega+\omega_{clk})t+\varphi_k)}_{\text{Leak + Data}} + \ldots$

Figure 5.2: A simple model to explain the *screaming-channel* leakage on the *nRF52832*. The figure shows a generic direct conversion transmitter, in which BLE data are modulated in GFSK, whereas the coupling of digital blocks with the power supply causes the amplitude modulation of the carrier with the AES leakages.

5.2.1 Orthogonality of the Modulation

Figure 5.2 shows a simple model that explains the *screaming-channel* leakage present on the *nRF52832* chip. The leakage is modulated with analog AM on additional carriers that appear at multiples of the clock frequency from the center frequency of the radio channel (Equation 4.1). Intended data symbols are modulated with GFSK and transmitted at 4 dBm with a modulation rate of 1 Mbps, following the BLE specification. GFSK is a constant envelope modulation scheme, in which the two digital symbols 1 are 0 are represented as two different frequency offsets of the carrier from its center frequency. Differently from a simple BFSK, the transitions between symbols are smoothed with a Gaussian filter, in order to reduce the occupied bandwidth. Since AM is orthogonal to GFSK, the intended receiver will ignore the additional AM leakage trace, whereas the attacker will be able to ignore the intended GFSK data. On the one hand, this situation is very convenient for the attacker, since extracting leakage traces is easy with a simple Quadrature Amplitude Demodulator tuned at the BLE channel frequency plus a multiple of the clock frequency, as we have seen in Chapter 4. On the other hand, analog AM is by far less resistant to noise than GFSK, and the leakage does not propagate as well as the intended data. Note that GFSK is a popular choice for many wireless protocols other than BLE, including Bluetooth, Adaptive Network Topology (ANT)/ANT+ [2], Gazel™ [242], and Enhanced ShockBurst™ [241], which are all supported by our target chip. Figure 5.2 presents a simple model to visually explain the leakage on the *nRF52832* chip.

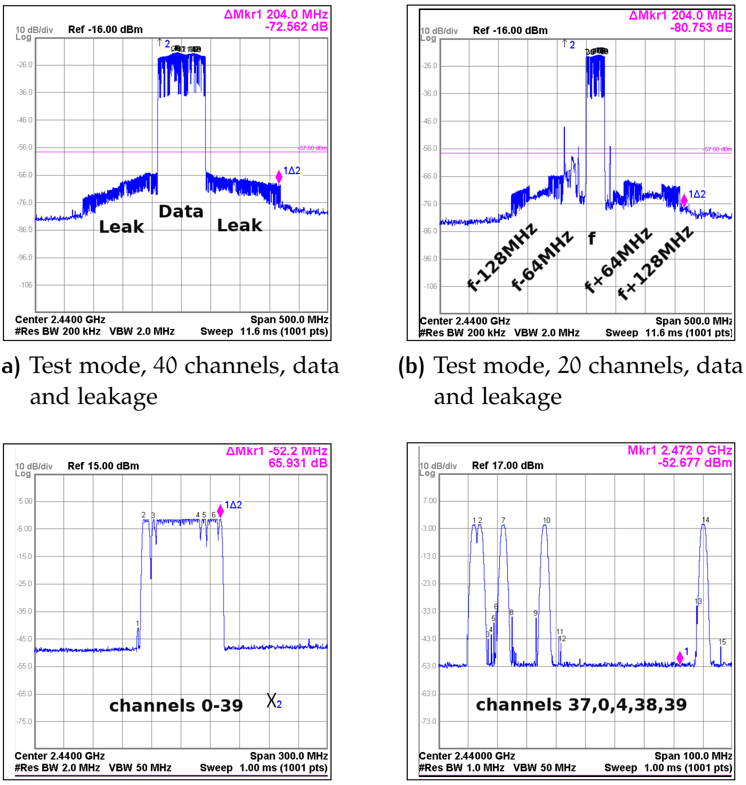
5.2.2 Discrete Packets as Deep Fade

As explained in Chapter 4, the radio channel is active intermittently (only when packets are being transmitted), and gaps between packets could be interpreted as a deep fade condition in the channel.

We can imagine specific cases in which the attacker might (at least partially) synchronize encryptions and packets. For example, the hardware block could be configured to encrypt packets on-the-fly during transmission, or the attacker could trigger an encryption just before a transmission. In this section, we are interested in studying the channel independently of specific attacks, and we use asynchronous packets and encryptions as a general case for our investigation. Therefore, we average many leakage traces corresponding to the same encryption, like explained in Chapter 4 and shown in Figure 4.9.

5.2.3 Frequency Hopping

Except for the real world attack in Chapter 6, in our experiments frequency hopping is disabled and the carrier is fixed at 2.4 GHz. We assume that a motivated attacker could build a receiver able to follow the hopping sequence, listen to all channels simultaneously with a wider band or multiple receivers, or listen at a fixed frequency and wait for the carrier to jump to that channel. The last case might be particularly reasonable for the advertising packets, which are sent over only three fixed channels. In addition, under certain conditions the attacker can reduce the number of channels used for hopping. A BLE device that initiates a connection can ask the peripheral to block up to 35 data channels out of 37. This is a standard feature of the BLE standard, used to reduce interference by adapting to the environment (adaptive frequency hopping). In practice it can be done by calling a function of the SDK, or by issuing a *Set Host Channel Classification Command* [228] to the BLE dongle. In Chapter 6 we show that initiating a connection and reducing the number of hopping channels is a reasonable assumption in a real-world attack scenario. An attacker not connected to the target could still try to jam some of the channels, hoping that another device that performs channel assessment will block them. Figure 5.3 shows the leakage in presence of hopping (in test mode) and the reduction of the hopping channels for a real connection. On the one side frequency hopping makes the attack more complex, on the other side the transmission of the leakage profits from the same advantages of frequency hopping as the regular data. For example, hopping helps reduce the impact of interference from other transmitters, and other narrow band signals in general. Other GFSK protocols might not use FHSS. For example, the ANT/ANT+ protocol avoids interference by dividing the channel into time slots. ANT+ devices might change channel at the application



(a) Test mode, 40 channels, data and leakage

(b) Test mode, 20 channels, data and leakage

(c) Real connection, 3 advertising and 37 data channels (default)

(d) Real connection, 3 advertising and 2 data channels (specified with the channel map)

Figure 5.3: To study hopping, we connect *device H* to a spectrum analyzer in maximum hold mode. Each time a packet is sent on a channel a spike appears at the corresponding frequency. In test mode we configure the device to sweep over the BLE channels, data and the leakage at the multiples of the clock are clearly visible (a)(b). In a real connection, the device uses all the 40 channels by default (c). We can reduce the channels used by a real connection to 5 by blocking the others (c).

layer if they detect that the current one is too crowded (Frequency Agility). Similarly, Enhanced ShockBurst™does not use FHSS, unless implemented at the application layer. Moreover, we observe that most protocols send an acknowledge packet at reception. An attacker can trigger a transmission (of an acknowledge packet) on a known channel by first sending a packet to the target device. To conclude, assuming a fixed channel is a reasonable choice for the study of the leakage vector.

5.2.4 Interference

The frequency of the leakage is the BLE channel frequency plus (or minus) a multiple of the 64 MHz clock (Equation 4.1). It is reasonable to assume that, for each channel, there exists a choice of the harmonic for which the leakage is less vulnerable to interference from other channels and protocols. For example, if we take the BLE channels from 2.402 GHz to 2.480 GHz, and the second harmonic of the clock, the leakage will span from 2.530 GHz to 2.608 GHz. In this range, which is well outside the ISM band, the leakage is less likely to be degraded by WiFi, BLE, and other 2.4 GHz protocols.

5.3 FROM DIGITAL LOGIC TO THE RADIO SPECTRUM

The path from digital logic to the radio spectrum through the radio front-end is a distinguishing feature of *screaming-channel* leakages. We want to investigate the effect of this path on the strength of the leakage and on its shape. In this phase, we keep the distance small and fixed. We normalize the traces, as described in more detail in Section 5.4.4.

5.3.1 Strength of the Leakage

We want to understand whether *screaming-channel* leakages are as strong as the conventional ones, or some loss appears when the leakages go through the radio transmitter.

5.3.1.1 *Datasets*

We are interested in the order of magnitude of the possible difference between *screaming-channel* and conventional leakages, for comparable setups and number of traces. To this purpose, we collect a set of $5000 \cdot 500$ profile traces and $1500 \cdot 500$ attack traces, for the three following cases:

- **Conventional (*device E*).** Traces are collected with a custom loop probe at the clock frequency (64 MHz). The loop probe is placed close to the power supply pin of a *BLE Nano v2* without metallic shield, where the signal is stronger. Further amplification with a ZKL-1R5 LNA is required.

- ***Screaming Channels* via Cable (*device G*).** Traces are collected at radio frequency (2.528 GHz), but using a custom cable connection between the radio and a *BLE Nano v2* modified for this purpose.

- ***Screaming Channels* at 10 cm (*device E*).** Traces are collected at radio frequency (2.528 GHz), using a standard WiFi antenna at a distance of 10 cm from a *BLE Nano v2*, in a home environment. The radio is a *HackRF*. Note that this case is representative of a simple realistic *screaming-channel* attack, and it is comparable with the conventional setting, but it not necessarily the optimal one.

For each case we carefully optimize the collection parameters. We use a HackRF radio and a sampling rate of 5 MHz.

5.3.1.2 *Comparable Strength*

We then evaluate the ρ-test. Results are shown in Figure 5.4. In the conventional setting, several POIs emerge for each byte of the key, and correlation is high. On the contrary, correlation decreases for the *screaming-channel* settings, and only one POI emerges for each byte. The correlation for the cable connection is lower, probably because the load for transmitter is not optimal (we replaced the antenna with a custom connection via cable). The correlation at 10 cm is on the same order of magnitude than the conventional one, though smaller. Better results for correlation and number of POIs can be achieved in better settings, as shown throughout this chapter (e.g., in Section 5.4.5 and Section 5.6.2).

To give a more practical metric, we compute the number of attack traces necessary to recover the key using profiled attacks and key enumeration, as shown in Table 5.1. Conventional attacks are more efficient because of the higher correlation and because of the higher number of available POIs, but they require physical access and external amplification.

5.3.1.3 *First-Order Leakage*

It is also interesting to compare profiled correlation attacks and univariate template attacks. As explained in Chapter 2, both profiled correlation and templates can capture the nonlinearity of the leakage model, thanks to the profiling step which estimates the model for every possible value of $p \oplus k$. However, template attacks can capture a second order relation between the leakage and the model, whereas profiled correlation attacks can only capture linear correlation. For

	Profile	max ρ, r_z	Attack	Type, POIs	Variable	Rank
Conv.	5k	0.95, 128	3	t.a.p.c., 11	$p \oplus k$	2^{18}
Conv.	5k	0.95, 128	5	t.a.p.c., 11	$p \oplus k$	0
Conv.	5k	0.95, 128	9	t.a.p.c., 1	$p \oplus k$	2^{16}
Conv.	5k	0.95, 128	9	p.c., 1	$p \oplus k$	2^{11}
Conv.	5k	0.95, 128	14	t.a.p.c., 1	$p \oplus k$	0
Conv.	5k	0.95, 128	14	p.c., 1	$p \oplus k$	0
10 cm	5k	0.79, 76	130	t.a.p.c., 1	$p \oplus k$	2^{21}
10 cm	5k	0.79, 76	130	p.c., 1	$p \oplus k$	2^{21}
10 cm	5k	0.79, 76	1273	t.a.p.c., 1	$p \oplus k$	2^{8}
10 cm	5k	0.79, 76	1273	p.c., 1	$p \oplus k$	0
Cable	5k	0.43, 32	1500	p.c., 1	$p \oplus k$	$> 2^{21}$

Table 5.1: Summary of the attacks in the conventional, *screaming-channel* via cable, and *screaming-channel* at 10 cm settings. All trace numbers should be multiplied by 500. *p.c.* stands for profiled correlation, and *t.a.p.c.* stands for template attack with pooled covariance. *Conv.* stands for conventional. The rank was computed with enumeration. The higher correlation and number of POIs result in better attacks for the conventional setting, as expected.

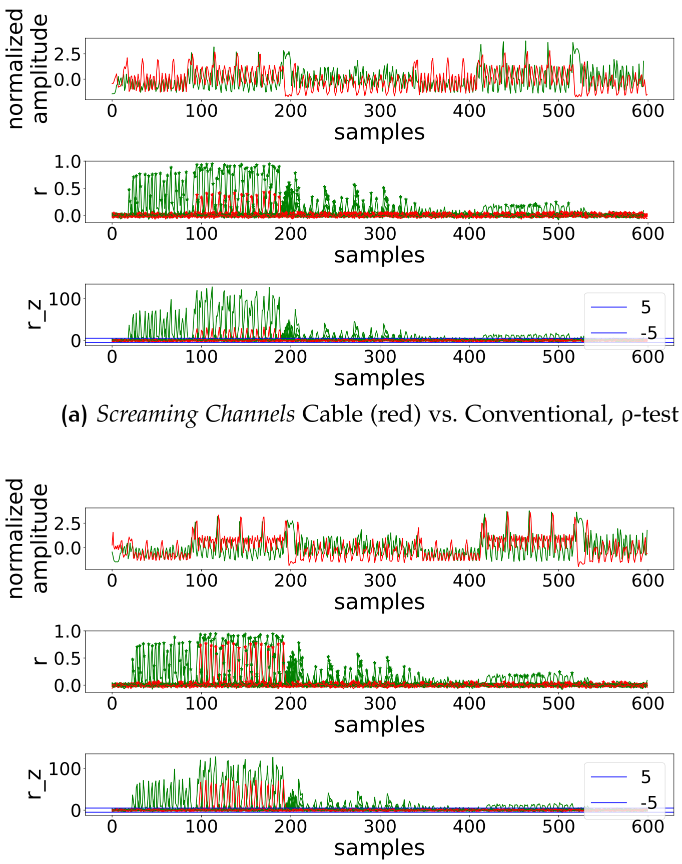
both the conventional and *screaming-channel* cases, template attacks are not more efficient. We can then assume that, for our sample size, correlation is good enough, and there is no significant advantage in using methods that can capture higher order statistics, such as templates. Additionally, profiled correlation attacks are computationally less expensive.

5.3.2 Distortion

We want to understand whether *screaming-channel* leakages follow the same leakage model as conventional ones, or are distorted on the way to the radio link. This is both interesting from a fundamental point of view, and useful to improve the attacks by choosing the best leakage model. We reuse the same data as for the strength of the leakage (Section 5.3.1).

5.3.2.1 *Conventional Leakages are Not Distorted*

The conventional leakage follows very closely a simple Hamming Weight model, and the highest POIs correspond to the output of the first Sbox. The ρ-test shows very similar correlation at the same point both for $p \oplus k$ and $HW[Sbox[p \oplus k]]$ (Figure 5.5). This confirms the point as the output of the Sbox, and it shows that a linear leakage model is a good assumption. As an additional check, we compute the non-



(a) *Screaming Channels* Cable (red) vs. Conventional, ρ-test

(b) *Screaming Channels* at 10 cm (red) vs. Conventional, ρ-test

Figure 5.4: Results of the ρ-test (average trace, correlation, confidence). Comparison between *screaming-channel* via cable and conventional (a), and between *screaming-channel* at 10 cm and conventional (b). Correlation peaks and number of POIs are higher for the conventional case.

profiled correlation $\hat{r}(L, Y)$ with $y = HW[Sbox[p \oplus k]]$, and we observe that a similar (negative) correlation peak appears: $\hat{r}(L, Y) = -0.9$, $-10\log(p) > 20$. This matches with the profile built before.

5.3.2.2 *Screaming-channel Leakages are Distorted*

On the contrary, the *screaming-channel* leakages have a distorted profile, both via cable and at 10 cm. In these cases, choosing $y = HW[Sbox[p \oplus k]]$ significantly reduces the correlation peaks of the ρ-test compared to templates built for $y = Sbox(p \oplus k)$ or $y = p \oplus k$, because the Hamming Weight model is not a good assumption. If we compare the profile built for $y = p \oplus k$ for the conventional and the *screaming-channel* case, they are not correlated due to the distortion of the latter. Instead, *screaming-channel* profiles for cable and 10 cm show a correlation of 0.59 ($-\log10(p) = 25$) between each other. This is clearly visible in Figure 5.6, together with the shape of the distorted profile in

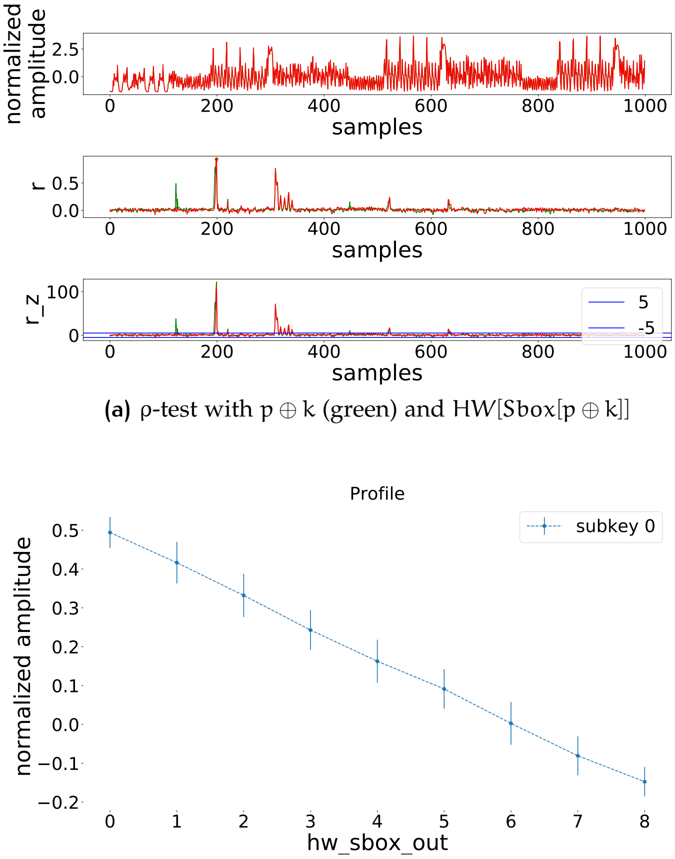
	Profile	max ρ, r_z	Attack	Type, POIs	Variable	Rank
Conv.	5k	0.93, 117	14	t.a.p.c., 1	HW	0
Conv.	5k	0.93, 117	14	p.c., 1	HW	0
Conv.	5k	0.95, 128	14	p.c., 1	$p \oplus k$	0
Conv.	5k	0.95, 128	14	t.a.p.c., 1	$p \oplus k$	0
10 cm	5k	0.20, 14	1273	t.a.p.c., 1	HW	$>2^{27}$
10 cm	5k	0.20, 14	1273	p.c., 1	HW	$>2^{27}$
10 cm	5k	0.79, 76	1273	t.a.p.c., 1	$p \oplus k$	2^8
10 cm	5k	0.79, 76	1273	p.c., 1	$p \oplus k$	0

Table 5.2: Attack results confirm that choosing $y = p \oplus k$ is important to take into account the distortion present in *screaming-channel* leakages, whereas it does not make any significant difference in the conventional case. *p.c.* stands for profiled correlation, whereas *t.a.p.c.* stands for template attack with pooled covariance. *Conv.* stands for conventional.

Figure 5.7. Results are similar for the other bytes of the key. Table 5.2 shows the impact on the attacks.

We further study this distortion using linear regression. In particular, we want to understand whether a linear combination of the bits of the output of the Sbox is enough to model the leakage better than the Hamming Weight, or if the leakage is non linear. First, we choose $y = \mathrm{Sbox}(p \oplus k)$, and we find the parameters A and B for which $\widehat{model}_{linear}(y) = A + B \cdot Y$ best fits the measured leakage L, where Y is the vector of the bits of y. While still linear, this model can take into account different weights for the bits, whereas the Hamming Weight model considers them all equal. Then, we remove the assumption on linearity, and we estimate the profile directly for each value of y: $\widehat{model}_{full}(y) = \bar{L}_y$ (as we do for the ρ-test). For the conventional case, the linear fit is a good approximation, and $\widehat{model}_{linear}$ has a correlation of 0.91 with $\widehat{model}_{full}$ ($-\log10(p) = 984$). For the *screaming-channel* case, the linear fit is not a good approximation, and the $\widehat{model}_{linear}$ has a correlation with $\widehat{model}_{full}$ of only 0.17 ($-\log10(p) = 22$). This is visible in Figure 5.8. Results are similar for the other bytes. We conclude that, for the *screaming-channel* case, the relation between the bits of the output of the Sbox and the leakage is not linear.

Since it appears only for *Screaming Channels*, we conjecture that the nonlinearity of the model is due to the path between the digital logic and the radio transmitter. Estimating the profile for the 256 values of $y = \mathrm{Sbox}(p \oplus k)$ (or of $y = p \oplus k$) can capture the nonlinearity of the model, and it is the best strategy for analysis and attack.



(a) ρ-test with $p \oplus k$ (green) and $HW[Sbox[p \oplus k]]$

(b) Profile for $y = HW[Sbox[p \oplus k]]$

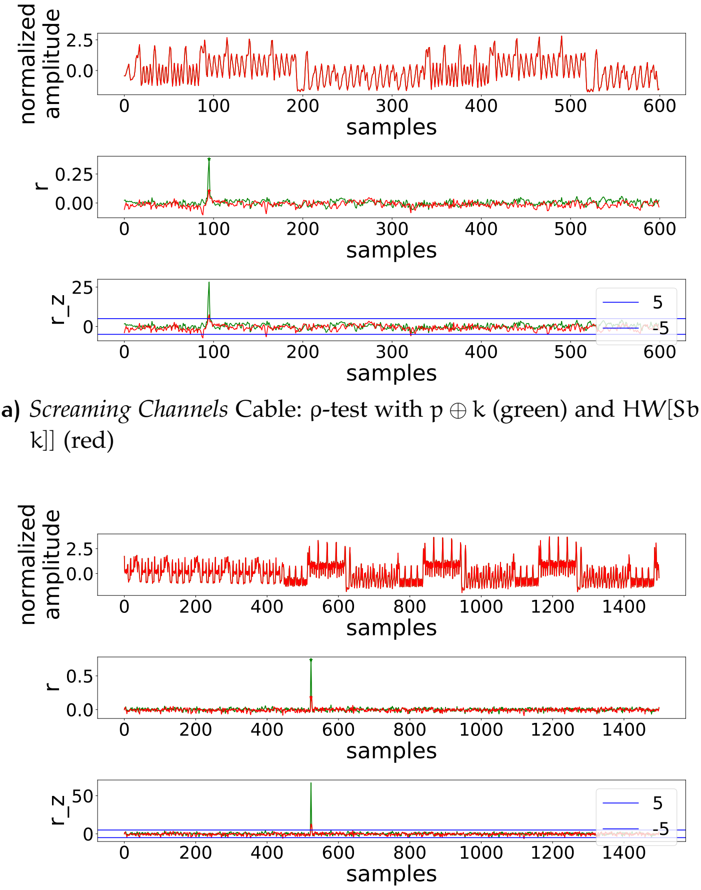
Figure 5.5: In the conventional case the Hamming Weight model is a good assumption. Results of the ρ-test for the first byte of the key (a). Assuming the $HW[Sbox[p \oplus k]]$ leakage model (red) does not significantly reduce correlation for the highest POI, compared to choosing $p \oplus k$ (green). The profile (b) clearly shows almost linear negative correlation.

5.3.3 The Impact of Channel Frequency

Profiles built at different frequencies might differ because of the different frequency response of the components along the path that brings the leakage to the antenna.

5.3.3.1 *Datasets*

Our target device with our firmware can transmit on 81 channels from 2.400 GHz to 2.480 GHz, spaced by 1 MHz. The advertising channels of BLE are particularly interesting for attacks because they are at a fixed known frequency. We decide to analyze the frequency used for the initial tests (2.400 GHz), and the advertising channels: channel 37 (2.402 GHz), channel 38 (2.426 GHz), and channel 39 (2.480 MHz). For extraction, we tune at $f_{channel} + 2 \cdot f_{clk} = f_{channel} + 2 \cdot 64\,\text{MHz}$. Using a *USRP B210* with a standard WiFi antenna at 10 cm from a



(a) *Screaming Channels* Cable: ρ-test with $p \oplus k$ (green) and $HW[Sbox[p \oplus k]]$ (red)

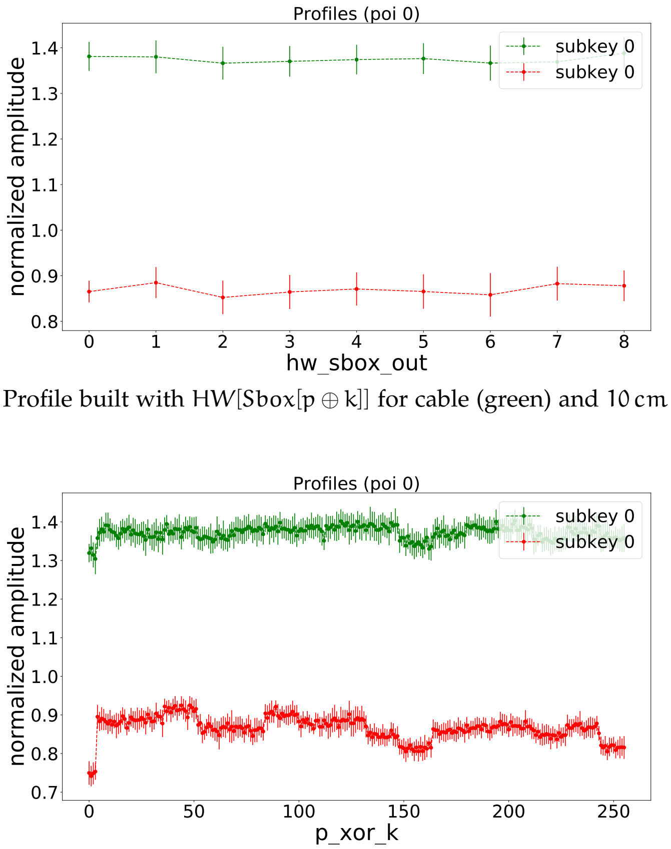
(b) *Screaming Channels* 10 cm: ρ-test with $p \oplus k$ (green) and $HW[Sbox[p \oplus k]]$ (red)

Figure 5.6: *Screaming-channel* leakages follow a distorted leakage model. For both the connection via cable (a) and at 10 cm (b), choosing $HW[Sbox[p \oplus k]]$ (red) drastically decreases the ρ-test correlation compared to choosing $Sbox(p \oplus k)$ or $p \oplus k$ (green).

BLENano v2 (*device E*) in an office environment, we collect $5000 \cdot 500$ profiling traces and $2000 \cdot 500$ attack traces, for each of the channels.

5.3.3.2 *Channel Frequency Does Not Impact Distortion*

For each set we run the ρ-test, and we correlate each profile with the one at 2.400 GHz to observe if there is any distortion. Results in Table 5.3 show that the amount of correlation and the shape of profiles is mostly constant. We conclude that the distortion of *Screaming Channels* does not have a strong dependency on the frequency of the channel, at least when looking at the first harmonic of the carrier and the second harmonic of the clock.



(a) Profile built with $\mathrm{HW}[\mathrm{Sbox}[p \oplus k]]$ for cable (green) and 10 cm (red)

(b) Profile built with $p \oplus k$ for cable (green) and 10 cm (red)

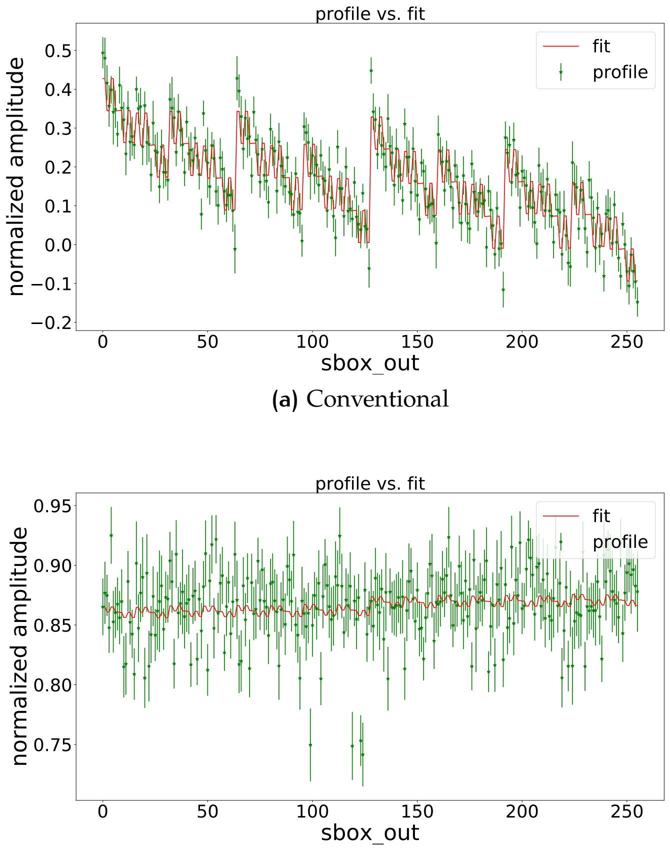
Figure 5.7: *Screaming-channel* leakages follow a distorted leakage model. Profiles for cable and 10 cm are similar (a)(b) are similar, but they differ from the conventional case shown in Figure 5.5b.

5.4 THE RADIO LINK

The *screaming-channel* leakages travel over a radio link between the target device and the receiver of the attacker. The distance that can be achieved with this setting is considerable compared to conventional attacks. We want to understand the properties of this channel and to exploit it at its best.

5.4.1 A Simple Model of the Leakage Transmission Chain

With a simple model from radio theory, we show the inherent advantage of *Screaming Channels* over conventional leakages, especially in case of large attack distance. Moreover, we explain which optimizations of the setup have the largest impact in improving the reception of the leakage.



(a) Conventional

(b) *Screaming Channels* at 10 cm

Figure 5.8: For the bits at the output of the Sbox, the linear model (red) is a good approximation of the leakage model (green) for the conventional case (a), but not for *Screaming Channels* (b).

5.4.1.1 *A Simple Model*

Figure 5.9 shows a simple model of the transmission chain between the target and the radio of the attacker. The input of the chain is the leakage trace, which has a Signal-to-Noise Ratio (SNR) $\mathrm{SNR}_{in} = P_{in}/N_{in}$, where the numerator is the power of the leakage trace and the denominator is the power of the noise (e.g., thermal noise) over the frequency band of interest. Note that, as explained in Chapter 2, this is the SNR of a single a trace, not the SNR in the Differential Power Analysis (DPA) sense (related to the data-dependent difference among traces corresponding to different plaintexts and keys). As seen in Chapter 4, the leakage trace is upconverted and amplified by the radio transmitter. Then, it goes through the channel, a pre-amplifier, and finally the receiver. Each block has a gain G and it introduces some additive noise N. The SNR at the output of each block is reduced by a factor called noise factor $F = 1 + \frac{N}{N_{in}G}$. The total noise factor of our chain is $F = F_{tx} + \frac{F_{ch}-1}{G_{tx}} + \frac{F_{pre}-1}{G_{tx}G_{ch}} + \frac{F_{rx}-1}{G_{tx}G_{ch}G_{pre}}$, and the SNR at the output is $\mathrm{SNR}_{out} = \mathrm{SNR}_{in}/F$. Note that, at each

Table 5.3: Similarity among profiles ($\hat{r}(P_i, P_2)$) and strength of the leakage (max ρ,r_z) for different channel frequencies (and the corresponding frequency at which we tune). Profiles are similar, showing that the distortion of *screaming-channel* leakages is constant with frequency. The same applies for the amount of correlation.

	Channel f (GHz)	Tune f (GHz)	$\hat{r}(P_i, P_2)$, -log10(p)	max ρ,r_z
P_0	2.400	2.528	1.00, inf	0.79, 74.73
P_1	2.402	2.530	0.83, 66.39	0.79, 75.27
P_2	2.426	2.554	0.86, 75.82	0.77, 71.22
P_3	2.608	2.480	0.85, 71.32	0.68, 58.49

step of the chain, the gain of the previous stage reduces the impact of the noise of the next stage. Early amplification has dominant positive effects on the SNR compared to amplification in the following stages. As explained in Chapter 2, the gain[1] of the channel in far field is $G_{ch} = G_{antenna\ tx} G_{antenna\ rx} (\lambda/2\pi d)^2$.

5.4.1.2 *Inherent Advantage of Screaming Channels and Setup Optimization*

The two main advantages of *screaming-channel* leakages over conventional leakages can be clearly spotted in the formula of the noise factor F. First, the trace leakage is amplified as early as possible in the chain by the radio transmitter (G_{tx}), reducing the impact of the noise factor of the channel F_{ch}. This is important because F_{ch} increases with the square of the distance d. Second, the intended radio transmitter is likely to have a good antenna gain $G_{antenna\ tx}$, which also contributes to reduce the total noise factor. As the intended transmitter block is not present in the case of conventional leakages, these advantages cannot be exploited for long range attacks. From the formula for F we can also observe the advantage of two well-known optimizations of the setup. First, the attacker can choose a directional antenna with a high gain $G_{antenna\ rx}$. Second, the attacker can use a low noise amplifier with a good noise factor to preamplify the traces before feeding them to the radio. The next amplification stages have less impact on the noise factor of the analog part, but they are important to match the trace amplitude with the dynamic range of the ADC to avoid losses in resolution.

5.4.2 A More Complex Channel

The radio channel between target and attacker is not affected by thermal noise only. First, as seen in Section 5.2, *Screaming Channels* are

[1] Note that the gain is less than 1 so it is actually a "loss".

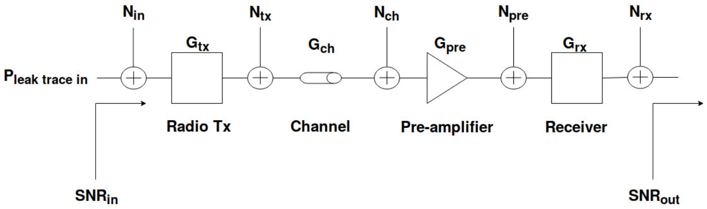


Figure 5.9: Simple model of the transmission chain between target and attacker.

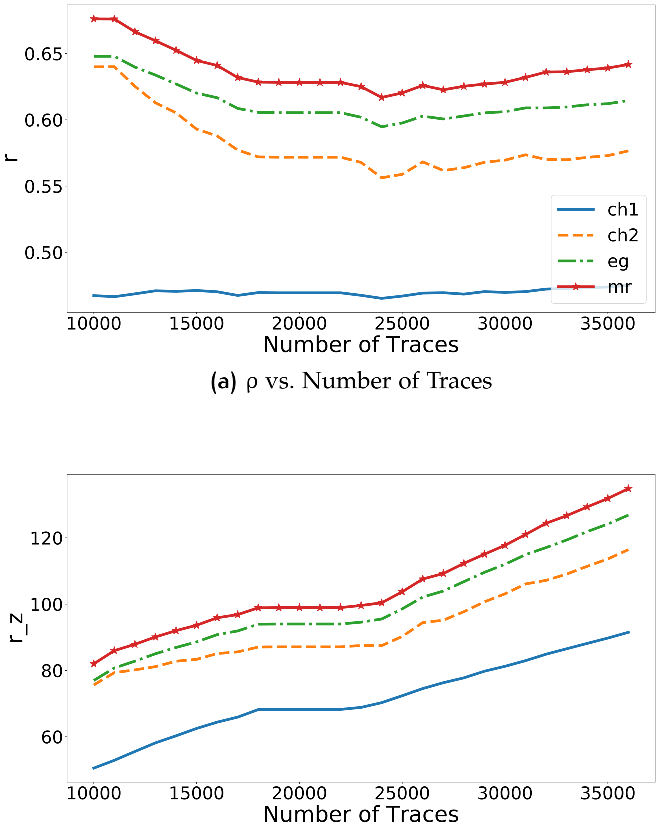
inherently characterized by intermittent deep fade. The channel is not active when packets are not transmitted, and leakage traces are not synchronous with the packets. Second, a number of other non ideal effects must be taken into account. The difference between the clocks of the receiver and transmitter will produce time-varying frequency offsets that impact demodulation. The channel introduces distortions because different frequencies in the band of interest undergo different losses. In real environments, the signal encounters obstacles that produce reflections. The resulting multiple paths have different delays and losses. Finally, gains and losses are different for different distances and setups, and they are not necessarily constant with time and temperature. These problems should be addressed during reception and preprocessing. Although at least some of these issues are typical of most communication channels and side channels, they are particularly important for radio communications such as *Screaming Channels*.

5.4.3 Time, Spatial, and Frequency Diversity

As explained in Chapter 4 and Section 5.2, our preprocessing techniques that averages many traces is a form of time diversity, in which the same message is sent multiple times over a noisy channel.

We can further improve the quality of the leakage traces using multiple spatial and frequency streams. Differently from time diversity, they increase the complexity and cost of the setup, but they do not require the target to repeat multiple times the same operation. Frequency diversity is possible because the leakage is broadcast at multiple harmonics, as explained in Chapter 4. Since these harmonics are spaced by 64 MHz, they are likely to be impacted by uncorrelated noise. For example, an interfering signal will not have a bandwidth large enough to impact two harmonics at the same time. However, this large spacing is also more demanding for the receiver, and impossible with our current experimental setup. For this reason, we leave it for future work, and we focus on spatial diversity.

We use two receive chains of a *USRP B210* radio to receive the leakage traces from two antennas at different positions. The leakage traces might arrive at the two antennas with different delays, because they go through different paths. However, they are transparently synchronized



(a) ρ vs. Number of Traces

(b) r_z vs. Number of Traces

Figure 5.10: Comparison among spatial diversity and single channels. Maximal ratio and equal gain show a net advantage in terms of correlation over channel 1 and channel 2 for any number of traces.

by the extraction process. Once aligned, the two traces coming from the two antennas can be combined together. We implement both equal gain and maximal ratio combination techniques. For the latter, we use a simple way of computing the quality of a trace: $\bar{l}(t)/\mathtt{std}(l(t))$. To compare the results, we collect a set of $37000 \cdot 500$ profiling traces at $10\,\mathrm{cm}$ in a home environment (*device E*). We use two standard WiFi antennas spaced by $3\,\mathrm{cm}$. The advantage of maximal ratio and equal gain over each of the single channels appears clearly in Figure 5.10, where the maximum correlation (and confidence) of different techniques are compared for a varying number of traces . We will show concrete examples of attacks in Section 6.1.2.

5.4.4 Normalization, Channel Estimation, and Profile Reuse

As seen in Chapter 3, normalization is an important method to improve the portability of profiles built in different conditions (e.g., device instance, setup, time). The radio channel is an additional source of variability for *Screaming Channels*. The total gain of the channel G (including the setup) changes with distance and it might not be stable over time. Different distances also force the attacker to use different equipment, increasing the variability of the setup. Normalization has to take into account these considerations.

5.4.4.1 *Normalization and Channel Estimation*

Building on previous work [51, 67, 124, 178], we propose a more complete form of normalization. Instead of applying z-score normalization to the entire set, or to remove the DC offset from each trace, we apply z-score normalization to each trace. Assuming a normal distribution, each point of the resulting traces will belong to $\mathcal{N}(\mu = 0, \sigma = 1)$. The advantage of this method is that it works as a form of per-trace channel estimation, which can filter out the variations of gain with time, temperature, and other external factors. This is particularly important for *screaming-channel* traces, because an acquisition campaign might be long and the environment might be subject to change. Assuming that the overall gain G_{total} is constant for the short duration of a trace, at reception we have the following trace, average, and standard deviation:

$$\begin{aligned} l_{rx} &= G_{total} l_{tx} \\ \bar{l}_{rx} &= G_{total} \bar{l}_{tx} \\ std(l_{rx}) &= G_{total} std(l_{tx}) \end{aligned} \tag{5.1}$$

Therefore, the normalized trace at reception is independent of the channel gain:

$$\frac{l_{rx}(t) - \bar{l}_{rx}}{std(l_{rx})} = \frac{G_{total} l_{tx}(t) - G_{total} \bar{l}_{tx}}{G_{total} std(l_{tx})} = \frac{l_{tx}(t) - \bar{l}_{tx}}{std(l_{tx})} \tag{5.2}$$

Independently from the conditions in which the leakage was transmitted and received, all normalized traces are comparable. This improves the portability of profiles, as explained in Section 5.5. In [291], a recent research that applies attacks based on deep learning to *Screaming Channels*, min-max normalization is used instead.

5.4.4.2 *Additional Preprocessing Techniques*

As of now, this normalization is applied after having extracted a single trace from m traces. We leave as future work the investigation of better preprocessing techniques, including normalization, to apply to each trace before averaging. For example, we could exclude those points that are affected by deep fade, and remove noise components with filters in the frequency domain or with Singular Spectrum Analysis (SSA) [96, 209, 281]. As of now alignment is mainly based on cross-correlation, we could instead use Dynamic Time Warping (DTW) [293] to account for instabilities in the clock. We could as well track the frequency offset, and equalize the traces if the channel gain is not constant with frequency.

5.4.5 The Impact of Distance (and Setup) on Correlation and Distortion

In the far-field[2], the main effect of distance is the quadratic reduction of the gain of the channel. To compensate for this loss we have to use higher amplification and higher-quality components. This is why we cannot fully decouple the effect of distance from that of the setup. We optimize reception for each distance we want to study, and then we compare the results. This is a fair and realistic case, since also the attacker will adapt the setup to the conditions.

5.4.5.1 *Datasets*

In addition to the traces collected in Section 5.3, we collect the following sets of $5000 \cdot 500$ traces:

- **Home** $20\,\mathrm{cm}$ **(*device E*).** Traces are collected with a standard WiFi antenna and a *HackRF* radio. The target is a *BLE Nano v2*.
- **Office** $1\,\mathrm{m}$ **(*device F*).** Traces are collected with a *USRP N210* radio. To avoid reflections, and to increase the gain, we use a directional antenna with $24\,\mathrm{dBi}$ gain [266], and two low-noise amplifiers with $20\,\mathrm{dB}$ gain and $1.5\,\mathrm{dB}$ noise figure [175]. The target is a *BLE Nano v2* (with *Particle Debugger*).
- **Anechoic** $5\,\mathrm{m}$**.** Same as above, but in an anechoic chamber, against a *BLE Nano v2*.
- **Anechoic** $10\,\mathrm{m}$**.** Same as above, but at larger distance.

For each set, we run the ρ-test, and we build the profile, choosing $y = p \oplus k$ as leakage variable. Then, we measure the similarity among profiles by correlating each profile with the one taken at $10\,\mathrm{cm}$. Results about distortion and correlation are visible in the last two columns of Table 5.4, respectively.

5.4.5.2 *Correlation Stays High*

We observe that with enough amplification it is possible to collect good traces even at large distance. The quality of the setup and of the environment seem to play a bigger role. For example, the amount of correlation at $5\,\mathrm{m}$ in an anechoic chamber with a good setup is

[2] Our leakage is at $2.528\,\mathrm{GHz}$ and the maximum size of the antenna of the target is $6\,\mathrm{mm}$, which gives the following far field conditions: $d > 2D^2/\lambda = 0.6\,\mathrm{mm}$, $d \gg D = 6\,\mathrm{mm}$, and $d \gg \lambda = c/f = 119\,\mathrm{mm}$.

Table 5.4: Similarity among profiles ($\hat{r}(P_i, P_2)$) and strength of the leakage (max ρ,r_z) for different distances and setups. All profiles taken for *Screaming Channels* are similar, showing that the distortion of *screaming-channel* leakages is constant. The amount of correlation stays high despite distance, and it rather depends on the quality of setup and environment.

	d (m)	Environment	Antenna	$\hat{r}(P_i, P_2)$, -log10(p)	max ρ,r_z
P_0	0.00	conventional	loop probe	-0.07, 0.56	0.95, 127.35
P_1	0.00	cable	n.a.	0.55, 21.09	0.39, 28.96
P_2	0.10	home	standard	1.00, inf	0.79, 75.72
P_3	0.20	home	standard	0.96, 142.77	0.77, 72.30
P_4	1.00	office	directional	0.40, 10.32	0.41, 30.66
P_5	5.00	anechoic	directional	0.96, 139.51	0.85, 89.84
P_6	10.00	anechoic	directional	0.92, 107.80	0.77, 71.71

even higher than correlation at 10 cm in a home environment with a poor setup. An attacker can profit from the best conditions during the profiling phase. Overall, the amount of correlation tends to be high in all cases.

5.4.5.3 *Distortion is Stable*

All the profiles built in *screaming-channel* conditions correlate well among each other. Only the setup via cable shows a lower correlation. On the contrary, the profile built in the conventional setting is not correlated with the *screaming-channel* ones. This shows that the distortion of the profile is stable with distance and across setups. This confirms the hypothesis made in Section 5.3.2.2 that distortion happens on the path from the digital logic to the radio channel.

5.5 PROFILE REUSE

In this thesis we focus on profiled correlation attacks, since they are well adapted to *screaming-channel* traces. As explained in Chapter 2, we can evaluate profile reuse by computing the correlation between different profiles. We have already observed that the profile of the leakage is stable with distance, setup, and channel frequency. In the following we fix the parameters above, and we focus on the impact of two additional factors: different acquisition campaigns on the same device, and different device instances (different instances of the *BLE Nano v2*). In this phase, we use 5 *BLENano v2.0* devices (*A*, *B*, *C*, *D*, *E*, and *F*), with a *Particle Debugger*. We also try device *F* with a *DAPLink Board* (that we call F_{DAP}). All measurements are taken in the same

Table 5.5: Comparison among several profiling campaigns.

	$\hat{r}(P_i, P_2)$, -log10(p)	ρ, r_z
P_{A0}	1.00, inf	0.81, 80.70
P_{A1}	0.97, 136.23	0.80, 77.68
P_{A2}	0.93, 80.13	0.69, 60.39
P_{A3}	0.90, 80.63	0.66, 55.87
P_{A4}	0.98, 138.86	0.83, 85.01
P_{A5}	0.98, 136.37	0.83, 83.38
P_{A6}	0.98, 144.04	0.84, 86.90
P_{A7}	0.98, 139.21	0.83, 84.68

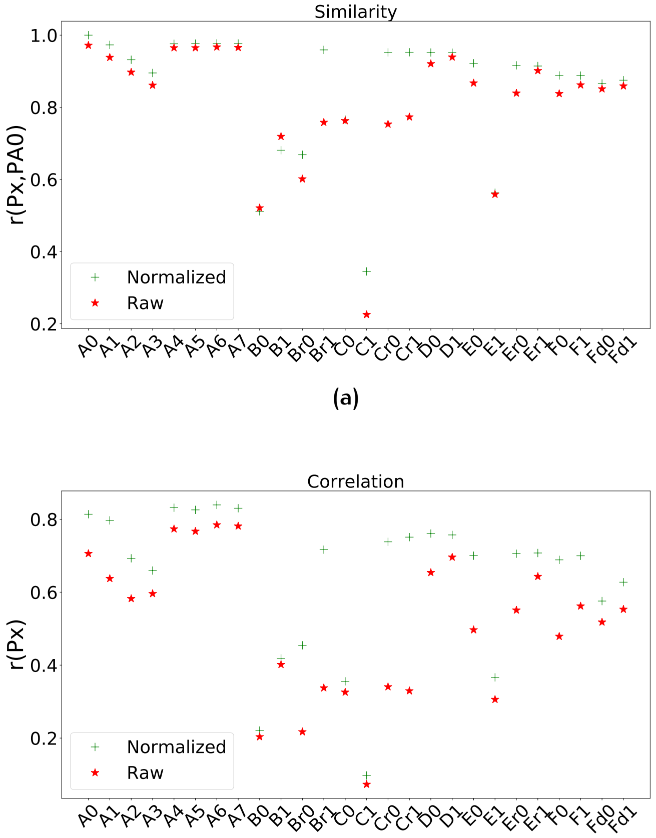
office setup at 10 cm, using a *USRP B210* SDR, and a standard WiFi antenna.

5.5.1 Reuse Over Time

In real attack scenarios, profiling and attack traces are most likely collected in distinct acquisition campaigns, at different times. The difference between acquisition campaigns cannot be avoided, even when profiling and attacking the same device. Using device A, we collect 8 profiling sets of $5000 \cdot 500$ traces, followed by 8 attack sets of $2000 \cdot 500$ traces. The total collection time is around 22 hours. Results in Table 5.5 show that the amount of correlation and the shape of the profile are quite stable for different campaigns despite several hours of difference. In Section 6.1.3.2, we will show that a profile can be reused several months later as well.

5.5.2 Reuse Across Devices

To evaluate the portability of profiles across devices, we also collect 2 sets of $5000 \cdot 500$ profiling traces and 2 sets of $2000 \cdot 500$ attack traces, for devices B, C, D, E, F, and F_{DAP}. Results in Figure 5.11 show that profiles taken during different acquisition campaigns and on different devices can be reused without considerable loss. Only a few cases (*B0*, *B1*, *C1*, *E1*) behave as outliers, because the quality of the profile is low. Probably, this is due to some external factor during the acquisition in a realistic office environment (e.g., the presence of an external source of noise). In this case we repeat the measurement. The leakage is clearly visible and in the same order of magnitude in most profiles in Figure 5.11b. Profiles are very similar to each other, as shown in Figure 5.11b, where each profile is compared with the one taken on device A. In the few cases when the correlation among profiles is low, the cause is the low quality of one of the profiles rather



(a)

(b)

Figure 5.11: Similarity with the profile of device *A0* (a) and strength of the leakage (b) for profiles built on different devices during several acquisition campaigns over a a period of several hours. Apart from the few cases when the quality of a profile is low, similarity (a) and correlation (b) are significantly high. The confidence is $-\log 10(p) > 10$ and $r_z > 10$ even for the worse cases. The suffix *r* indicates a second acquisition campaign for the cases that had problems. The suffix *d* indicates the *DAPLink* debugger. The advantage of normalization over raw traces in clearly visible.

than a significant difference in their shape. Normalization brings a clear advantage over raw traces. In addition to correlation, we also provide some attack metrics in Figure 5.12. For each profile, we run an attack with traces collected with the same device or with traces collected on device *A*. We show the evolution of the key rank with different campaigns and devices. In general, the key rank increases when attacking different devices, but not significantly, and it remains in the same order of magnitude as in the case of different acquisition campaigns. Normalization clearly improves the portability of the templates. The same applies when averaging the profiles of the 16 bytes together, as we will explain more in detail in Section 5.6.2.1.

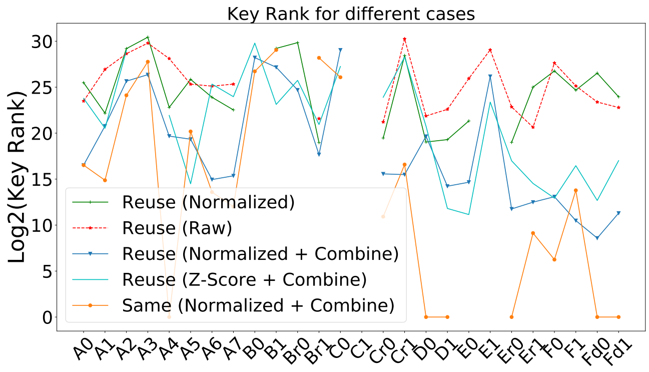


Figure 5.12: Key rank for different devices and collection campaigns. The baseline (orange) shows the best case (attacking and profiling the same device, normalized traces, average of the 16 bytes as explained in Section 5.6.2.1). The other lines show the key rank when using the attack traces of device *A0* (first acquisition) with each of the other profiles. In most cases, the key rank shows a moderate increase, which is in the same order of magnitude for time and device changes. Missing points correspond to a key rank bigger than 2^{30}. Our per-trace normalization (blue) has similar results that z-score normalization of the set (cyan).

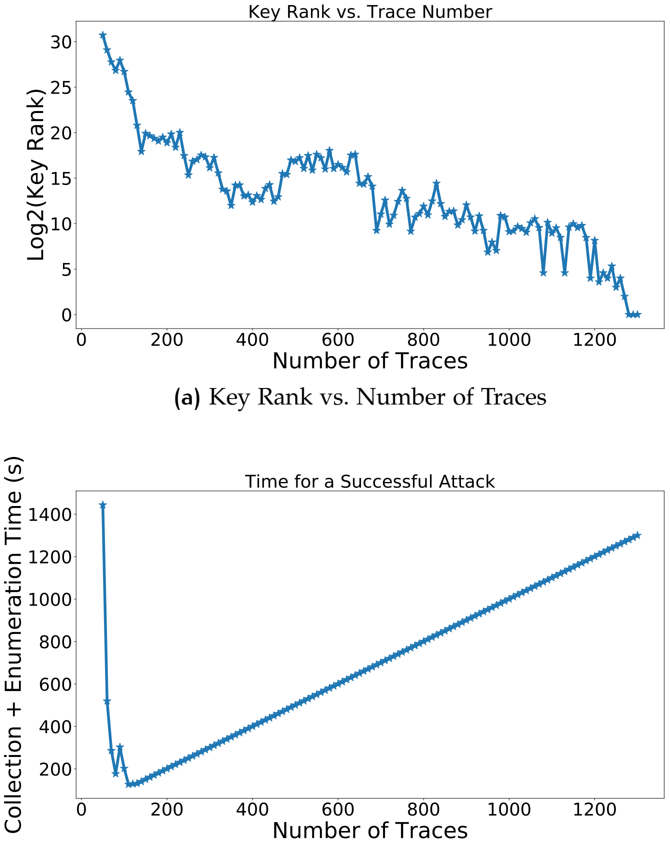
5.6 OTHER PRACTICAL OBSERVATIONS

We have investigated a novel channel and uncovered its distinctive properties. We now explore a few additional practical aspects:

- The point at which key enumeration becomes useful given the slow collection speed of *screaming-channel* traces.
- The impact of the extremely low sampling rate of *Screaming Channels* on the number of POIs, and how to best exploit them when there are more than one.
- How to combine the profiles of several bytes for a more accurate estimate.
- The role of the connection between the target device and the attacker in our current experimental setup.

5.6.1 Key Enumeration

In general, side-channel attacks quickly recover most of the bits of the key, but then take many more traces to converge to full recovery. When the convergence rate slows down, it may be faster to bruteforce the remaining bits than to collect more traces. This is particularly true for *screaming-channel* attacks, because acquisition is slow, and it is only possible when there is an ongoing transmission.



(a) Key Rank vs. Number of Traces

(b) Total Attack Time vs. Number of Traces

Figure 5.13: Analysis of key enumeration, for $5000 \cdot 500$ template traces and $2000 \cdot 500$ attack traces at a distance of $10\,\mathrm{cm}$ in a home environment.

To show a concrete example, we collect $5000 \cdot 500$ template traces and $2000 \cdot 500$ attack traces at a distance of $10\,\mathrm{cm}$ in a home environment with a *HackRF* radio on *device E*, at a speed of around $1\,\mathrm{trace/s}$. Figure 5.13a shows how the key rank decreases slowly with the number of traces. It is therefore faster to collect fewer traces and run key enumeration, provided we do not reach the limit at which enumeration becomes very expensive, as shown in Figure 5.13b. The enumeration speed depends on the resources. For example, a common laptop[3] with the `Intel AES-NI` hardware AES instruction takes a few seconds to enumerate up to 2^{20}, a few minutes to reach 2^{30}, and several hours for 2^{37}. An attacker could parallelize enumeration to improve performance.

[3] HP ENVY, Intel(R) Core(TM) i7-4700MQ CPU @ 2.40GHz, 11GiB Memory.

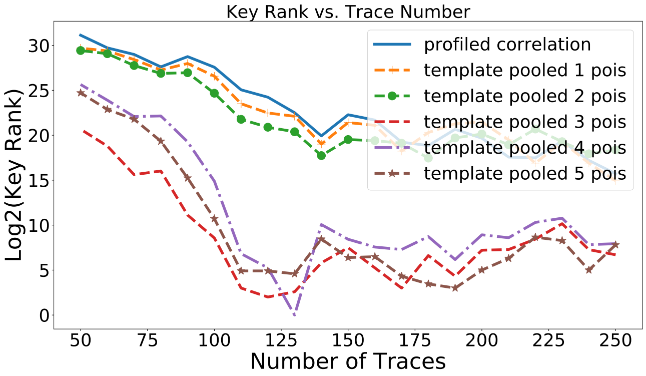


Figure 5.14: Comparison of multivariate template attacks with profiled correlation, for 5000 · 500 template traces and 2000 · 500 attack traces at a distance of 10 cm in a home environment The attacks with more than one POI clearly converge faster.

5.6.2 Low Sampling Rate and Informative Points

The clock frequency of the target is 64 MHz. Because of the limitations imposed by the radio hardware and data processing, we sample at a rate of 5 MHz. This limits our ability to capture many informative points in the traces, reducing the interest of dimensionality reduction, multivariate template attacks, and other techniques that combine several POIs.

Nevertheless, when we profile in good conditions and with enough traces, we can identify more than one POI, and obtain some advantage from multivariate template attacks. Since collection is slow, and because we have to estimate the leakage for 256 values to account for the distortion, the number of traces that we have in each of the 256 classes is normally not large. In this context, using the pooled covariance approach helps to increase the accuracy of the estimate and to avoid computational problems. In the following, we show the advantages of this method in three cases in which it is applicable.

We take the same example as in Section 5.6.1. We run multivariate template attacks with pooled covariance, with variable number of traces and POIs. Figure 5.14 shows the net advantage over a profiled correlation attack.

We compare a profiled correlation attack and a multivariate attack with pooled covariance and 5 POIs on the traces at 10 m in an anechoic chamber. Given 50000 · 500 template traces, the first attack retrieves the key with 35 · 500 attack traces and enumeration up to 2^{34}, whereas the second succeeds with 15 · 500 traces and enumeration up to 2^{31}.

In practice, in our experiments we observe that collection is rarely good enough to extract multiple POIs, and we focus on the use of profiled correlation attacks with one POI. The latter are also computationally more efficient, and thus more useful when the attack set is large (e.g., in challenging scenarios). However, multivariate attacks on higher quality traces remain an interesting direction for future research.

5.6.2.1 *Combining the Profiles of Different Bytes*

In a software implementation of *AES-128* such as the `tinyAES` one, there are 16 bytes that leak at 16 different points in time. In our attacks, we profile each of them independently.

For a byte i, given a leakage variable $p \oplus k$, each of the 256 possible values of each profile $\hat{model}_{Y_i}(y_i)$ is estimated with $N/256$ measurements, taken at different points in time. Assuming that the profile $\hat{model}_{Y_i}(y_i)$ is the same for each byte i, we can build a single profile using 16 measurements from each trace, instead of 1. Each of the 256 possible values of the single profile $\hat{model}_Y(y)$ is estimated with $16 \cdot N/256$ measurements, taken at different points in time. This brings a gain in accuracy equivalent to collecting 16 times more traces.

The hypothesis that the profile of each byte is the same is reasonable in our case, since the instructions executed for each byte are always the same. We confirm this hypothesis by observing that the correlation between the leakageage models of different bytes is very high (in the order of 0.99).

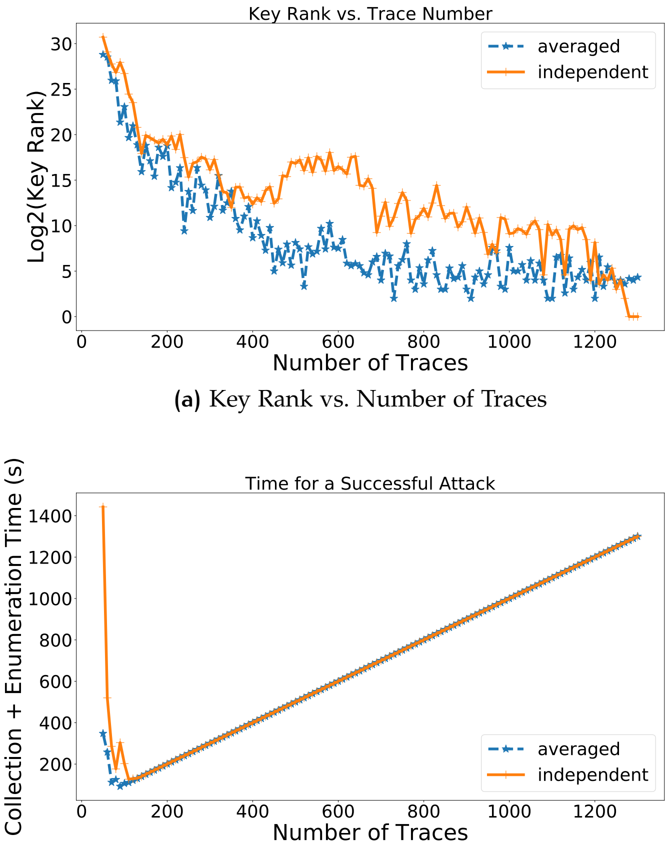
In practice, we still profile the bytes independently, but we add an option to profiled correlation attacks to use a single profile obtained from the average of the 16 different profiles: $\hat{model}_Y(y) = \sum_{i=0}^{15} \hat{model}_{Y_i}(y_i)$.

To show the gain of this combining method, we use the same example as in Section 5.6.1. Results are shown in Figure 5.15. The advantage of combining the bytes is evident, especially when the number of traces is low. We will show another example in Section 6.1.3 for an attack at 15 m in an office environment.

This approach and the pooled covariance approach proposed in [50] are similar but different, as they operate on different dimensions. We assume that each byte has the same average leakage, so that we can compute a pooled estimate of this average leakage with a pool of 16 bytes, for each possible value of the leakage variable. Instead, the authors of [50] assume that the covariance is the same for each value of the leakage variable, and they propose to estimate the pooled covariance.

5.6.3 Different Connection Types

In our setup, we use different types of connections between the control laptop, the target, and the radio. In the following, we extend the analysis of our setup to investigate if the type of connection has an impact on the attacks, which could be due, for example, to an unexpected coupling path.



(a) Key Rank vs. Number of Traces

(b) Total Attack Time vs. Number of Traces

Figure 5.15: Comparison between independent profiles and averaged profiles, for $5000 \cdot 500$ template traces and $2000 \cdot 500$ attack traces at a distance of $10\,\mathrm{cm}$ in a home environment.

The control laptop drives the target (*device E*) through a serial line. The connection is made via a USB cable and a *YKUSH* switchable hub [299]. It has individual power drivers for each port, which should reduce coupling and noise propagation from/to the laptop.

The *HackRF* and the *USRP B210* radio are connected to the control laptop through a USB cable. This cable and the one from the YKUSH are connected to the laptop through the same USB hub, upstream w.r.t. the YKUSH to reduce coupling. We have also written a version of the code in which control of the radio and of the target can be split and distributed on two different laptops. In this case the laptops share only the power supply, or nothing if one of the two is powered on battery. Unfortunately, the delays of remote communication between the two laptops make this solution less practical and harder to configure, and this may also have an impact on the quality of the collected traces.

The *USRP N210* uses Ethernet for control and data. We use three different types of network connections with the control laptop. First, a direct connection via an Ethernet cable. Second, a connection that goes through a switch. Third, we connect both the radio and the laptop to

Table 5.6: Comparison of different connection types.

Environment	Same USB	Same power	Floating	Direct Ethernet	Building LAN
Antenna	$p \oplus k$	$p \oplus k$	$p \oplus k$	fixed vs. fixed	fixed vs. fixed
max ρ	0.64	0.12	0.46	0.89	0.77
max r_z	51.64	8.24	33.98	97.99	70.22

two different ports of the Local Area Network (LAN) of the building. Unfortunately, in this last case network delays may cause data loss and reduce the quality of the traces.

At a distance of $10\,\mathrm{cm}$ in a home environment, we collect $4700 \cdot 500$ profiling traces for each of the three possible USB connections with a *HackRF* radio, and $4700 \cdot 500$ fixed vs. fixed traces for direct Ethernet connection and connection through the LAN of the building for the *USRP N210*. The maximum correlation found with the ρ-test and its confidence are shown in Table 5.6. Direct connections lead to better results. It is hard to pin point the exact reason, as indirect connections suffer from larger delays and data losses that may reduce the quality of the traces. We leave a more detailed study of this aspect for future work. For our large distance attacks, we use the *USRP N210* connected via a switch or the LAN of the building. We believe that the ability of the attacker to plug to the same power/Ethernet network as the victim in the same building is a reasonable hypothesis. More comparison data between Ethernet connections will be given in Section 6.1.3.

5.7 CONCLUSION

5.7.1 Lessons Learned

From what we have learned about the channel, we can draw the following conclusions:

- **Collection.** As expected, the radio channel plays an important role. The attacker should maximize the gain to compensate for the quadratic power loss of the leakage with distance. The use of a directional antenna and/or of spatial and time diversity helps to reduce several forms of noise, including multi-path reflections and other radio sources. Time diversity is also useful to overcome the deep fade condition that arises when no packet is being transmitted. For GFSK transmissions with this chip, the intended data transmission is otherwise orthogonal to the leak. The quality (e.g., noise figure) of the radio hardware and of the environment (e.g., anechoic chamber) might be more important than the effect of distance. For example, a profile collected in an anechoic chamber with good hardware at 5 m might be better than one taken in a home environment with low-end hardware at 10 cm. Trace collection is particularly slow for *Screaming Channels*, because the attacker has to wait for packet transmission to extract useful signals.

- **Profiling.** On our target chip, *screaming-channel* leaks have a nonlinear leak model, whereas conventional leaks follow the Hamming Weight model. This distortion is mostly independent of distance, setup, and channel frequency. The attacker can capture a nonlinear model by choosing $y = p \oplus k$ and estimating the model for each possible value of y. Thanks to normalization and channel estimation, profiles can be reused in different conditions, at different distances, and against different instances of a device. The attacker can, if necessary, perform profiling in convenient conditions, and attack a different instance in a more challenging setup. The huge advantage on profiling is by far worth the small disadvantage of attacking a different instance. Combining the profiles of different bytes can lead to a better pooled estimate. Given the extremely low sampling speed, the number of POIs is small.

- **Attack.** Attacks must be able to capture the nonlinearity of the leakage model. This is possible thanks to the profiling step of profiled correlation and template attacks (with a good choice of $y = p \oplus k$ to estimate the full profile). We observe that univariate template attacks (which can capture a second order relation between the leak $l(y)$ and the model $\mathtt{model}(y)$), do not perform better than profiled correlation attacks (which can only capture the linear correlation between leak and model). We conclude that, for our sample size, looking at linear correlation is enough. Multivariate template attacks are instead more efficient, because the exploit the information of more than one POI. However, given the low sampling frequency and the complex channel, multiple POIs are available only in a few favorable cases (e.g., in the anechoic chamber). Profiled correlation attacks are therefore the preferred attack tool in this thesis. They also have the additional advantage of being practical and fast. Key enumeration is often very useful since collection is slow, especially in challenging attack scenarios.

5.7.2 Summary of the Improvements

We take the attack at 10 m in an anechoic chamber presented in Chapter 4 as reference, and we apply the improvements that we have learned.

	Profile	max ρ,r_z	Attack	Type, POIs	Rank
Chapter 4	130k	0.14, 52	1428	t.a., 10	0
Less Profiling Traces	50k	0.15, 34	3000	p.c., 1	$> 2^{33}$
1 Full Profile	50k	0.63, 167	3000	p.c., 1	2^{29}
2 Normalization	50k	0.18, 41	3000	p.c., 1	$> 2^{33}$
1 + 2	50k	0.78, 235	3000	p.c., 1	2^{13}
3 Combining + 1 + 2	50k	0.78, 235	1775	p.c., 1	0
3 Combining + 1 + 2	50k	0.78, 235	100	p.c., 1	2^{23}
3 Combining + 1 + 2	50k	0.78, 235	35	p.c., 1	2^{33}
4 Univariate + 1 + 2	50k	0.78, 235	1775	t.a.p.c., 1	2^{18}
4 Univariate + 1 + 2	50k	0.78, 235	100	t.a.p.c., 1	2^{28}
4 Univariate + 1 + 2	50k	0.78, 235	35	t.a.p.c., 1	2^{36}
5 Multivariate + 1 + 2	50k	0.78, 235	1775	t.a.p.c., 5	0
5 Multivariate + 1 + 2	50k	0.78, 235	100	t.a.p.c., 5	2^{23}
5 Multivariate + 1 + 2	50k	0.78, 235	15	t.a.p.c., 5	2^{31}

Table 5.7: Before exploring more challenging attacks, we show as reference the attack at 10 m in an anechoic chamber of Chapter 4. We apply several improvements incrementally to show their impact.

Results in Table 5.7 show the impact of each technique. Before starting, we reduce the number of profiling traces and we move to profiled correlation with 1 POI (Less Profiling Traces). Considering the nonlinear leakage model by estimating the full profile (1 Full Profile) significantly increases the correlation, compared to using the Hamming Weight model. Indeed, the leakage model for *Screaming Channels* is nonlinear. Similarly, correlation further improves when normalizing the traces (2 Normalization). We recall that higher correlation results in a reduction of the number of traces required for a successful attack. Combining the profiles of the 16 bytes further improves the attacks (3 Combining). Univariate template attacks with pooled variance (4 Template) are not better than profiled correlation attacks, showing that the leakage is of the first order. Multivariate template attacks with pooled covariance (5 Multivariate) are better when more POIs are available. This is the case with many traces in the anechoic chamber, but generally only one or few POIs emerge from traces collected in more challenging environments.

6 ATTACKS AND COUNTERMEASURES

In this chapter, thanks to the better understanding of *screaming-channel* leakages gained in Chapter 5, we target challenging attack scenarios. We explore different targets (optimized code, hardware block), real-world environments (home with obstacles, office at large distance), and a real-world application (Google Eddystone beacons authentication). We also describe possible countermeasures.

6.1 CHALLENGING ATTACKS

6.1.1 More Challenging Targets

While non-optimized `tinyAES` is a good benchmark to study *screaming-channel* effects, we are also interested in evaluating more realistic targets.

6.1.1.1 *Optimized Code*

We compile `tinyAES` with higher optimization level (-O3), and we investigate the *screaming-channel* leakage at $10\,\mathrm{cm}$ with a *HackRF* radio and a standard WiFi antenna. We collect $5000 \cdot 500$ traces on *device E*. On one side, collection becomes faster, but on the other side the code performs fewer memory accesses, leading to lower consumption and lower correlation. Figure 6.1a shows the shape of the leakage trace, whereas Figure 6.1b is the result of the ρ-test. The correlation is around 15 times smaller than for the profile taken in the same conditions on non-optimized code (Section 5.6.1). This shows that an attack is still possible, but with a significantly larger number of attack traces. For example, for profiled correlation attacks the number of traces required for key recovery increases with the square of the correlation ($N \propto c/\rho^2$) [260]. The shape of the leakage for the two profiles has a certain degree of similarity, shown in Table 6.1. In Section 6.2 we will show an example of attack against a real application compiled with optimizations.

6.1.1.2 *Hardware AES-128*

The nRF52832 has a dedicated hardware block for *AES-128*, which can be used both in Electronic Code Book (ECB) and Counter with Cipher Block Chaining Message Authentication Code (CCM) mode. ECB could be useful, for example, to build simple authentication schemes at the

Table 6.1: Correlation among profiles (byte by byte for POI 0) between optimized and non-optimized traces.

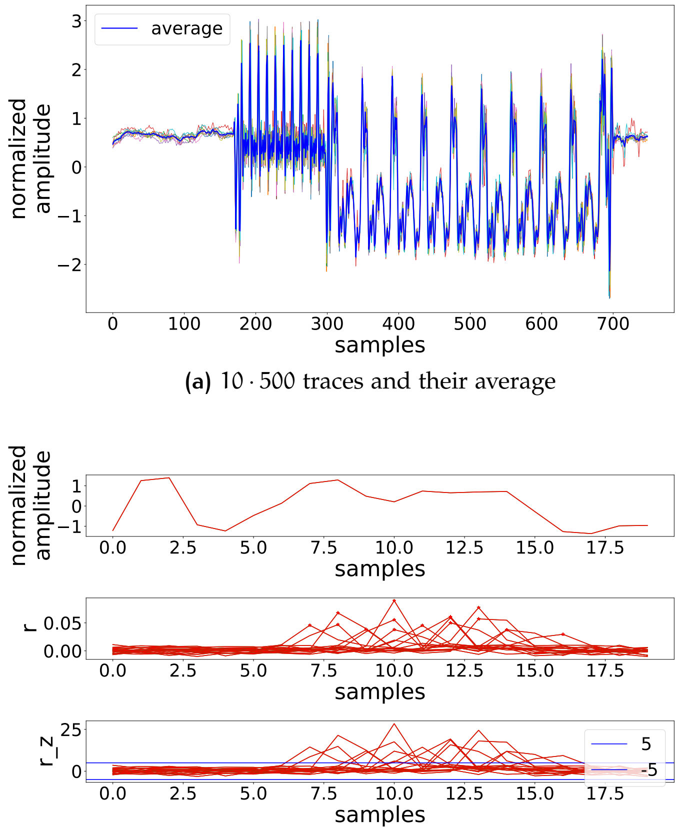
Byte	0.00	1.00	2.00	3.00	4.00	5.00	6.00	7.00	8.00	9.00	10.00	11.00	12.00	13.00	14.00	15.00
ρ	0.30	0.32	0.38	0.36	0.20	0.36	0.38	0.18	0.19	0.31	0.50	0.25	0.15	0.24	0.37	0.15
-log10(p)	6.14	6.86	9.41	8.53	2.84	8.41	9.25	2.50	2.76	6.44	16.94	4.36	1.72	4.10	8.81	1.83

application layer [61]. CCM is used by BLE to protect the link layer, and it has priority. It can be configured to encrypt packets concurrently and synchronously to packet transmission, which is a positive point for *Screaming Channels*.

Leakage Detection. We collect $350000 \cdot 100$ profiling traces on *device F* at $10\,\mathrm{cm}$ in an office environment with a *USRP B210* radio. To simplify collection, we transmit a continuous wave, and we add a preamble (a simple series of nested loops) before calling the encryption function. In our version of the SDK [190], the driver copies plaintext and key into a structure read by the hardware block. Similarly, the ciphertext is copied back from this structure to the location required by the user. The leakages originating from p, k, or c, appear on the radio signal, as shown in Figure 6.2. The hardware block uses a Direct Memory Access (DMA) mechanism to load plaintext and key from RAM memory, and to write back the ciphertext. We are able to observe small correlation peaks for $HW[p]$, $HW[k]$, and $HW[c]$ (Figure 6.2), due to the hardware block reading and writing data. Using the Hamming Weight is not the best choice to capture distortions, but it requires fewer traces to observe the peaks, because it reduces the number of classes. Despite the large number of traces and the quality of the setup, we were not able to detect any leakage due to the internal operations of the hardware block. Connecting the device via cable and with an additional amplifier did not improve this result. We conjecture that this is due to the lower power consumption and higher execution speed of the hardware block compared to the software implementation.

Differential Power Analysis (DPA). Since the internal operations of the hardware block are not visible with the current setup, we were not able to run a DPA attack (e.g., on the output of the Sbox). This result does not exclude the possibility that these leakages are visible with more complex reception hardware, or that the hardware block of other mixed-signal designs is strongly affected by *Screaming Channels*.

Simple Power Analysis (SPA). The correlation peaks found for the memory transfers of plaintext or key material used by the hardware block can be attacked using SPA. Besides SPA being harder than DPA, correlation is low, making an attack hard in practice. Despite collecting



(a) 10 · 500 traces and their average

(b) ρ-test

Figure 6.1: Leakage at optimization level O3, at 10 cm in a home environment. The *AES-128* encryption is clearly visible (a). The ρ-test detects the POIs which show significant correlation with the leakage variable (b).

1000000 · 100 attack traces we were not able to reduce the rank of the key enough to run a key enumeration attack. However, attacking the transfer of sensitive material to the hardware block is an interesting open direction for future research.

In practice, we conclude that the hardware cryptographic block of the *nRF52832* chip is as of now by far harder to attack with *Screaming Channels* than a software implementation.

6.1.2 More Challenging Environments

Radio transmissions in real-world environments such as a house are made more complex by the presence of interfering devices and reflections on obstacles. Nevertheless, *screaming-channel* attacks against `tinyAES` (without optimizations) are still feasible.

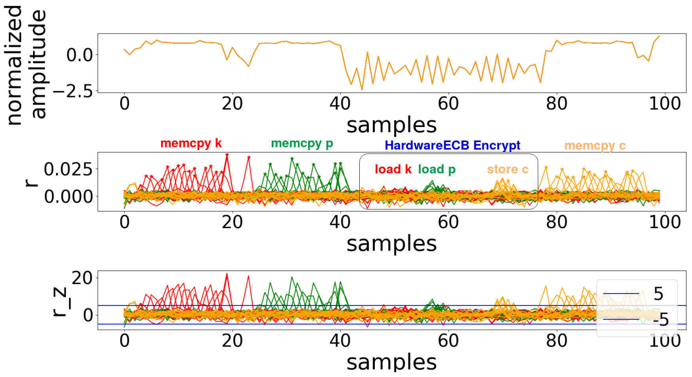


Figure 6.2: ρ-test on hardware *AES-128*, using 350000 · 100 traces at 10 cm, in an office environment. Peaks for HW[k] (red), HP[p] (green) and HW[c] (orange) are visible. The bigger peaks correspond to the firmware copying the values to the data structure known by the hardware block. The smaller peaks are due to the hardware block loading those values.

6.1.2.1 0.55 m *in Home Environment with Obstacles*

In a realistic scenario, it may be hard for an attacker to find a direct line of sight to the target to use directional antennas, but the problem can be solved with spatial diversity.

Attack Set for *Device E*. We place our target on a table, at 0.55 m from a B210 radio with two receive chains. The distance was chosen based on the size of the table. We use two standard WiFi antennas at a distance of 3 cm from each other. In the middle, we introduce three common off-the-shelf objects with different shapes and materials, which cover the line of sight. Moreover, the table, the curtain, and other walls and obstacles in the room may reflect the signals as well. In our experiments the setup is fixed, but spatial diversity would also be beneficial should the obstacles move over time, because it dynamically finds the best combination of the signals coming from the two antennas. We collect 2000 · 500 attack traces in this setup, shown in Figure 6.6a.

Profiling *Device E* in a Convenient Setup. Before attacking in this challenging condition, we build a profile of the leakage at 10 cm in line of sight, using 37000 · 500 traces, combined from two channels with the maximal ratio technique (same profile as discussed in Section 5.4.3). We then attack using profiled correlation and average of the 16 bytes as explained in Section 5.6.2.1. We can recover the key using enumeration up to 2^{24}, 2^{28}, and 2^{27}, with 1990 · 500 traces from channel 1, equal gain, and maximal ratio, respectively. Maximal ratio is better than equal gain, as expected. Channel 1 is much better than channel 2, whose rank is beyond our computational power.

Profiling *Device F* in a Convenient Setup. The attack using maximal ratio succeeds also with a profile (10000 · 500 traces) built at 1 m in an office environment with a directional antenna (same setup as in Section 6.1.3). Since profiling and attack traces were collected with a different configuration for the trigger, we need to manually realign them by applying a fixed offset. The rank increases from 2^{27} to 2^{30}.

6.1.2.2 1.6 m *in Home Environment*

We increase the distance to 1.6 m in a home environment, where the target and the radio are close to walls, tables, and other objects that may reflect. We leverage spatial diversity to improve the results. Figure 6.6b shows the setup, in which we collected $20000 \cdot 500$ attack traces.

Profiling *Device E* in a Convenient Setup. We use the same profile at 10 cm as Section 5.4.3 and Section 6.1.2.1. Using the traces combined from the two channels with the maximal ratio technique, a profiled correlation attack succeeds after enumeration up to 2^{31}. With the traces combined with equal gain, enumeration succeeds at 2^{35}, whereas channel 1 requires 2^{31} and channel 2 needs more than 2^{34} (then we time out).

Profiling *Device G* in a Convenient Setup. The attack using maximal ratio succeeds also with a profile ($10000 \cdot 500$ traces) built on another device connected via cable (see Section 6.1.3). Since profiling and attack traces were collected with a different configuration for the trigger, we need to manually realign them by applying a fixed offset. The rank increases from 2^{31} to 2^{34}.

6.1.3 More Challenging Distances

One of the distinctive features of *screaming-channel* leakages is that they can be observed at large distance. We study from how far we can attack in an office environment. A simplified scheme of the setup is shown in Figure 6.6c.

6.1.3.1 10 m *in Office Environment*

In an office environment, we set target device and radio receiver 10 m apart. To avoid reflections, and to increase the gain, we use a directional antenna with 24 dBi gain [266], and two low-noise amplifiers with 20 dB gain and 1.5 dB noise figure [175]. This is the same setup used in Chapter 4 for the anechoic chamber. We additionally test three different connections with the *USRP N210* radio: direct Ethernet cable, connection through a switch, and connection through the Ethernet network of the building. In the last case we experience packet loss due to the speed of the network, but we are still able to collect traces.

Leakage Detection. As a first step, we evaluate the presence of the leakage with a t-test. Following [65], we choose a fixed vs. fixed configuration in order to maximize the leakage signal, and consequently reduce the number of traces required for detection. However, because of the distortion seen in Chapter 5, the maximum difference appears for $p \oplus k = 0$ and $p \oplus k = 48$, instead of $HW[p \oplus k] = 0$ and $HW[p \oplus k] = 8$. Therefore, we collect two sets $\mathcal{L}_{p \oplus k=0}$ and $\mathcal{L}_{p \oplus k=48}$ of $400 \cdot 500$ traces each. To be precise, we run 1000 encryptions and select the first 500 of them. This ensures we always average 500 traces despite sudden sources of noise, network delays, holes between packets, and other reasons for which we might miss the extraction of some encryptions. The number 500 tells us about the actual number of traces we need at 10 m, whereas the number 1000 give us a rough idea of how much harder it is to extract traces in a more complex environment or setup. We consider that the t-test successfully detects a leakage when the maximum value $\max(|t|)$ has $p < 10^{-5}$. With a few tens of traces the leakage appears for each type of connection, as shown in Figure 6.3a. Detection is slower for the connection through the network of the building, most likely because of data loss. Figure 6.3b shows that the leakage at the output of the first Sbox is clearly visible.

Attack Set for *Device F* at 10 m. Once we have confirmed that the leakage is visible even at 10 m in an office environment, we collect an attack set for *device F*. We acquire $6000 \cdot 500$ traces (actually 1000 encryptions for each trace, as for detection), connecting to the radio through a switch. Collecting a profile set in this scenario would be complex and it would require a large amount of traces to overcome noise. Fortunately, as seen in Section 5.4 and Section 5.5, profiles can be reused across distances, setups, and against different devices. Therefore, we can collect the profile set in more convenient conditions, and reuse it.

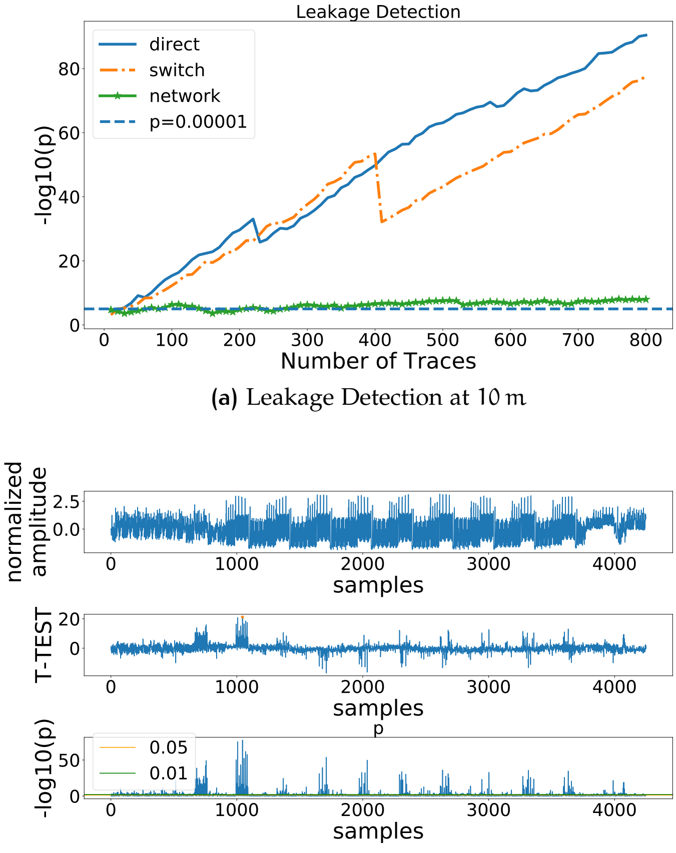
Profiling *Device G* in a Convenient Setup. We collect a profile set on another instance (*device G*), which we have modified for direct connection via cable with a *HackRF* radio. The main advantage of the cable connection is that we can pay much less attention to the noise in the environment. Moreover, using a simpler radio and no external amplifier greatly reduces costs and complexity. In the end, a laptop with two USB ports is enough to collect $10000 \cdot 500$ traces for the profile in a few hours.

Successful Attack. With a profiled correlation attack (leakage variable $p \oplus k$, average of the profiles of the 16 bytes, one POI) and key enumeration up to 2^{28} we can recover the key using $1500 \cdot 500$ attack traces at 10 m.

6.1.3.2 15 m *in Office Environment*

We increase the distance to 15 m, using the same setup and configuration as at 10 m.

Attack Set for *Device A* at 15 m. We collect $7000 \cdot 500$ traces (actually 1000 encryptions for each trace, as before). We observe that



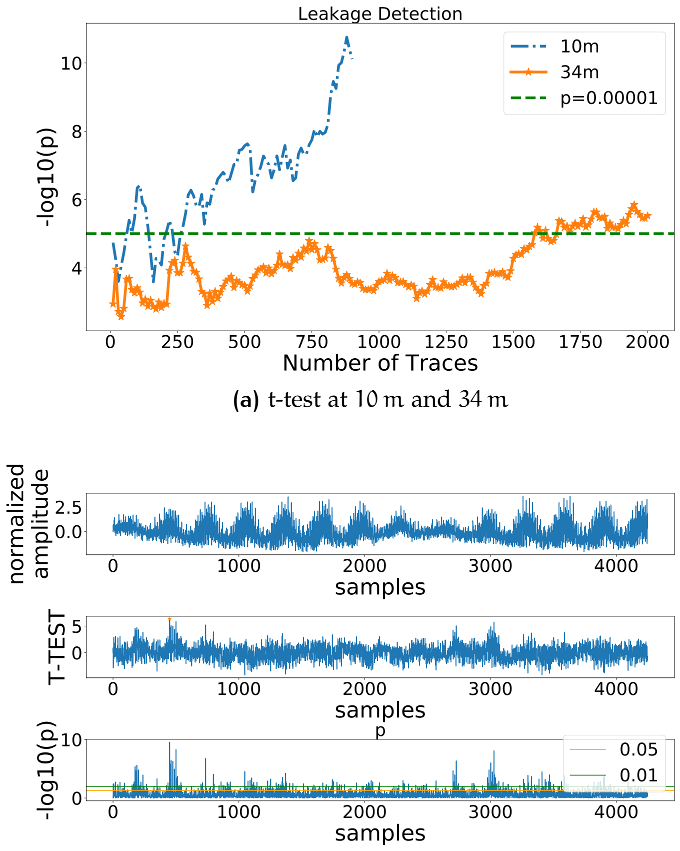
(a) Leakage Detection at 10 m

(b) t-test at 10 m

Figure 6.3: At 10 m we can easily detect the leakage with all types of connections (a). The t-test has peaks for all data-dependent points, including the output of the Sbox, which can be used to attack (b).

a meticulous tuning of the setup is particularly important in these conditions. We need to calibrate the *USRP N210* radio right before the acquisition campaign, and to position the directional antenna with care. It took us several attempts before managing to collect a set of traces good enough for an attack. On the contrary, at shorter distances *Screaming Channels* are easy to mount with simpler and cheaper hardware, without need of extremely accurate tuning. We conclude that, as expected, *screaming-channel* attacks can be easier or harder to mount than conventional attacks, depending on distance and environment.

Profiling *Device G* in a Convenient Setup. As before, we assume that the attacker can profile a different device with a convenient connection by cable, using $10000 \cdot 500$ traces. These profiling traces were collected 5 months and 15 days before the attack traces. To sum up, profiling and attack traces differ in device instance, propagation medium and distance, antenna and receiver type and quality, and acquisition time.



(a) t-test at 10 m and 34 m

(b) t-test at 34 m, $4000 \cdot 500$ traces

Figure 6.4: The t-test succeeds even at 34 m.

Successful Attack. Using profiled correlation (leakage variable $p \oplus k$, average of the 16 bytes, one POI), the attack succeeds with $5000 \cdot 500$ traces and key enumeration up to 2^{23}.

6.1.3.3 34 m *in Office Environment*

After attacking at 15 m, we increase the distance to 34 m, keeping the same setup and configuration. In this case we connect the laptop and the *USRP N210* through the Ethernet network of the building (at the price of some packet loss). We conduct the same fixed vs. fixed t-test experiment as we did at 10 m. The leakage is still visible and the t-test succeeds, as shown in Figure 6.4.

6.1.3.4 60 m *in Office Environment*

The detection of leakage traces in the radio signal is based on a band pass filter, tuned into a characteristic frequency component of the leakage trace. Since this component has a very narrow band, we can use a very narrow filter. For example, for the setup at 10 m we use a 6th-order Butterworth filter with cutoff frequencies of 1.95 MHz and

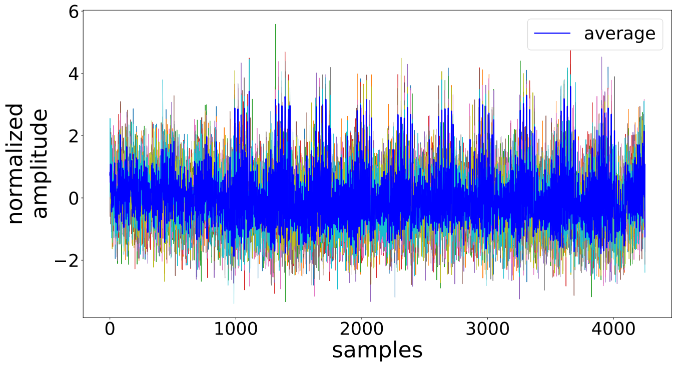


Figure 6.5: 10 · 500 traces and their average, at 60 m in an office environment. The *AES-128* encryption is still clearly visible.

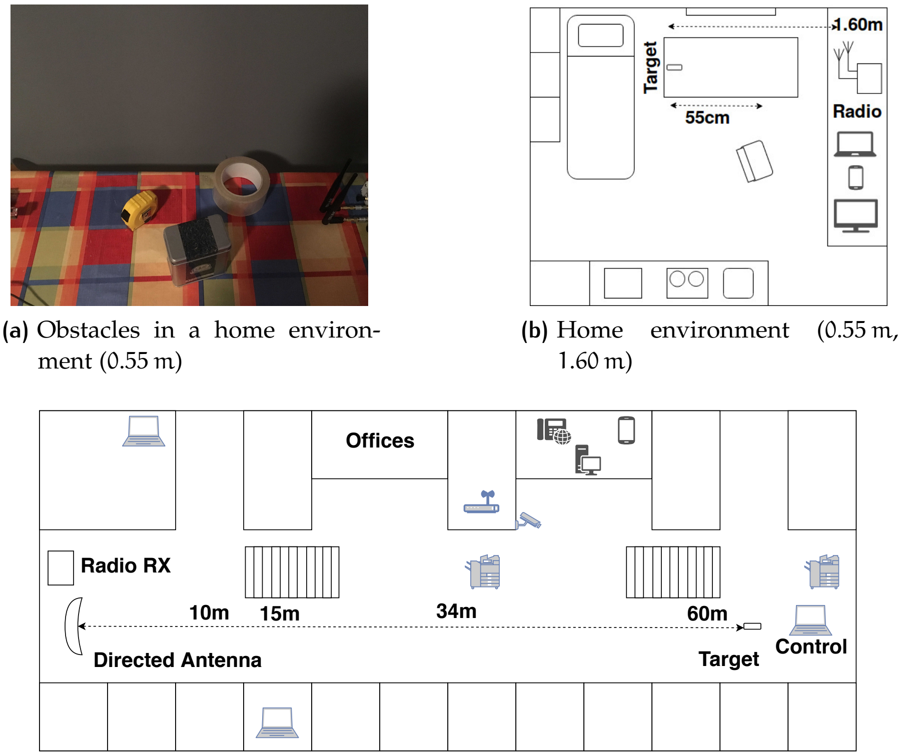
2.02 MHz. The noise in such a small bandwidth will be rather small. As a consequence, trace detection is quite robust to noise, and we expect to be able to extract traces even at distances larger than 34 m. Keeping the same setup, we place the target at 60 m from the antenna. Figure 6.5 shows 10 · 500 traces and their average. The key schedule and the 10 rounds of *AES-128* are clearly visible.

6.1.4 Summary of Results

In this section we have leveraged our analysis and attack improvements in order to investigate challenging scenarios. First, we have shown the level of correlation for optimized code and hardware *AES-128*, which gives an idea of the complexity of an attack. Second, we have used spatial diversity to attack in a real home environment, with obstacles and other sources of reflections. Finally, we have mounted a full attack at 10 m and 15 m in a real office environment, by combining the profiles of multiple bytes and with efficient key enumeration. We were able to reuse profiles built in more convenient setups on a different device than the one under attack. We could detect the leakage at 34 m with a fixed vs. fixed t-test, and extract traces at 60 m, still in a real environment. These are all significant steps towards realistic exploitation of the *screaming-channel* vector.

6.2 PROOF-OF-CONCEPT ATTACK AGAINST A REAL SYSTEM

The purpose of this chapter is to mount successful attacks in challenging and realistic conditions, after having uncovered the properties of the channel in Chapter 5. Our results could be the basis for a preliminary analysis of the risks associated with *Screaming Channels* in real



(a) Obstacles in a home environment (0.55 m)

(b) Home environment (0.55 m, 1.60 m)

(c) Office environment (10 m, 15 m, 34 m, 60 m)

Figure 6.6: Some more challenging attack setups.

systems and protocols. To provide a concrete example, we study the case of authentication for Google Eddystone beacons, and we show a proof-of-concept attack.

6.2.1 Google Eddystone Beacons

BLE beacons are devices designed to be placed at strategic places and continuously transmit some information on the advertising channels. People passing nearby can receive such messages on their mobile phones. Beacons can be used for numerous applications, ranging from marketing in shopping malls to indoor location services. For example, they can be placed near a monument and broadcast the URL of a webpage with information about it. There exist several beacon formats, such as, Apple iBeacons, Radius Networks AltBeacons, and Google Eddystone.

Eddystone [98] is a beacon format introduced by Google in 2015. Its design includes some interesting security features that ensure confidentiality and integrity of telemetry data, protect from spoofing and tracking, and require authentication to access and configure the beacon. One of their uses is to build the Physical Web [99], where objects broadcast a URL. There are four frame types:

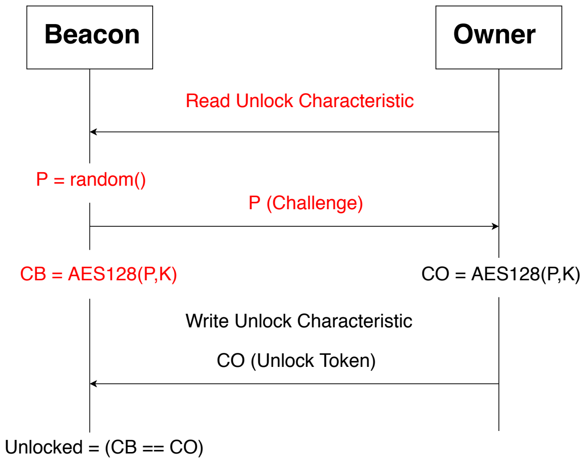


Figure 6.7: Unlock scheme used by Google Eddystone Beacons. Note that a non-authenticated device can trigger encryptions with known plaintext by reading the unlock characteristic (steps marked in red).

- **Eddystone-UID.** The identifier of the beacon.
- **Eddystone-URL.** The frame used to broadcast a URL.
- **Eddystone-(e)TML.** The frame used to broadcast (encrypted) telemetry data.
- **Eddystone-EID.** An ephemeral identifier that can be resolved only by a registered service, whereas it looks random to others.

To configure a beacon (e.g., to register an EID), the owner has to connect to it using another BLE device, and use the Eddystone Configuration GATT Service. Some beacons go in the connectable state after some kind of input (e.g., a button), other beacons have to be always connectable (e.g., if they are not easily accessible). Before being able to interact with the beacon, the owner has to unlock it using a 128-bit key (unlock code). The unlock method is shown in Figure 6.7. The device that wants to authenticate reads the unlock characteristic of the beacon. The beacon generates a random challenge that it sends back. Then it creates an unlock token by encrypting the random challenge with a preshared unlock code. Similarly, the owner creates the unlock code on the other side, then it writes it in the unlock characteristic. Finally, the beacon checks if the unlock code computed by the device that wants to authenticate is correct, and if so it opens the lock. The advantage of this method is that the unlock code is never sent in plaintext over the air. An attacker can read the unlock code, but it is a useless information because a new challenge is generated at each new unlock request. The disadvantage is that a device that connects to the beacon can trigger encryptions with known plaintext, which is a good scenario for side-channel attacks.

6.2.2 Proof-of-concept Attack Against Google Eddystone Authentication

We observe that the Google Eddystone unlock method is a good target for *screaming-channel* attacks.

6.2.2.1 *Threat Model*

Let us imagine the following scenario. A fleet of beacons is located in a position which is not easily accessible. For this reason, and to easily configure many beacons without manual access, they are configured to always stay in connectable mode. To avoid malicious reconfiguration (e.g., broadcast of a malicious URL), they are protected with a lock. The unlock scheme lets an attacker force many encryptions with known plaintexts, which is a favorable scenario for side-channel attacks. However, to exploit a conventional side channel, an attacker would have to get physical access to the devices, which are in a non easily accessible place, maybe even in a protected area such as a shopping mall. For these reasons, side-channel attacks are not taken into account in the threat model, and the unlock encryption is not protected. A countermeasure against side channels would also increase the cost of the device. This is a reasonable assumption for this kind of devices.

While conventional side channels require physical access, a *screaming-channel* attack could be performed discretely from a certain distance. This is a new threat to take into account.

6.2.2.2 *Real Target with Optimizations and Frequency Hopping Enabled.*

For this proof-of-concept attack, we target the Google Eddystone demo available in the manufacturer's SDK [190]. In this version, the encryption is performed by default with the `tinyAES` library compiled with optimization level O3[1]. We take the demo as is, without any modification, and we flash it on a *PCA10040* board. This is a realistic target with optimized code, frequency hopping enabled, and higher layers and protocols built on top of BLE.

6.2.2.3 *Trace Collection*

Using a mobile phone app, we simulate the owner of the device and we configure it to be always connectable, then we lock it. We can also set the transmission and advertising power to 4 dBm. We extended

[1] More recent versions of the SDK allow choosing other options such as hardware encryption, which is the default. However, we still think that software encryption with `tinyAES` is a reasonable and interesting example which might be found in the wild. In general, application code might prefer software implementations for portability, or to avoid competing with the link layer for the use of the hardware encryption block.

Listing 6.1: Minimizing frequency hopping.

```
> # Minimize Frequency Hopping
> sudo hcitool lecc --random E5:8F:A5:90:43:80 # connect
> sudo hcitool cmd 0x08 0x0014 0x0000000003 # set channel map
> sudo hcitool ledc 64 # disconnect
>
> # Double Check
> sudo btmon # monitor hci (in another terminal)
>
> sudo hcitool lecc --random E5:8F:A5:90:43:80 # connect
> sudo hcitool cmd 0x08 0x0015 0x40 # read channel map
>
> # Excerpt of the Expected Output (btmon):
> # LE Read Channel Map (0x08|0x0015) ncmd 1
> #   Status: Success (0x00)
> #   Handle: 64
> #   Channel map: 0x0300010004
> #     Channel 0-1
> #     Channel 16
> #     Channel 34
>
> sudo hcitool ledc 64 # disconnect
```

the collection code to be able to interact with any Eddystone beacon using the BLE card of the computer or an external off-the-shelf BLE dongle. The code can set a new random key, and then lock the device. It can also force many encryptions with known plaintext by repeatedly reading the unlock characteristic and saving the answer.

Minimizing the Frequency Hopping Problem.

In the scenario we consider, the attacker initiates the BLE connection to the target device. Therefore, the attacker can use the channel map to minimize the problem of frequency hopping by reducing the number of channels actually used for communication, as seen in Section 5.2.3. Following the BLE [228] specifications, we craft a *Set Host Channel Classification Command* to block all data channels but $0, 1, 16, 34$. On the Linux host of the attacker, we issue this command to the BLE dongle that initiates the connection using the *hcitool* program, as shown in Listing 6.1. Once set, the new channel map will be valid for the following connections as well.

Extracting Encryption Traces. We start triggering encryptions in the beacon by reading the unlock characteristic. To have an idea of the leakage trace to expect, we first extract the encryption trace with a conventional setup at 64 MHz. Here we observe a trace that is similar to Figure 6.1a, which is around 46 µs long. We then move to a *screaming-channel* setting. To minimize interference from other transmissions in the ISM band, we tune at the second harmonic of the clock after channel 34: $f = 2.474\,\text{GHz} + 2 \cdot 64\,\text{MHz} = 2.602\,\text{GHz}$. By reading the firmware, we observe that the encryption happens quickly after the read request to the unlock characteristic and the generation of the random unlock challenge. The response to the other device (containing the random challenge) is sent only after the encryption is complete, therefore we cannot expect that packet to contain any useful trace. We have to rely on other packets transmitted at the same time of the encryption. We notice that the device always sends a packet just after receiving the unlock characteristic read request. This might be the acknowledgement packet generated at the link layer. The encryption and this packet are almost synchronous. Most of the time the key schedule is visible in the leakage towards the end of the packet, as shown in Figure 6.8a. On a certain number of packets the first *AES-128* round is visible as well. We extract this signal containing the key schedule and the beginning of the first round, and we use it as template to collect more. Every time we trigger one encryption, we extract the leakage corresponding to the packet, and then we correlate it with the template. If correlation is high enough to confirm that we found the desired signal, we align it and save it. Note that in this case we cannot use time diversity, and we collect single encryptions.

6.2.2.4 *Key Recovery Attack*

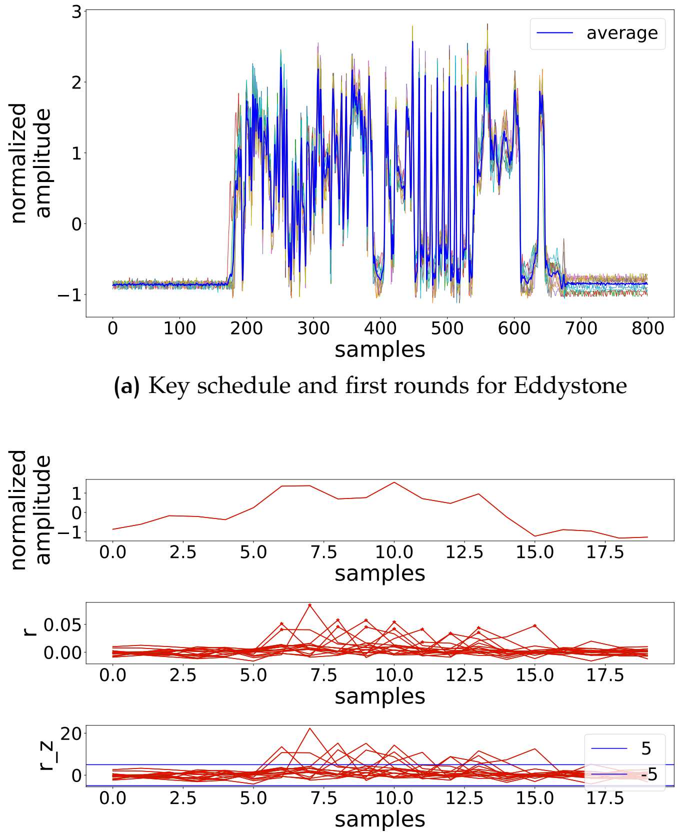
Once we have confirmed that we can extract encryption traces with known plaintext, we move to the attack phase.

Profiling *Device H* in a Convenient Setup. For simplicity, we connect to *device H* via cable, and we collect $70000 \cdot 1$ profiling traces with random plaintext and a fixed random key, over a period of around 3 days. Correlation peaks for $y = p \oplus k$ are clearly visible in Figure 6.8b.

Attacking *Device H* in a Convenient Setup. We collect $33000 \cdot 1$ traces. We run a profiled correlation attack with leakage variable $p \oplus k$. To obtain a better result, we replace the trace point corresponding to a POI with the average of the three points around the POI. This is a simple way to capture the information present just around the POI, without resorting to more complex multivariate attacks. We can recover the key with key enumeration up to 2^{30}.

6.3 COUNTERMEASURES

Smart devices usually consider physical side channel attacks outside of their threat model. However, *Screaming Channels* show that EM side channels are possible even without close proximity to the target. This novel threat requires consideration, and the development of countermeasures. These protections must be adapted to the nature of smart devices, which, unlike security-oriented systems (e.g., smart cards), have a very limited budget for security.



(a) Key schedule and first rounds for Eddystone

(b) ρ-teset for Eddystone

Figure 6.8: The key schedule and the first rounds of *AES-128* are clearly visible in the traces we collect for Eddystone (a). The correlation peaks for $p \oplus k$ are clearly present as well (b).

6.3.1 Conventional Countermeasures

Screaming Channels are equivalent to power and electromagnetic side channels, with the difference that the leakages are visible at large distance. Conventional countermeasures against DPA (e.g.,[129, 162]) are effective against *Screaming Channels*, too. However, the additional design or licensing cost might exceed the very tight budget available for the product to protect.

6.3.2 Specific Countermeasures in Hardware

Another strategy consists in trying to reduce the root problem behind *Screaming Channels* at the physical level. Again, the increased design and testing cost could be prohibitive. However, research in this direction is important in a long-term perspective, to reduce the risk that future devices accidentally present very strong and dangerous *screaming-channel* effects. Designers should be aware that noise coupling can bring security problems, besides the known functional issues.

There exist several design techniques that help reduce noise coupling, for example, guard rings, various substrate modifications techniques or even active noise cancellation techniques [5]. In addition, more isolation between components is possible with System in Package (or System in a Package) (SiP) technologies [151], which integrate multiple dies inside one package. Covering these electronic design strategies in detail is outside of the scope of this thesis. It would be interesting to study whether fully-digital radios are less susceptible to *Screaming Channels* than conventional architectures.

6.3.3 Specific Countermeasures in Software

The most interesting class of countermeasures tries to reduce the exploitability of *Screaming Channels* taking simple clever actions in software. Offloading encryption to a dedicated hardware block[2] is likely to make attacks more complex. Indeed, as of now we were not able to attack the hardware AES block of the *nRF52832* chip, even if we found interesting research directions to attack memory transfers of secret material. A problem of this solution is that code using dedicated hardware blocks is less portable. Another countermeasure consists in making sure that the radio is not transmitting any packet while software is running a sensitive computation. This solution is simple, cheap, and effective. However, depending on the application, this solution might be hard to implement in presence of strong timing constraints, or complex software stacks.

For the specific case of Google Eddystone beacons, the authentication protocol could limit the number or speed of failed attempts. However, this solution could be leveraged by an attacker for denial of service attacks. If the device is easily accessible by the legitimate user, it could be put in non-connectable mode unless a button is pressed, to prevent the attacker from connecting over the air.

6.4 CONCLUSION

In this chapter we have presented several attacks in challenging conditions. In particular, we have demonstrated successful attacks at large distance, in real environments, with profile reuse. We have also shown a proof-of-concept attack against a real system, and discussed several countermeasures. In the following chapters we will focus on a different attack: *Noise-SDR*.

[2] As we have seen before this is the default in the newest SDKs from Nordic Semiconductor.

Part IV

NOISE-SDR

We conjecture that an attacker might be able to use the electromagnetic leakages produced by unprivileged software to interact with radio receivers on the same platform as the processor, or nearby. Even though many example of leakage modulation for data exfiltration exist in Soft-TEMPEST literature, they are generally limited to very simple modulation schemes with custom parameters and protocols. Instead, we propose to leverage a 1-bit coding technique to shape arbitrary radio signals from electromagnetic noise that can be only turned on and off by software. We call this method '*Noise-SDR*' because it effectively builds a Software Defined Radio from electromagnetic noise. We implement and evaluate *Noise-SDR* on several smartphones, showing examples of transmission, tracking, leakage detection, and interaction with radio receivers on the same platform.

7 MOTIVATION, DESIGN, AND IMPLEMENTATION

In this chapter, taken from [36], we present the design and implementation *Noise-SDR*, a method to shape the noise generated by unprivileged code running on a smartphone into a generic radio signal.

7.1 INTRODUCTION

The execution of software and other activities on electronic devices often triggers the emission of electromagnetic leakages. The idea to intentionally control these signals to exfiltrate data from a computer was first introduced in 1998 under the name of Soft-TEMPEST [11, 149]. It consists in displaying a specially-crafted pattern on the screen to modulate the video signal. The resulting leakage at the frequency of AM radio can be easily picked up with a standard handheld device. A notorious application of Soft-TEMPEST is 'Tempest for Eliza' [273], a tool that transmits Beethoven's 'For Elise' as a sequence of tones, generated by a computer screen.

Over the years, data exfiltration from air-gapped networks using software-controlled leakages has received great attention in literature. Besides using electromagnetic [113, 114, 118, 303], magnetic [112, 117, 170], and electrical [123, 246] signals, researchers have also studied optical [111, 116, 122, 243], vibrational [109, 125, 261], acoustic [42, 110, 115, 120, 121], and thermal [108] emissions. A detailed survey of these covert channels is provided in [41, 43].

Even though researchers have explored a large number of leakage vectors, the modulation schemes and the protocols used for transmission have instead always been kept very simple. The root cause is that software has very limited control over the underlying leakage, for example, it can turn it on and off.[1] As a result, in the vast majority of cases the modulation schemes are limited to the families of OOK and FSK, with custom parameters and simple custom protocols. Sometimes, multiple threads are used to generate multiple OOK [117] or FSK [110] subcarriers, as a simple form of OFDM. This reduced modulation capability is a considerable limitation, especially for electromagnetic leakages. Indeed, it prevents the use of advanced radio communication techniques and existing protocols, and it constrains the possible use cases to simple exfiltration channels.

[1] The only exception are covert channels using an output device such as the speakers (e.g.,[42]), but we could argue that this is an intended analog signal output, rather than an unintended leakage triggered from software.

To overcome this problem, we take a radically different approach to modulation. Previous work modulates the leakage with the data bits to exfiltrate, for example, turning the leakage on and off to send logic '1's and '0's. Instead, we first modulate the data that we want to transmit on an arbitrary baseband signal using an existing SDR tool, for example, GNURadio. Then, we 'embed' this arbitrary signal into a binary signal using a 1-bit coding scheme. Finally, we use this binary sequence to turn on and off the leakage. As a result, we are able to transmit an arbitrary radio signal, modulated in amplitude, frequency, and phase. In particular, we use the RF-PWM [105, 150, 193, 217, 287] technique, borrowed from the theory of fully-digital radios, and adapted to the Soft-TEMPEST case.

We implement this idea on modern Arm smartphones using the leakages produced by fast DRAM accesses. We call it *Noise-SDR*, because it turns unprivileged software running on a smartphone into a generic SDR, able to shape an arbitrary radio signal from electromagnetic noise. We choose DRAM leakages because they have been proven to be a strong leakage source on x86 in [303]. Although considerably more flexible than classic Soft-TEMPEST transmitters, *Noise-SDR* has some hardware constraints in terms of frequency and bandwidth. The frequency range is limited to the harmonics of the underlying leakage, for example, all the multiples of an 800 MHz DRAM clock. The bandwidth is limited by the speed at which the leakage can be controlled. *Noise-SDR* can reach a bandwidth of tens of kHz on ArmV7-A, and a few MHz on ArmV8-A.

With *Noise-SDR*, we radically increase the degrees of freedom that attackers have, opening a whole new set of opportunities. Advanced radio techniques become readily available to the attacker, who can leverage existing software-defined radio tools to implement existing and custom protocols. Without any effort, the transmission can be optimized for a given scenario (e.g., distance or speed) leveraging existing techniques.

The mobile nature of smartphones leads to additional attack opportunities, for example, device tracking. In addition, smartphones carry a large number of radio transceivers, and sensors. Attackers can use *Noise-SDR* as a RFI *aggressor* to jam, spoof, or otherwise affect one of these *victim* components on the target device.

Table 7.1 compares *Noise-SDR* with existing (electro)magnetic channels.

Table 7.1: Comparison of software-controlled Electromagnetic and Magnetic leakages.

Name	Type	PHY	Protocol	Applications
Noise-SDR	EM	Arbitrary (RF-PWM)	Arbitrary (e.g., AM, NBFM, *PSK31*, *2xPSK500*, *MFSK128*, RTTY, *Olivia*, SSTV, *HamDRM*, *FT4*, *LoRa*, *GLONASS C/A Code*)	Smartphone as generic SDR, smartphone tracking, leakage identification, RFI/jamming/spoofing on the same device, etc.
Soft-TEMPEST [11, 149]	EM	AM, FSK	Custom	Exfiltration (display to AM radio)
AirHopper [114, 118]	EM	FSK (AFSK, DTMF)	Custom (raw or packet)	Exfiltration (computer screen to smartphone)
USBee [113]	EM	FSK (BFSK)	Custom	Exfiltration (USB bus to SDR)
GSMem [119]	EM	OOK (Binary Amplitude Shift Keying (BASK))	Custom	Exfiltration (computer to mobile phone)
BitJabber [303]	EM	OOK, FSK (MFSK)	Custom	Exfiltration (computer to SDR)
MAGNETO [112]	Magnetic	OOK, FSK (BFSK)	Custom	Exfiltration (computer to smartphone)
ODINI [117]	Magnetic	OOK (ASK, OOK-OFDM using multiple cores), FSK	Custom	Exfiltration (computer to magnetic bug)
Matyunin et al. [170]	Magnetic	OOK, FSK (‘period based’)	Custom	Exfiltration (laptop to smartphone)

7.2 noise-sdr

We describe the structure of *Noise-SDR*, and how it can shape arbitrary radio signals from electromagnetic leakages.

7.2.1 Noise-SDR

Figure 7.1 shows an overview of *Noise-SDR* and how it can be used as part of a standard radio communication chain. Assume, for example, that we want to transmit the message '*Hello!*' to a radio receiver, using a given radio protocol. For this, *Noise-SDR* has to generate a radio signal at frequency f_c, and to modulate it with an arbitrary baseband signal.[2] A standard tool (e.g., GNURadio, FLDigi) can be used to generate an arbitrary baseband signal. The tools can handle both the PHY layer and the upper layers of the chosen protocol. The RF-PWM stage of *Noise-SDR* generates a square wave at frequency f_0, modulated by the baseband signal. The square wave can be seen as the sum of several sinusoidal waves, the first of which is called fundamental component and has frequency f_0. The amplitude, phase, and frequency of the fundamental component can be controlled by changing the width, frequency, and phase of the pulses of the square wave.[3] The square wave is the on/off sequence for the next block. To generate a physical electromagnetic signal, *Noise-SDR* executes a leaky operation in software, for example, accessing DRAM. The resulting leakage has an harmonic at frequency F_l. *Noise-SDR* can turn it on and off by executing the leaky operation or not, under the control of the square wave coming from the RF-PWM block. As a result of the intermodulation between the leakage and the fundamental component of the square wave, *Noise-SDR* generates a sinusoidal signal at frequency $f_c = F_l + f_0$, modulated by the arbitrary baseband signal.[4] Note that also the other components of the square wave are transmitted (e.g., $F_l + 2f_0$). However, if f_0 is big enough, they fall far away from f_c and they can be safely ignored, as if they were not part of the transmission. Indeed, the receiver filters out any signal which is not in the desired bandwidth around f_c. This is the common behavior of a receiver, and it is not specific to *Noise-SDR*. The signal propagates over a radio channel until the antenna of the receiver. The receiver filters the useful signal around the carrier frequency f_c. Once the signal is down-converted back to baseband, it can be decoded, retrieving the message '*Hello!*'. The receiver may or may not be software-defined.

[2] The modulated carrier at frequency f_c is $a(t)\cos(2\pi f_c t + \theta(t))$, where $a(t)$ and $\theta(t)$ are the amplitude and phase. The quadrature notation $I(t) = a(t)\cos(\theta(t))$ and $Q(t) = a(t)\sin(\theta(t))$ is the one used in practice.

[3] As shown in Chapter 2, the fundamental component of a square wave with frequency f_0, phase $\theta(t)$, and duty-cycle $\delta(t) = \frac{\arcsin(a(t))}{\pi}$ is $\frac{2}{\pi}a(t)\cos(2\pi f_0 t + \theta(t))$.

[4] The multiplication (intermodulation) of $a(t)\cos(2\pi f_0 t + \theta(t))$ with the leakage $\cos(2\pi F_l t)$ produces a component $a(t)\cos(2\pi f_c t + \theta(t))$, where $f_c = F_l \pm f_0$.

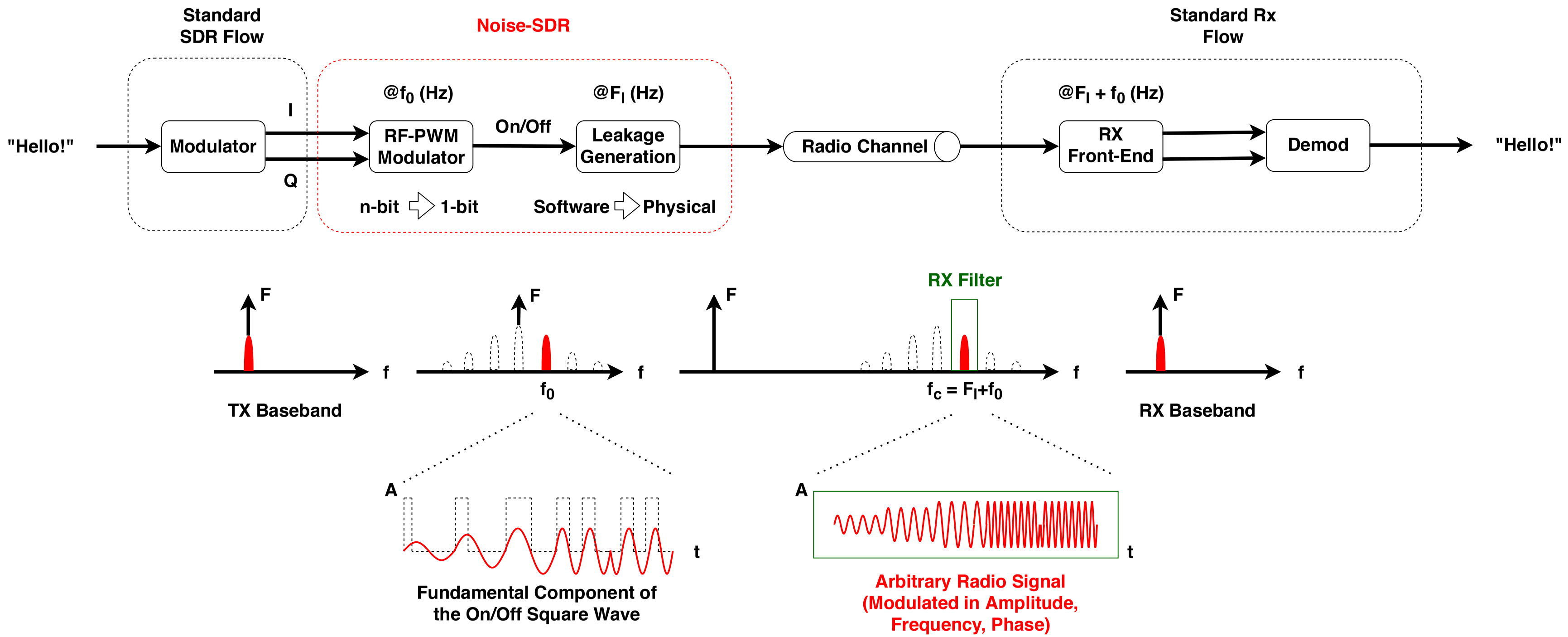


Figure 7.1: *Noise-SDR* is an SDR able to transmit arbitrary radio signals using software-controlled electromagnetic leakages. The input baseband signal can be generated with a standard SDR flow (e.g., GNURadio). The output radio signal can be received with a standard radio receiver chain. In this sense, *Noise-SDR* acts as a generic SDR peripheral, such as an *Ettus Research USRP B210*, but it is completely implemented in software. *Noise-SDR* can turn on and off an electromagnetic leakage by running some leaky software operation, such as a memory access. *Noise-SDR* modulates the amplitude, frequency, and phase of the radio signal by modulating the pulse width, frequency and phase of the on/off sequence. More precisely, *Noise-SDR* uses the RF-PWM technique to modulate the first harmonic of a square wave. All the others are well spaced in frequency, and they can be ignored because they will not be picked by the receiver. The intermodulation between on/off sequence (at frequency f_0) and the leakage (at frequency F_l) sets the carrier frequency to $f_c = F_l + f_0$.

7.2.2 The RF-PWM Stage

The RF-PWM block modulates an arbitrary baseband signal onto the first harmonic of the on/off square wave, following the theory described in Chapter 2.

Figure 7.2 shows a practical example of RF-PWM modulation. To show both amplitude and angle changes in the output signal, we choose an excerpt of a *HamDRM* transmission [248]. We first generate the baseband signal using existing tools. Then, using the RF-PWM technique, we modulate the on/off square wave. The fundamental component of this square wave is a sinusoidal wave at frequency f_0, modulated by the baseband signal. It is clearly visible in the last plot of Figure 7.2, with amplitude and phase changes. The full transmission with a *Samsung Galaxy S5 Mini* is visible in Figure 8.2 in Chapter 8.

Figure 7.3 shows an intuitive view of the implementation. After having generated a frequency and phase modulated square wave by taking the sign of a cosine wave, we adjust the pulse width using the pre-distorted amplitude. We output a list of timings that describe the on/off sequence. Algorithm 1 in Appendix a.2 provides a detailed description of the discrete-time implementation in practice.

7.2.3 The Leakage Stage

The leakage stage turns on and off a leakage at frequency F_l from software.

Noise-SDR can turn on and off a leakage by executing a '*leakyOperation()*' that triggers the emissions of the underlying hardware. An accurate timing source '*now()*' is used to control the on/off timings following the on/off sequence produced by the RF-PWM stage. Figure 7.4 shows this algorithm and the equivalent hardware model: a 1-bit DAC mixed with a frequency generator. Note that we use one (sinusoidal) harmonic of a generic leakage. For example, the square wave clock of the DRAM will produce a sinusoidal fundamental component at the clock frequency.

We identify the rationale behind the ideal properties of the leakage and timing sources:

- *leakyOperation()* must produce a strong leakage at a stable frequency F_l. Ideally, F_l must fall in the band of the transmission that we want to achieve.

- *leakyOperation()* must be short. The shorter the operation, the higher the resolution at which we can generate the RF-PWM on/off square wave. This has a huge impact on all the parameters of the generated signal. In particular, the shorter *leakyOperation()*,

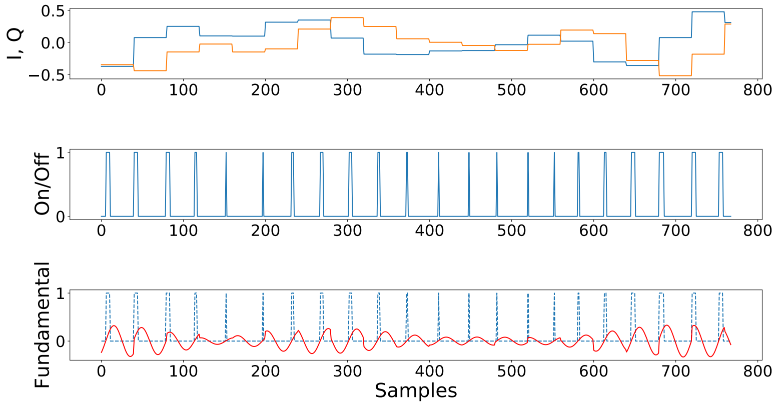


Figure 7.2: A practical example of RF-PWM modulation. The first plot shows the baseband signal. The second plot is the on/off sequence modulated in pulse width, frequency, and phase according to the baseband signal. The last plot shows the desired output (in red): the fundamental component of the square wave, modulated by the baseband signal. This example is an excerpt of a *HamDRM* transmission of a digital image, chosen because it shows both amplitude and phase changes in the signal.

the higher the bandwidth of the signal that we can generate (the sampling rate of our *Noise-SDR*).[5]

- *now()* should have a good resolution, for the same reason as above. Additionally, it should be as stable as possible, for example, for those protocols that require a stable frequency.

7.2.4 Comparison With Conventional Radios

RF-PWM has been demonstrated with real silicon implementations [105, 150, 193, 287]. Instead, *Noise-SDR* implements RF-PWM in software, and then uses it to pilot a software-controlled leakage. Figure 7.5 compares an equivalent hardware model of *Noise-SDR* with a conventional superheterodyne transmitter.

Like a superheterodyne transmitter, *Noise-SDR* generates a carrier frequency in two steps. First the baseband signal is modulated onto an intermediate frequency component, then it is further up-converted to radio frequency. For *Noise-SDR*, the intermediate frequency stage is the RF-PWM block that modulates the arbitrary baseband signal on the fundamental component at frequency f_0. The radio frequency stage is the leakage stage, which further up-converts the signal to frequency

[5] The time resolution sets the sampling frequency F_{res} at which we generate the square wave. Because of the sampling theorem, the highest f_0 is $F_{res}/2$. At its turn, the maximum bandwidth B of the baseband signal is $f_0/2$. In addition, the resolution of the amplitude $a(t)$ and phase $\theta(t)$ also depends on the resolution at which we can control the width and timing of the pulses. See Appendix a.2 for details.

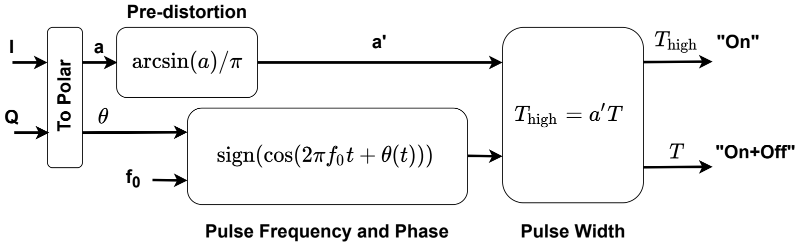


Figure 7.3: Generation of the on/off sequence with the RF-PWM technique. Intuitively, we generate a frequency and phase modulated square wave by looking at the sign of a cosine wave. Then, we adjust the pulse width to modulate the amplitude. We output a list of on/off timings. See Algorithm 1 in Appendix a.2 for the detailed discrete-time implementation.

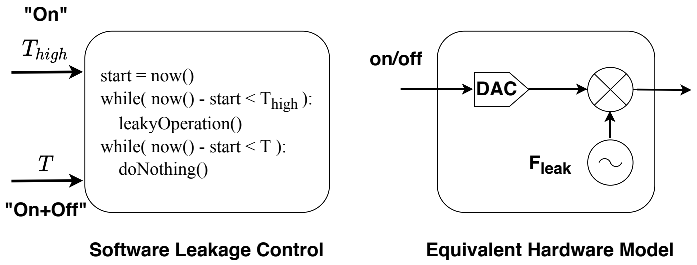


Figure 7.4: Algorithm to control the leakage from software, and equivalent hardware model. Differently from previous work, in *Noise-SDR* the on/off bits are not the data to transmit, but the bits of an RF-PWM square wave, whose fundamental component is an arbitrary signal that acts as IF stage.

$F_l + f_0$. With this architecture *Noise-SDR* can generate arbitrary signals from a leakage at fixed frequency.

Unlike conventional transmitters, *Noise-SDR* uses a 1-bit square wave as intermediate carrier. Only the fundamental component is considered, and all the other harmonics are ignored. In general, *Noise-SDR* does not apply any filter. All the spurious components are transmitted, but they can be ignored. They pollute the spectrum, but they do not impact the receiver. Finally, *Noise-SDR* does not use an amplifier.

7.3 IMPLEMENTATION

We now describe a modular implementation of *Noise-SDR*. Chapter 8 will provide an evaluation of these methods on several smartphones.

7.3.1 Modular Approach

As shown in Figure 7.6, we implement a generic RF-PWM block, and several independent leakage control units.

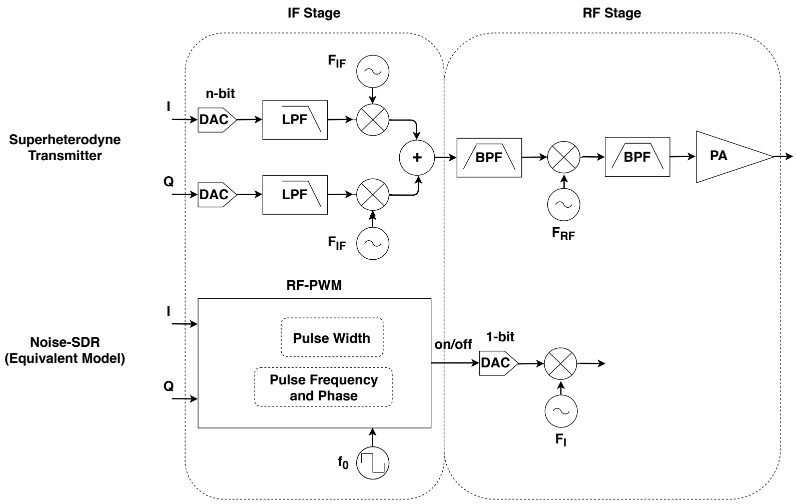


Figure 7.5: Comparison of *Noise-SDR* (equivalent model) with a conventional superheterodyne transmitter [16]. We use the RF-PWM stage as an intermediate frequency stage, and the leakage control stage as a radio frequency stage.

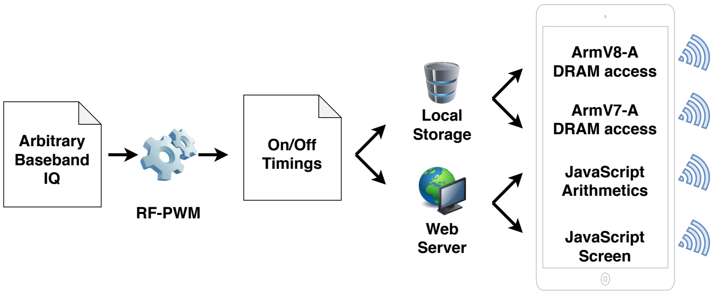


Figure 7.6: A modular implementation of *Noise-SDR*. The RF-PWM stage generates an on/off sequence, which can then be used to control different types of leakages.

The unprivileged *leakyOperation* that produces a short strong leakage at frequency F_l can be implemented in several ways, leveraging different leakages. For example, in this thesis we choose fast DRAM accesses, using DMA on ArmV7-A, and cache maintenance instructions on ArmV8-A. Leakages can also be produced with *JavaScript*, running arithmetic operations, or controlling a pattern on the screen. Each of these methods uses the algorithm in Figure 7.4 to turn on and off the leakage following an RF-PWM square wave. The timings of this on/off sequence are computed by a generic RF-PWM stage written in *Python* independently of the leakage source (Algorithm 1 is described in Appendix a.2). For native Arm code, the on/off sequence is precomputed and stored in */data/local/tmp/*. This is an advantage for use cases that require sending a fixed message (e.g., device tracking), because it simplifies the software that has to run on the smartphone. However, RF-PWM modulation is fairly simple, and it could be performed live on the mobile device.[6] For *JavaScript* code, the attack is different. The RF-PWM leakage is precomputed and stored in a web server together with the leakage control code. They are then executed in the browser when the page is visited.

7.3.2 Implementation with DRAM Accesses from Native Code

We present a complete implementation of *Noise-SDR* with unprivileged native code.

DRAM accesses are an excellent *leakyOperation* to trigger a short and strong electromagnetic leakage. DRAM emissions are known to impact the performance of other communication devices on the same smartphone because of electromagnetic and radio-frequency interference [179]. Offensive research on Rowhammer [81, 282, 304] has developed several techniques to access DRAM at high speed on Android phones running on Arm, without privileges, or even remotely. On x86, DRAM emissions have been exploited to detect Rowhammer attacks [305], and to build a fast covert channel [303].

The hammering step of Rowhammer attacks consists in repeatedly accessing a DRAM location at high speed, to flip a bit in a neighbouring line. To access the DRAM directly, the CPU has to somehow bypass the data cache. As explained in [304], ArmV8-A code offers an unprivileged instruction to clean and invalidate a virtual address in data cache. The attacker can use this instruction after each load. As a result, the next load will always be served from DRAM and not from the data cache. Listing 7.1 can be used in a loop to repeatedly access a location in DRAM with this technique. *DC CIVAC* is the instruction that cleans and invalidates an address in data cache. *DSB* is a memory barrier that was empirically found to improve hammering [304], but we did not observe a significant impact on DRAM leakage. We use Listing 7.1 as *leakyOperation* on ArmV8-A. It is fast (hundreds of ns), and it turns on a strong leakage (at the harmonics of the DRAM clock frequency).

On ArmV7-A unprivileged cache maintenance instructions are not available. However, direct DRAM access is possible on Android, using DMA (calling the *ION* allocator via the *ioctl* system call) [282]. We have empirically observed that the strongest leakage around the DRAM clock frequency can be triggered by allocating a small memory chunk with *ION* (followed by a free), as shown in Listing 7.2.[7] We use this as *leakyOperation* on ArmV7-A, but it also works on *ArmV7-8*. It is fast (tens of µs), but slower than Listing 7.1.

[6] Modern smartphones have enough resources to run popular SDR flows. For example, GNURadio has been ported to Android https://www.bastibl.net/gnuradio-on-android/, to control an external SDR.

[7] Only in a few cases (*Motorola Moto E6S*, *Google Nexus 5*), simply reading from an address in DRAM (allocated with *ION*) gives better results.

Listing 7.1: *leakyOperation* for ArmV8-A native code.

```
__attribute__((naked)) \
void hammer_civac(uint64_t *addr) {
  __asm volatile("LDR X9, [X0]");
  __asm volatile("DC CIVAC, X0");
  __asm volatile("DSB 0xB");
  __asm volatile("RET");
}
```

Listing 7.2: *leakyOperation* for ArmV7-A native code.

```
void ION_test(int len) {
  ION_free(ION_alloc(len));
}
```

We implement the leakage control block of Figure 7.4 in *C++*. We use *clock_gettime* to measure the timings of the on/off pulses at the ns resolution. In the absence of timers we could use calibrated loops as well. We load the RF-PWM signal from file and we store it into a static array for fast access. The code is compiled with optimization level *-O3* using the *clang* toolchain of the *Android NDK r21d*.

7.3.3 *JavaScript* and Other Sources

In this thesis we focus on DRAM accesses from native code, however, *Noise-SDR* can also be implemented with other languages and leakage sources. We show a few preliminary examples, even though we leave a deeper investigation for future work.

Soft-TEMPEST leakage from *JavaScript* have been demonstrated by [68] using the arithmetic operations in Listing 7.3, and *performance.now()* as timer.[8] We have implemented *Noise-SDR* using Listing 7.3 as *leakyOperation*. In this case, the available resolution is limited (hundreds of µs to a ms). The main problem is that timing measurements in *JavaScript* are made less accurate as a defense against side channel attacks [39, 144, 237]. Additionally, the leakage is weak and visible only on a limited number of phones.

It is well known from (Soft) TEMPEST literature that screens, HDMI/VGA cables, and connectors produce a leakage that can be controlled through the image on display (e.g., [11, 118, 149, 273]), also on mobile devices (tablets) [126]. In our very simple implementation of *Noise-SDR*, we draw horizontal stripes of different colors to generate the on/off pulses of the leakage. This can be done in *JavaScript* using

[8] https://github.com/fulldecent/system-bus-radio/blob/master/TEST-DATA.tsv lists the leakages reported by users.

Listing 7.3: Implementation of *leakyOperation* in *JavaScript*, inspired by [68].

```
for (var i = 0; i < 100; i++) {
  reg = 1 - Math.log(reg) / 1.7193;
```

Listing 7.4: Excerpt ot the *JavaScript* code that draws the RF-PWM signal in form of stripes on a web page, to control the electromagnetic leakages from the screen. The timings have been converted from ns to pixels by multiplying them with $fps \cdot displayHeight/10^9$.

```
while(px + T[i] <= displayHeight){
  rect(0,px, displayWidth,Thigh[i]);
  px = px + T[i];
  i = (i + 1) % data.length;
}
```

the *p5.js* library, as shown in Listing 7.4. A stripe of $fps \cdot H/10^9$ pixels corresponds to roughly $1\,ns$ of signal, where fps is the frame rate and H is the height of the screen. This approach is simple and it does not require to know the exact parameters of the screen (e.g., the pixel clock). For simplicity, we ignore the effect of the blanking intervals.

The display is not the only component connected with a bus that leaks. On a *Google Pixel XL* we have observed a strong leakage from the MIPI bus connected to the front camera. It can be easily modulated by pointing the camera at the same video that we sent to the screen in the previous paragraph. However, this case is not particularly interesting in practice.

Apart from the CPU, other blocks can access DRAM directly. For example, [81] has demonstrated remote Rowhammer attacks from the GPU using WebGL in *JavaScript*. We believe this is an interesting direction for future research, as we have observed strong leakages when rendering textures in *JavaScript*. However, reversing the GPU and controlling on/off timings would be more complex and less portable than the other methods that we have shown.

8 EVALUATION

In this chapter, taken from [36], we evaluate *Noise-SDR* with many interesting examples and use cases, on several smartphone models. We start by demonstrating the transmission of diverse protocols, each with different characteristics. Then we show how to optimize for distance or speed in a tracking scenario, and how to detect the leakages automatically. Finally, we show some examples of RFI aggression where *Noise-SDR* is used to inject signals in other components on the phone.

8.1 TRANSMISSION OF MANY DIVERSE PROTOCOLS

Noise-SDR is able to transmit existing protocols, without the effort of designing and implementing a custom solution. Attackers can leverage popular SDR tools to generate the desired baseband signals.

8.1.1 Motivation

We have tested a large number of protocols, listed in Table 8.1, to show different characteristics of *Noise-SDR*. Some protocols are analog, some are digital. They include examples of amplitude, frequency, and phase modulations.The signal of protocols that use orthogonal subcarriers has both amplitude and phase changes. There are two cases of spread spectrum: chirp and direct sequence. As for the bandwidth, it ranges from a few Hz, which requires a good clock stability, to half MHz, which requires a high sampling rate. In addition, some of these protocols use forward error correction (e.g., *FT4*), and upper layers with packets and addressing (e.g., *LoRa*). Each of these protocols is optimized for different scenarios. For example, *FT4* is designed to work at large distance below the noise floor, whereas *2xPSK500* is designed for a much higher baud rate (1000 bps). Finally, all these protocols are supported by popular SDR and amateur radio tools. The generation and reception of the signals does not require any effort and it is accessible to attackers without extensive knowledge about radio communications.

8.1.2 Experimental Setup

In this phase we use a generic reception setup, not optimized for any specific distance, frequency, protocol, or use case. We simply choose a

Table 8.1: Tested protocols.

Name	Description	Bandwidth	Reference
Voice AM	AM	10 kHz	[247]
Voice NBFM	NBFM	12.5 kHz	[249]
PSK31	BPSK, USB	31 Hz	[251]
RTTY45.45	BFSK, USB	170 Hz	[252]
MFSK128	MFSK, USB	1.928 kHz	[75]
Olivia 64/2000	MFSK, USB	2 kHz	[250]
2xPSK500	2 BPSK subcarriers, USB	1.2 kHz	[75]
SSTV	FM, USB	2.5 kHz	[253]
HamDRM	QAM, OFDM, USB	2.4 kHz	[248]
FT4	4-GFSK, USB	90 Hz	[268]
LoRa	CSS	Customizable (8 kHz)	[78, 226, 267]
GLONASS C/A Code	DSSS	0.511 MHz	[44]

good positioning for the antenna, usually close to the sides or back of the smartphone. The reception hardware is an *Ettus Research USRP B210 SDR* [71] with a *NAE-HPROBE-15 antenna* [186]. The SDR software is *Gqrx* [56], *GNURadio* [82], *Qsstv* [212], *FLDigi* [76], *WSJT-X* [269], and *gnss-sdr* [310]. In general we do not need external amplification, and often a standard *WiFi* antenna works at a few centimeters from the smartphone. Only for *GLONASS* on a *Innos D6000* (and for *LoRa* on a *Google Nexus 5*) we need a *TEKBOX TBWA2 amplifier* [263], powered via USB. For most smartphones the leakage is strong and stable when the screen is on. Since transmissions last a few seconds, this is a reasonable case. We connect to the smartphone via *adb*, generally via *WiFi* to avoid unintended coupling.

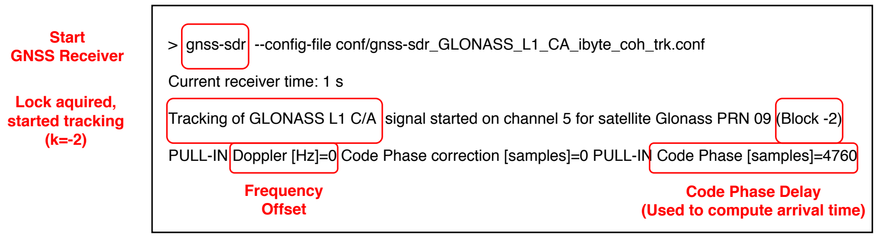
8.1.3 Results with DRAM

Table 8.2 shows the result of our experiments with 17 smartphones. Their architecture and LPDDR version are specified when available at *phonedb*[1]. We make the following three observations. First, many but not all the smartphones exhibit a leakage triggered and modulated by DRAM accesses. We report the main frequency and the harmonics visible with our setup. Second, when a leakage exists, *Noise-SDR* can use it to generate arbitrary signals of many types. Third, the frequency of transmission is determined by the frequency of the underlying leakage, which depends on the phone model, though only a few standard values exist. Finally, most amateur radio protocols fit in the bandwidth of ArmV7-A smartphones, whereas *GLONASS* requires ArmV8-A to reach a sampling rate in the order of MHz.

[1] http://phonedb.net/

Table 8.2: Transmission experiments with DRAM accesses.

Model	Architecture	LPDDR	Frequency	Harmonics n	Bandwidth	Tested Protocols
Innos D6000	ArmV8-A	3	400 MHz	$1-4$	few MHz	All but *HamDRM*, $n = 2$
Nokia 3.1 (TA-1063)	ArmV8-A	3	13.56 MHz (*NFC*)	7	few kHz	*PSK31*, $n = 7$
Samsung Galaxy A30S (SM A397FN)	ArmV8-A	4	1794 MHz	1	few MHz	*GLONASS C/A*, $n = 1$
Samsung S7 Exynos (SM-G930F)	ArmV8-A	4	1794 MHz	1	few kHz	Simple tunes and chirps
Samsung Galaxy S5 Mini (SM G800F)	ArmV7-A	n.a.	200 MHz	1-11,13-19,26	tens of kHz	All but *GLONASS*, $n = 1$
Samsung M31 (SM-M315F/DSN)	ArmV8-A	4	1794 MHz (rare)	1	few MHz	*NBFM*
Samsung Galaxy J7 (SM-J730FM)	ArmV8-A	3	None identified	-	-	-
Samsung Galaxy Young (GT-S631ON)	ArmV7-A	n.a.	None identified	-	-	-
Sony Xperia C5 (E5533)	ArmV8-A	4	400 MHz	1-11	few MHz	*NBFM*, *LoRa*, $n = 6$
Sony Xperia X (F5121)	ArmV8-A	3	None identified	-	-	-
Motorola Moto E6S	ArmV7-A	3	400 MHz	1,2	few kHz	Simple tunes and chirps
Google Nexus 5 (D821)	ArmV7-A	2	200 MHz	$1-5, 8, 12$ $16, 20, 24$	tens of kHz	*NBFM*, *MFSK128*, *FT4*, *SSTV*, *Olivia*, *LoRa*, $n = 1$
Google Pixel XL	ArmV8-A	4	None identified	-	-	-
Google Pixel 2	ArmV8-A	4	None identified	-	-	-
Wiko Fever	ArmV8-A	3	None identified	-	-	-
Huawei P8 Lite (PRA-LX1)	ArmV8-A	3	None identified	-	-	-
Huawei P10 (VTR-L09)	ArmV8-A	4	None identified	-	-	-
Huawei P8 SE (GRA-L09)	ArmV8-A	3	None identified	-	-	-
OnePlus 7 Pro PE (GM1913)	ArmV8-A	4	None identified	-	-	-



```
> gnss-sdr --config-file conf/gnss-sdr_GLONASS_L1_CA_ibyte_coh_trk.conf
Current receiver time: 1 s
Tracking of GLONASS L1 C/A signal started on channel 5 for satellite Glonass PRN 09 (Block -2)
PULL-IN Doppler [Hz]=0 Code Phase correction [samples]=0 PULL-IN Code Phase [samples]=4760
```

Figure 8.1: Reception of a *GLONASS C/A Code* with *gnss-sdr* and a *USRP B210 SDR*. The receiver locks (it finds the initial frequency offset and code phase delay) and then starts tracking. The signal was transmitted with an *Innos D6000* using the leak at 800 MHz plus the offset of satellite $k = -2$ ($f_0 = 875$ kHz). The harmonic at 1600 MHz is too weak for reception. We could also receive a *GLONASS C/A Code* sent by a *Samsung Galaxy A30S* at 1795.5 MHz.

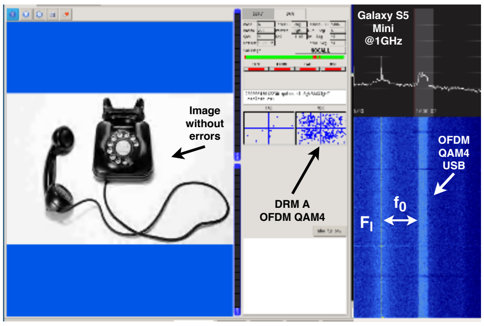


Figure 8.2: Example of digital image transmission using HamDRM (OFDM, QAM4, error correction) on a *Samsung Galaxy S5 Mini*. The signal, modulated in amplitude and phase, is sent in SSB at 1 GHz ± 9.6 kHz.

We highlight two interesting examples. First, both an *Innos D6000* and a *Samsung Galaxy A30S* can transmit a *GLONASS C/A Code*. Figure 8.1 shows how the *gnss-sdr* receiver locks into the code and starts tracking.[2] Second, a digital image can be transmitted without errors using *HamDRM* with a *Samsung Galaxy S5 Mini*, as shown in Figure 8.2. The transmitted signal is modulated both in amplitude and phase, and transmitted in form of frames with error correction. If an error occurs in a frame, the receiver can detect it and discard it. If the frame is sent again, even out of order, the receiver will consider it. As a result, if we transmit the image in a loop, in a few iterations we can send all frames without errors. This is a simple form of retransmission on error.

[2] For a description of how code locking and tracking works in most *GNSS* systems refer to [24, 136]. In summary, the receiver detects and synchronizes with the frequency offset and phase delay of the transmitted code.

Table 8.3: Preliminary transmission experiments in *JavaScript*.

Model	Leakage Type	Leakage Frequency	Protocols
Google Nexus 5	Arithmetics	Several carriers 32 MHz to 200 MHz	*PSK31* (not reliable)
Google Nexus 5	Screen	Several carriers 100 MHz band	*voice AM* (distorted)
Samsung Galaxy S5 Mini	Screen	Several carriers 100 MHz band	*PSK31* (not reliable)
Wiko Fever	Screen	100 MHz	Simple tunes

8.1.4 Preliminary Results with Other Sources

Table 8.3 shows some preliminary results of transmission using other leakage sources (arithmetic operations, screen) in *JavaScript*. Differently from DRAM, these leakages fall in the FM radio band (around 100 MHz). In this thesis we focused on implementing *Noise-SDR* with DRAM accesses, and further experimentation with *JavaScript* is left for future work.

We observe a leakage by placing a *NAE-HPROBE-15* probe on the back of a *Google Nexus 5*. There are several leakage frequencies between 32 MHz, and 200 MHz). These leakages can be modulated both from *JavaScript* (Listing 7.3) and from C++ with *ION* (Listing 7.2). Some preliminary experiments show that *PSK31* works, though not reliably.

8.2 DEVICE TRACKING

Since they carry many radio transmitters and other actuators, smartphones are often vulnerable to tracking without user consent. For example, WiFi [168, 180] and ultrasounds [13, 58, 173, 184] have been used to this purpose. We show how *Noise-SDR* can be used to implement a simple system to transmit wireless beacons that uniquely identify devices.

8.2.1 Motivation and Threat Model

We want to show some theoretical and practical advantages of our approach in an interesting scenario, raising awareness about the risks associated with advanced control over electromagnetic leakages. Without any design effort, an attacker can leverage an existing low-power physical layer, to achieve robust transmission at a reasonable distance. Similarly, higher layers could be used to identify the transmitter and send useful data. In our threat model, the user puts the phone in airplane mode to avoid being tracked (e.g., via *WiFi* beacons). However, the user has installed an applications that contains the few lines of code required to implement *Noise-SDR*.

8.2.2 Theoretical and Practical Considerations

Our transmitter is only capable of very weak emissions. In [226], existing Low-Power Wide Area Network (LPWAN) solutions and strategies are explained and compared in light of the information theoretic limit of channel capacity. Different methods can be used to increase the transmission distance without increasing the transmission power, including: reducing the data rate, reducing the bandwidth, using a spreading factor, using forward error correction, and using M-ary orthogonal modulations. Similar considerations are also valid for amateur radio protocols.

Protocols which use a very small bandwidth over a long transmission time are more sensitive to frequency offset variations during transmission. For example, the Weak Signal Propagation Reporter (WSPR) [270] uses a 6 Hz-wide signal that lasts 120 s. This requires hardware with a very good clock stability. Protocols with a higher data rate, for example *IEEE 802.15k*, use a larger bandwidth, and they require hardware with a high sampling rate.

8.2.3 Design

Taking into account the previous considerations, we opt for the *FT4* protocol [268] released in 2019. Designed for amateur radio contests, it has the following advantages:

- It works at very low SNR thanks to forward error correction and a small bandwidth Channel capacity is further increased by transmitting at fixed known time instants, using the Universal Coordinated Time (UTC).

- It uses a 4-tone GFSK signal, occupying a bandwidth of 90 Hz, and lasting 4.48 s. It is less sensitive to clock instability than WSPR, and it does not require a high sampling rate.

- It regularly transmits a call-sign (amateur radio identifier), which is well suited for our tracking use case. Different transmitters can use different channels (frequency division).

- We can profit from existing high-quality tools for reception (e.g.,WSJT-X [269]) without any design effort.

8.2.4 Implementation Details and Reception Hardware

As an example, we implement and evaluate tracking based on *FT4* on a *Samsung Galaxy S5 Mini*. We transmit an example signal available online[3], but more signals can be easily generated with *WSJT-X*. Thanks

[3] See https://www.sigidwiki.com/wiki/FT4

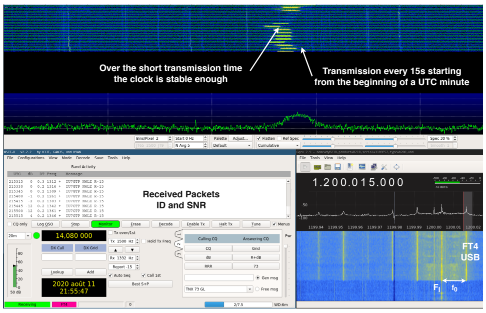

Over the short transmission time the clock is stable enough
Transmission every 15s starting from the beginning of a UTC minute
Received Packets ID and SNR
1.200.015.000
14,080 000
2020 août 11 21:55:47
FT4 USB
F_i f_0

Figure 8.3: A *Samsung Galaxy S5 Mini* successfully transmits an identifier using the *FT4* protocol. The clock is stable during the transmission of a packet. Transmissions are synchronized with UTC to simplify reception.

to *Noise-SDR*, all we have to do is to convert the baseband audio signal into a SSB complex signal with GNURadio, and then into an RF-PWM signal with Algorithm 1. We choose an intermediate frequency f_0 of 12 kHz. We add a call to *gmtime* to synchronize transmissions with the UTC time (every 15 s), facilitating reception because the receiver knows when to expect a signal. To avoid any coupling, the smartphone is placed on non-conductive material and it is controlled with *adb* over *WiFi*. For reception we use *Gqrx* to control an SDR and demodulate SSB. The resulting audio is then fed to *WSJT-X* for *FT4* demodulation and decoding. If tracking multiple phones at different DRAM frequencies, *Gqrx* could cycle over a list of common frequency values, similarly to *WiFi* and *BLE* scanning.

For reception we use simple off-the-shelf hardware: one monopole antenna one *TEKBOX TBWA2* amplifier powered via USB, and one low-end SDR (*RTL-SDR*) or high-end SDR (*USRP B210*). The *USRP B210* has a more stable clock, higher amplification, and it is useful to reach higher distances. Our setup is part of our laboratory equipment, and it was not chosen to optimize performance or cost for this use case. It works well with the leakage at 1.2 GHz from the *Samsung Galaxy S5 Mini*. Figure 8.3 shows a working example of reception.

8.2.5 Evaluation

We evaluate our *FT4*-based tracking solution with a simple example, yet representative of the use case. We count how many packets can be received at a certain distance, out of 160 packets transmitted by a smartphone laying on a desk in an office environment. We repeat the measurement at several distances, for each of the four sides of the phone laying on a desk in an office environment (40 packets for each side). The experiment takes 10 min for each iteration, which is a reasonable time frame for the tracking scenario. At the distance of 5 m, we can still detect and decode (without errors) 5 packets. For each

Table 8.4: FT4 reception.

d (m)	rx (160 tx)	avg(snr) (dB)	std(snr) (dB)	% correct
0.01	123	7.44	1.18	100%
1.00	71	-3.70	3.04	99%
4.00	37	-11.41	1.79	97%
5.00	5	-11.60	0.55	100%

distance (and the four sides combined), Table 8.4 shows the SNR at the receiver, the number of packets received, and the percentage of correctly decoded packets.

8.2.6 Extra: Fast Short-Distance Communication

The *FT4* packets are very small and slow. In some applications, the attacker might want to transmit a larger amount of data at higher speed. With our method, longer packets optimized for speed (instead of distance) could be easily interleaved with *FT4*. An attacker could capture them when close enough to the victim. For example, using *MFSK128* we were able to transmit 1 kb of text at 300 bps, up to a distance of 15 cm without errors.

8.3 AUTOMATED LEAKAGE DETECTION

Noise-SDR is also useful in non-malicious scenarios, for example, for the identification of electromagnetic leakages.

8.3.1 Motivation and Preliminary Observations

The problem of signal detection is an important topic in a wide range of contexts and applications. For side channel and data exfiltration attacks, the problem is made more complex by the fact that many of the signal parameters (e.g., the carrier frequency), are unknown. Significant results in carrier frequency identification have been obtained in previous work, by detecting known patterns generated by alternating different operations at known frequencies (e.g., [29, 30, 210, 301]). Identifying informative frequencies can also be seen as a feature selection problem when trying to classify software running on a device from its electromagnetic emissions, as it was done, for example, in [229]. However, these techniques require to implement custom software patterns and analysis methods. On the contrary, we propose to leverage the properties of existing radio protocols, made available by our generic modulation primitive. The main advantage of this method is that, without any extra effort, it builds on existing theoretical results and practical implementations of radio communications.

Table 8.5: Leakage harmonics controllable with *LoRa*.

Model	Frequencies(MHz)
Innos D6000	400, 800
Samsung Galaxy A30S	1794
Samsung Galaxy S5 Mini (SM G800F)	200, 800, 1000
Sony Xperia C5 (E5533)	800, 2400, 2800, 3200, 3600
Google Nexus 5 (D821)	200, 800

In our experience with multiple modulations, we have observed that linear chirps with a sweep bandwidth of a few tens of kHz and a slope of around 45° are clearly visible on spectrogram plots because of their unique shape. Indeed, chirp detection has been formulated also as a line detection problem with the Hough transform [262]. Their spread spectrum nature makes chirps naturally robust against narrow band interference. At the same time, chirps can use a relatively small sweep bandwidth, requiring simple hardware and low sensitivity.

8.3.2 Leveraging *LoRa*

Motivated by the previous observations, we had the intuition to use an existing protocol that uses chirps. We turned to *LoRa*, a proprietary protocol that leverages CSS and forward error correction to operate even below the noise floor [78, 226]. We use the highly configurable GNURadio implementation described in [267]. In our experiments we use a chirp bandwidth of 8 kHz, spreading factor 7, and 4/8 code rate. We transmit a fixed message *'Hello from DRAM (LoRa)'*. For reception, we implement a GNURadio flow-graph that scans a wide range of frequencies. The search is split in a coarse grained loop for hardware tuning (e.g., at multiples of 100 MHz), and another loop for fine grained software tuning (e.g., at multiples of 10 Hz around the center frequency). Whenever the *LoRa* receiver demodulates a packet with a valid Cyclic Redundancy Check (CRC), the flow-graph stores the center frequency and offset. With this method, we could confirm and expand the results of our previous manual analysis (Table 8.2), by automatically testing *LoRa* transmission at all the possible harmonics. Results are shown in Table 8.5.

8.3.3 Leveraging *GNSS*

A *GLONASS* receiver (e.g., *gnss-sdr*) searches in parallel for all the *GLONASS* satellites separated by 562.5 kHz. Additionally, it scans over a (configurable) window over the center frequency of each satellite, to take into account a possible carrier offset. Such receiver could be used to efficiently search for the *GLONASS C/A code* over a large bandwidth with a fine granularity, as we did in Figure 8.1. In addition, the RF-PWM modulation is similar to the Binary Offset Carrier (BOC) modulation used by Galileo: the multiplication with the IF square wave at f_0 produces two lobes at $F_l - f_0$ and $F_l + f_0$. An efficient receiver could coherently add the two lobes to increase the SNR. We leave the use of Galileo or another BOC (e.g., [183]) for future work.

8.4 INTERACTION WITH OTHER RADIOS ON THE SAME PLATFORM

We show how *Noise-SDR* can interact with other radios on the same platform.

8.4.1 Platform Noise

As every complex electronic system, a smartphone's platform often suffers from EMI and RFI problems, exacerbated by the presence of many radio receivers close to other noisy digital blocks. For example, the noise produced on the platform might desensitize the antennas of GPS and WiFi receivers, severely affecting their performance [153, 154, 179, 239]. An in-depth analysis of platform noise in wireless systems is available in [255]. Another example of noise coupling is the effect of CPU load on the magnetometer measurements, which can be used for website and application fingerprinting [171, 172].

In the following we will see how *Noise-SDR* can be used to generate a signal to inject in other radios on the same platform.

8.4.2 Impact On The NFC Reader

With *Screaming Channels* we have shown that radio transmitters can amplify and rebroadcast local leakages. Previous work has discussed cascaded effects for covert channels [55, 69]. We observe that similar effects, at the platform level, can also happen with some of the many radio transmitters available on a smartphone.

In particular, we observed on a *Nokia 3.1* that the carrier generated by the *NFC* reader is modulated by the DRAM accesses.[4] The advantages of using the *NFC* carrier are that:

- The carrier, emitted by an intended transmitter, is strong and stable. It has at a known frequency set by the standard, independent of the DRAM clock. One of its harmonics falls in the band

[4] Note that this is different from the side channel presented in [146]. In that case, the activity of the processor of a passive tag, which is placed in parallel to the antenna, unintentionally changes the load measured by the reader, leaking information about cryptographic operations. Testing this attack against a phone in tag emulation is an interesting direction for future work, but unrelated to *Noise-SDR*.

of the *FM* receiver of the same smartphone, which could be a victim.

- The *NFC* picks and amplifies the leakage from DRAM. On some phones (e.g., *Nokia 3.1*) the DRAM alone does not produce visible leakages, whereas it does when combined with *NFC*.

Regarding the threat model, the *NFC* carrier has to be active. For our experiments, we simply place the phones close to a ski card. A small *NFC* antenna could also be easily hidden under a table, or in a cover (commercial examples exist). Attacks that use the smartphone's *NFC* to attack credit cards in the same pocket have been shown in [258].

8.4.3 Simple Examples FM Radio Jamming and Spoofing

8.4.3.1 *FM Radio Victim*

The FM radio receiver present in many phones is a potential target for *Noise-SDR* attacks. Previous work [118] has used the FM receiver to pick leakages from the monitor of an air-gapped machine nearby, to exfiltrate data, even without using earphones as antennas. Instead, we show two *Noise-SDR* cases in which a signal can be injected in the FM receiver directly from software running on the same smartphone.

Screen to FM Injection. In Section 7.3 we have shown that the RF-PWM technique can be applied to the leakages produced by displaying a pattern on the screen with *JavaScript*. We observe that on a *Wiko Fever* these leakages fall in the 100 MHz band, and can be picked by the FM receiver, as shown in Figure 8.4. Although weak, a simple tune can be clearly distinguished from noise.

DRAM to FM Injection via NFC. We have previously observed that it is possible to arbitrarily modulate the *NFC* carrier by accessing DRAM. On a *Nokia 3.1*, the 7-th harmonic of *NFC* is 94.92 MHz, inside the 94.9 MHz channel of *FM* radio. It is powerful enough to be picked by the receiver, as shown in Figure 8.4.

When *NFC* is active, the 7-th harmonic of the carrier is strong enough to jam the legitimate radio signal. The signal quality is heavily reduced. By modulating the *NFC* carrier we can inject audible audio signals which are then played by the *FM* radio application on the phone. Unfortunately, the signal is weak and limited to simple tunes.

8.4.3.2 *Related Examples of Audio Injection*

Modern smartphones are controllable with voice commands. For this reason, offensive research has studied the stealthy injection of malicious audio signals by an external attacker. One option consists in transmitting audio with a powerful RF signal that is picked by the earphones and demodulated by the FM radio receiver [137]. Alternatively, the coupling between the USB cable and the audio input can be

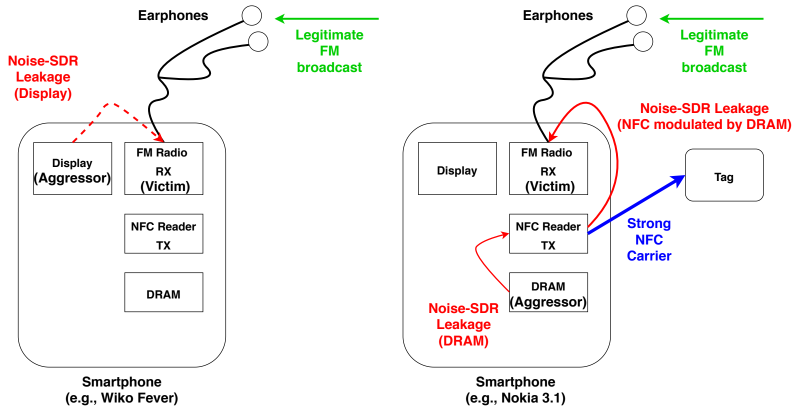


Figure 8.4: Two examples of injection in the FM receiver. The first method consists in displaying a pattern on the screen from *JavaScript*, and was tested on a *Wiko Fever* on channel 100 MHz. The second method, tested on a *Nokia 3.1*, is done in two steps. Unprivileged code modulates DRAM accesses using *ION*, producing a weak leakage that cannot be measured from outside the phone. However, this activity also modulates one of the harmonics of the carrier generated by the *NFC* reader. The resulting signal on the 7-th harmonic is strong enough to be audible on channel 94.9 MHz of the *FM* receiver.

leveraged to inject audio from the USB cable connected to a malicious charger, or even from a malicious power line [137]. Finally, a malicious audio command can be modulated on an ultrasound carrier, that is accidentally demodulated by the nonlinearity of the microphone on the smartphone [294]. Unlike these examples, our experiments inject audio in the FM radio from a source on the same device and not from an external one.

8.4.4 Preliminary Examples of GNSS Jamming and Spoofing

The *GNSS* receiver is an ideal victim of radio signals sent with *Noise-SDR*. Indeed, it is designed to receive extremely weak signals from space.

8.4.4.1 *Camera to GNSS Jamming*

We have already observed in Section 7.3 that the MIPI bus of the front camera of the *Google Pixel XL* produces leakages that can be modulated by the image that is being recorded. We also observe that some of the noise components fall in the *GPS* L1 band (1575.42 MHz), thus reducing the Carrier-to-Noise Ratio (CNR). By simply turning on and off the camera, without displaying any image, we can reliably trigger this condition, as shown in Figure 8.5 for the smartphone in the cockpit of a car. We have used the Sum-of-Squares (SOS) detector proposed

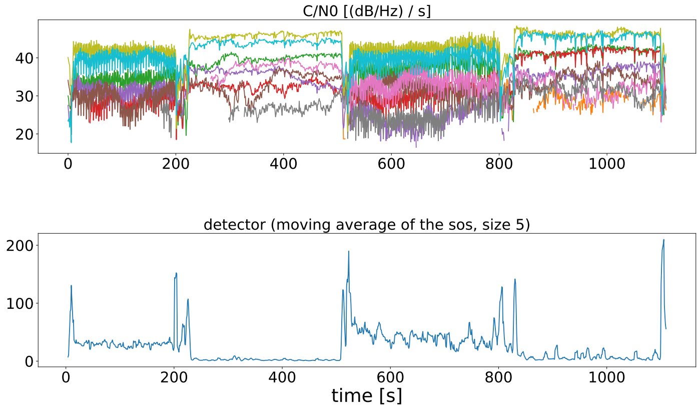


Figure 8.5: The sum-of-squares detector catches a degradation of the CNR at the *GPS* receiver when the camera is on.

in [23] to detect the jamming condition. An attacker with control over the camera could degrade the performance of *GPS* reception.

8.4.4.2 *DRAM to GLONASS Spoofing (Preliminary Results)*

On some ArmV8-A smartphones *Noise-SDR* is able to transmit a *GLONASS C/A* code that is recognized by the *gnss-sdr* receiver. Among the different *GNSS* systems (e.g., *GPS*, *GLONASS*, *Galileo*, *BeiDou*), we focus on *GLONASS* [44] because it uses a small bandwidth (0.511 MHz), and multiple frequencies close to 1600 MHz (which is a common value for DRAM leakages). We conjecture that, on some smartphones where the *GNSS* receiver is particularly sensitive to platform noise, *Noise-SDR* could inject a fake *GLONASS* satellite. For that to be possible, the following conditions should be true:

- The *GNSS* receiver should support *GLONASS*, which is common on modern multi-constellation receivers.
- The architecture should be ArmV8-A, to have enough bandwidth, unless better techniques are discovered to increase the bandwidth of *Noise-SDR* on ArmV7-A.
- The phone should have a strong DRAM leakage at 1600 MHz, which falls in the L1 *GLONASS* band. This is not an uncommon value.
- The attacker should choose a channel that falls close to 1600 MHz. For example, channel -2 at frequency 1600.875 MHz can be generated with $f_0 = 875$ kHz. Calibration could be used to improve the accuracy.
- There should be a strong coupling path between the aggressor DRAM and the victim *GNSS* receiver. Previous work on platform

> noise and our experiments with the camera of a *Google Pixel XL* suggest this could be true on current or future models.

The *Innos D6000* is the phone in our possession which is closest to have all the required characteristics. Unfortunately, although we managed to transmit a *GLONASS C/A code* for $k = -2$ at 800 MHz (see Figure 8.1), the harmonic at 1600 MHz was not strong enough to be measured outside the phone or to inject the code into the *GNSS* receiver.

8.5 COUNTERMEASURES

We have shown that *Noise-SDR* is an important threat in several scenarios. We now propose some countermeasures.

8.5.1 General Countermeasures

Unintended electromagnetic leakages and noise coupling paths between components can be reduced with careful RFI design (e.g., leakage modeling, internal shields), with a possibly higher design and manufacturing cost. With security in mind, intentional RFI activity should be considered. Covering electronic design techniques is outside the scope of this thesis.

The lower the time resolution, the stronger the constraints on the generated signal, to the point where RF-PWM is not practical. Countermeasures that mitigate timing side channel attack by preventing accurate timing measurements (e.g.,[39, 144, 237]) apply to *Noise-SDR* as well. However, the arms race with attackers (e.g.,[81, 100, 144, 155, 238]) shows that this solution is not definitive, because other sources of timing information can be exploited. In particular, *Noise-SDR* could easily replace timers with calibrated counters.

Phone covers which act as an electromagnetic shield can protect from leaking outside the phone.

Many techniques have been proposed to detect spoofing and jamming in radio protocols, for example, we followed [23] to detect *GNSS* jamming.

8.5.2 Countermeasures More Specific to DRAM

Those countermeasures against Rowhammer that focus on detecting or preventing fast DRAM accesses could also applied to *Noise-SDR*, but they are not a definitive solution. The detection of electromagnetic leakages [305] is not practical on mobile devices. Some of the approaches proposed for ArmV8-A [304] are not likely to be detected by observing cache misses, and cannot be easily forbidden in unprivileged code. Moreover, gadgets in system calls could be exploited from unprivileged code [304] (though not fast enough for flipping bits, they could be sufficient for *Noise-SDR* modulation). Finally, offensive research on Rowhammer is active and keeps finding several ways to access DRAM quickly (e.g,[81, 107, 156, 304]).

To avoid RFI issues, clocks in the system can continuously change their frequency following a chirp, spreading energy over a larger bandwidth. For example, [158] shows the lower impact that such clock has on *WiFi* performance. However, to the best of our knowledge this technology is not widespread in low-power DRAMs used in smartphones. Moreover, clock spreading used in *x86* desktops has not prevented attackers from detecting leakages due to intense memory activity [305], or from building effective covert channels [303]. Spreading could be actually useful to make custom protocols more robust, but it would complicate or prevent the synthesis of generic signals (a synchronized pre-despreading step would be necessary before transmission).

8.6 DISCUSSION

8.6.1 A Novel Problem

We have shown that an attacker can shape generic signals out of unintended electromagnetic noise, by simply running unprivileged Arm code on modern Android phones. We also have preliminary results from *JavaScript*. Because our approach turns smartphones into generic software-defined radios, attackers can leverage existing or custom radio protocols without any design effort.

We have shown that *Noise-SDR* signals can be injected into other transmitters and receivers on the same platform. We could modulate the NFC carrier, inject audio in the FM receiver, and we are close to doing the same with the satellite receiver.

We have presented several use cases, including the transmission of radio protocols (e.g., *AM*, *NBFM*, *PSK31*, *2xPSK500*, *MFSK128*, *RTTY*, *Olivia*, *SSTV*, *HamDRM*, *FT4*, *LoRa*, *GLONASS C/A Code*), a method for device tracking, and one for automated leakage detection. We have preliminary results on RFI aggression, jamming, and spoofing.

8.6.2 Limitations

Noise-SDR can transmit in a wide range of frequencies up to the order of GHz. Unfortunately, these frequencies are in limited number and fixed, because they are determined by the nature of the underlying hardware leakage (e.g., the harmonics of the DRAM clock). Nevertheless, the intermediate frequency f_0 adds a degree of freedom in the choice of the carrier frequency $f_c = F_l + f_0$.

The bandwidth of the signals that we can generate is limited by the time resolution at which we can control the pulses of the leakage. This requires having a good timer source and a short operation with a strong leakage. In our implementation we can reach tens of kHz on *ArmV7* and a few MHz on *ArmV8*. Appendix a.2 provides a formal description of the constraints imposed by the time resolution.

Another important parameter other then time resolution is the stability of the clock. A limited stability complicates the transmission of protocols that use a narrow bandwidth for a long transmission (e.g., *WSPR*).

8.6.3 Related Examples of Simple Radios

Many examples of radio transmitters built with very few and simple components have been developed by active communities of makers and hobbyists. They are often published online in form of blog posts and code repositories.

Simple tunes can be transmitted exploiting the leakages from memory accesses [68], or from the screen [273]. The LO leakage of an SDR receiver can be controlled to transmit data using the Radio Teletype (RTTY) protocol [215, 216]. A strong flexible transmission can be achieved by connecting a cable to the output interface of a microcontroller, for example, to a General Purpose Input Output (GPIO) pin, a PWM peripheral, or a fast serial bus. Successful examples of On Off Keying (OOK), Frequency Modulation (FM), Audio Frequency Shift Keying (AFSK), National Television System Committee (NTSC), Phase Alternating Line (PAL), and Amplitude Modulation (AM) were demonstrated with this method [57, 77, 169, 220, 240, 297, 309]. Transmitters based on clock and delay line Frequency Shift Keying (FSK) are shown in [257]. A Video Graphics Array (VGA) card uses multi-level high-speed DACs to generate a video signal that might leak from connectors and cables. These signals have been used to transmit Digital Video Broadcasting – Terrestrial (DVB-T), PAL, and NTSC [17], and FM music [135]. In [312], the output of the VGA DACs is fed to an RF modulator, whereas [165] uses the DAC of a USB-to-VGA converter as a Direct-to-RF Converter (DRFC), succeeding in the transmission of GSM and GPS signals. The USB-to-VGA converter has been reverse engineered, to achieve the continuous transmission of signal samples with a custom driver. In general, the best results are achieved when the output pins of peripherals or the DACs of VGA are available.

Although they often lack a formal theoretical explanation, these works were a useful source of inspiration, because they use smart signal processing tricks (e.g., undersampling) to overcome the limitations of simple hardware. However, with *Noise-SDR* we have taken a step further and achieve generic modulation on smartphones, using leakages produced as side effect of software execution.

8.7 CONCLUSION

In this chapter we have presented *Noise-SDR*, a general method to shape electromagnetic noise into arbitrary radio signals on modern smartphones. Going well beyond classic Soft-TEMPEST transmissions, *Noise-SDR* opens a whole new set of opportunities to the attackers, from leveraging existing radio protocols for device tracking and leakage identification, to injecting signals into other components of the same smartphone.

Part V

CONCLUDING REFLECTIONS

In this thesis we have conceived and demonstrated two novel attacks that exploit the interaction between digital and radio blocks in modern connected devices. We have open-sourced our tools and data, and *Screaming Channels* attacks have been replicated in industry and academia. This work, with its strength and limitations, has opened new directions and opportunities. We hope that future offensive and defensive research will continue investigating them.

9 DISCUSSION

We now discuss the advantages and disadvantages of our attacks and methodology, and many directions for future research.

9.1 STRENGTHS AND LIMITATIONS

9.1.1 Results

9.1.1.1 *Screaming Channels*

Screaming Channels are a novel side channel leakage vector with distinctive characteristics. On the one hand, they are easier to exploit than conventional side channels. Indeed, the leakage traces are amplified and transmitted at a power level than can be easily observed with standard and cheap radio equipment. They use the radio carrier emitted by the victim, and they do not require an external carrier to amplify the leakage. Most importantly, they can be mounted at a large distance from the victim, without requiring any hardware modification. On the other hand, attacking a device using *Screaming Channels* presents some challenges. First, side channel traces should be extracted from discrete radio packets that are themselves modulated with the intended data. Second, on the device that we have studied the leakage model is distorted, requiring a more complex profiling stage. We conjecture that the on-chip path between the digital block and the radio transmitter is likely to introduce distortions also on other targets. Finally, the leakages are sent over a noisy radio channel whose properties can vary over time, requiring a careful tuning of the reception equipment, using techniques borrowed from radio communications.

9.1.1.2 *Noise-SDR*

Going well beyond classic Soft-TEMPEST attacks and covert channels, *Noise-SDR* shows a method to shape arbitrary radio signals from a leakage, leveraging the RF-PWM technique. Unprivileged software running on a smartphone becomes a generic SDR able to transmit generic baseband signals generated with standard SDR tools. A large number of leakage carriers are available over the frequency spectrum. We could measure many of them from a few hundreds of MHz to some GHz. Unfortunately, given a smartphone they are in limited number and at fixed frequency, for example, at the multiples of the clock frequency of DRAM. The available power is also low and it depends on the specific device and frequency. Even though modulation is generic, the limited resolution and stability at which we can measure time from native code constrains the sampling rate and the clock stability of the transmission. For example, the available bandwidth ranges from a few tens of kHz on ArmV7-A to a few MHz on ArmV8-A. Besides showing many successful transmissions with diverse protocols, and applications such as device tracking and leakage detection, we have also demonstrated the use of *Noise-SDR* to impact receivers on the same platform. In particular, we believe that we are close to injecting a fake GLONASS signal into the satellite receiver.

9.1.1.3 *Replicability*

Our experiments with *Screaming Channels* can be easily replicated thanks to the code, data, and detailed instructions that we have open-sourced [311]. We will do the same for *Noise-SDR* after publication.

9.1.2 Methodology

9.1.2.1 *Manual Investigation*

Discovering and exploiting *Screaming Channels* on off-the-shelf devices required a significant amount of experimental work, and manual analysis and interpretation of the results. The main advantage of this approach is that we gained a deep understanding of the target and of its leakage. Over time, we developed a stable and flexible measurement setup and attack tools that simplify the study of the leakage. However, future work could study a methodology to automatically detect *screaming-channel* effects, leveraging previous work on automated carrier detection [29, 30, 210, 301] and the advanced modulation techniques of *Noise-SDR*.

9.1.2.2 *Large Scale Analysis*

We have experimented *Screaming Channels* on several chips and on many instances of the same device. Similarly, we evaluated *Noise-SDR* on many modern smartphones. Improving the analysis techniques in order to automate the analysis as much as possible would allow for a study at a larger scale. Our experiments with automated carrier detection with existing radio protocols already go in this direction.

9.2 CURRENT RELATED WORK

9.2.1 Other Attacks Against Mixed-Signal Chips

As we have seen in Chapter 3, remote physical side channel attacks are possible when attacker and victim share access to a remote hardware platform, and the attacker has access to some sort of sensor that measures the emissions of the victim. On a mixed-signal chip, an attacker with access to an ADC might be able to mount power side channel attacks against the CPU, exploiting the noise coupling path between the two [94]. This attack was published in 2019 after our initial publication of *Screaming Channels* in 2018 [38]. In another recent attack published in 2020 [49], the coupling between the ADC audio input and the switching power regulator on mixed-signal chips is exploited for audio eavesdropping at large distance. While we do not know if [49, 94] were inspired by *Screaming Channels*, we observe that all these works underline the importance of studying unexpected noise coupling paths in mixed-signal chips from a security point of view.

9.2.2 *Screaming Channels* Attacks With Deep Learning

In this thesis we have studied *Screaming Channels* with classic statistical methods, with the purpose of unveiling the properties of an unexplored side channel vector. In 2020, others [290, 291] have reproduced and extended our results using attacks based on deep learning. In these works, the target of the attacks is a *PCA10040* development board, whereas we mostly attack a *BLE Nano v2*. Both are based on the *nRF52832* mixed-signal BLE chip. In [291] several attacks at large distance are presented, including an attack at 15 m that requires fewer attack traces than the attack that we published in [35]. We believe that this attack profits from a better profiling phase with more traces and devices, and from the ability of the deep learning approach to deal with misaligned traces at a challenging distance. However, further research is required to compare these methods on traces collected in identical conditions and with the same victim and same reception chain.

9.3 FUTURE WORK

9.3.1 *Screaming Channels*

9.3.1.1 *More Types*

After having observed a few types of *screaming-channel* effects on some devices, we have focused on the *Nordic Semiconductor nRF52832* mixed-signal chip on some boards, including the off-the-shelf *BLE Nano v2* BLE dongle. In this case the *screaming-channel* leakage mostly comes from firmware execution, though we have also observed a weaker signal from DMA transfers to a cryptographic peripheral. The side-channel signal, modulated in amplitude, and the BLE packets, modulated in frequency, are orthogonal and easy to separate. In addition, the leakage is upconverted in two steps: first it is modulated onto the clock signal, and then onto the BLE carrier. We do not have access to the internal design on the chip, so we can only formulate an hypothesis on which is the coupling path that is responsible for this effect. After discussing with hardware designers, we concluded that a reasonable hypothesis is coupling through the power amplifier. Many other similar or different types of *Screaming Channels* could exist on this and other chips and boards. For example, coupling with an oscillator could lead to frequency or phase modulation. Studying more of these effects on more devices is an interesting direction for future work. Finally, *Screaming Channels* could also be used to attack peripherals on the system other than the hardware encryption block or the processor.

9.3.1.2 *More Protocols*

The *Screaming Channels* attacks presented in this thesis work on a BLE device. The orthogonality of side-channel leakage, modulated in amplitude, and intended packets, modulated in frequency, makes it easy to extract the desired signal. Frequency modulation is also used by other popular protocols, such as Bluetooth, ANT/ANT+ [2], Gazel™ [242], and Enhanced ShockBurst™ [241], all supported by the *Nordic nRF52832* chip that we have studied. On WiFi devices we were able to observe *screaming-channel* effects only in test mode, with the transmitter emitting a continuous wave. Developing complete attacks against WiFi devices is more challenging because WiFi packets use more complex modulation schemes. However, legacy beacons using DSSS phase modulation could be a simpler target, as phase modulation would be orthogonal to an amplitude-modulated leakage. Appendix a.1 discusses the WiFi case in more detail.

9.3.1.3 *Cryptography in Hardware*

In this thesis we have focused on application layer cryptography in software. However, both application layer and link layer encryption also use dedicated hardware blocks. Even though we could not observe a leakage from the internal computations of the hardware AES block of the *Nordic Semiconductor nRF52832*, we could easily detect memory transfers of key and plaintext in firmware and in hardware. This is a promising direction for future research. In particular, attacking the plaintext transfers is particularly interesting, because it is independent from possible side channel countermeasures in the cryptographic algorithm.

9.3.1.4 *More Attacks Against Real Systems*

With our proof-of-concept attack against the authentication method of Google Eddystone beacons we have demonstrated that *Screaming Channels* are a threat to real systems. Countless opportunities for further research exist in this direction. For example, encrypted telemetry and ephemeral identifiers used by Google Eddystone are also based on AES. Interestingly, on the *Nordic Semiconductor nRF52832* the required AES decryption can be done only in software, which leaks more than the hardware AES encryption block. Another promising target is public key cryptography in software. For example, BLE Low Energy Secure Connections (LESC) use Elliptic-Curve Diffie-Hellman (ECDH) to secure the pairing of two devices. Finally, we could imagine *screaming-channel* attacks against other proprietary IoT protocols.

9.3.1.5 *Improved Preprocessing and Attack Techniques*

In this thesis we have analyzed and exploited *Screaming Channels* using classic statistical methods. With a proper and careful choice of the methodology and techniques, we were able to both learn more about the distinctive characteristics of the leakage and optimize the attacks. Now that we have a better understanding of the leakage, it would be interesting to apply machine learning and deep learning techniques to compare their performance on specific aspects, such as preprocessing and profile reuse. In particular, interesting approaches that work well with cross-device attacks have been recently proposed [40, 62, 95]. Deep learning was applied to *Screaming Channels* in [291] with good results. The interest for methods that deal with highly dimensional data would increase when coupled with improvements on the sampling rate, which is now very low, and possible measurements of multiple features of the radio transmission at the same time. Our preprocessing techniques already take into account the fact that traces travel through a radio channel. For example, traces are normalized so that they are comparable independently from distance, setup, and slow time drifts in the amplification. In addition, we have successfully experimented with spatial diversity. However, there is still room for further improvement using techniques from radio communications. We could add carrier offset tracking, and more complex forms of channel estimation leveraging the key schedule as a training pattern, to cite a few examples.

9.3.1.6 *More (and Beyond) Mixed-Signal Chips*

Other researchers have shown additional security problems that can arise from the coupling of digital and analog blocks in mixed-signal chips [94]. We initially thought about the existence of *Screaming Channels* on mixed-signal chips, but similar effects could exist on other systems, too. Indeed, we have observed the coupling between the power regulator and the WiFi/BLE chip on an *ExpressIF ESP32*, and the modulation of the NFC carrier by DRAM accesses on a *Nokia 3.1* smartphone. Besides BLE, IoT protocols, and WiFi on small consumer devices, future work could also explore *screaming-channel* effects on planes, ships, satellites, and other systems where a powerful transmitter is present.

9.3.1.7 *Clever Specific Countermeasures*

Offensive research about *Screaming Channels* is meaningful as long as it helps improving our understanding of the problem and developing possible solutions. The nature of the devices affected by *Screaming Channels*, often very constrained in terms of cost, time to market, and power consumption, creates the need for clever specific countermeasures that go beyond classic side channel protections. While we have shown relatively simple mitigations for existing devices, we believe that special attention should be given to the conception of design and testing methods that would prevent serious *Screaming Channels* problems to appear in future devices. Many tools and techniques exist to study radio interference from the functional and performance point of view, but they could be enhanced to take into consideration possible security impacts. Besides mixed-signal chip manufacturers, the producers of simulation tools could be interested in developing analysis techniques that can capture information leakages from a digital block to the radio transmission during the multi-physical simulation of full chip designs.

9.3.2 *Noise-SDR*

9.3.2.1 *Improved Modulation*

With *Noise-SDR* we have demonstrated the idea of shaping arbitrary radio signals out of noise leveraging the RF-PWM technique. Future work could explore the use of other fully-digital radio techniques based on 1-bit coding, for example, *passband*-ΔΣ. Multiple threads or other forms of parallelism could further increase the control of the attacker on the leakage. The accuracy and stability of the clock could be improved though (real-time) calibration of the time sources.

9.3.2.2 *Spoofing GNSS*

We have already shown that *Noise-SDR* can be used as a RFI aggressor to interact with other radios on the same smartphone, including the NFC reader and the FM radio receiver. In particular, we have established the theoretical and practical foundation to inject a fake GNSS signal in the satellite receiver. We hope that future offensive research will build on this ground to make this attack actually possible.

9.3.2.3 *Beyond Arm Smartphones*

We have implemented the idea behind *Noise-SDR* on Arm smartphones, mostly using the leakages produced by DRAM accesses with native code. Smartphones are an interesting target because they carry a large number of receivers, but GNSS, WiFi, and BLE receivers can be found also in laptops, or in smart devices in the proximity of desktop computers. On these machines, there exist further opportunities to generate leakages with x86 processors, external DRAM, screens, and other buses and interconnections, as demonstrated by previous work on covert channels. Interesting sources of leakage could also be investigated on smart watches and other IoT devices. In addition, future work could develop cheap multi-constellation GNSS spoofers using fully-digital radio techniques on simple microcontrollers. The same could be done to develop cheap systems for signal injection and sensor spoofing.

9.3.3 Other Dangerous Interactions

9.3.3.1 *Parasitic Backscattering of Ambient Signals*

Previous work discussed in Chapter 1 has shown the interest of active side channel attacks that illuminate a sensitive device with a carrier. Two examples are particularly interesting. First, the power consumption of RFID or NFC tags modulates the observable impedance of the antenna leading to parasitic backscattering [204–206, 245] and parasitic load modulation [146], respectively. Second, the state of a WiFi card (e.g., on/off), which can be controlled by software, changes the input impedance of the antenna visible from outside, making it possible to build a backscattering covert channel [295, 296]. Future work could study further forms of coupling between the activity of a digital block and the impedance shown by the device or a radio front-end on the same device to an incident carrier. A possible information leakage would be even more interesting in case the incident carrier was an ambient transmission commonly present in modern households and offices, for example, BLE or WiFi. In addition, smartphone in card emulation mode are a promising target to explore parasitic load modulation at the NFC interface.

9.3.3.2 *Analog to Digital*

In this thesis we have mostly explored the effect of noise generated by digital blocks on radio frequency components. Indeed, the former are strong sources of noise, whereas the latter are very sensitive to it. However, also radio transmitters produce strong electromagnetic signals within the device and can impact other transceivers. Future work could investigate possible malicious uses of a radio transmitter to impact other analog and digital blocks on the same device.

9.4 CONCLUSION

In this thesis we have explored the security threats that can arise in modern connected devices because of the interaction between digital blocks and radio transceivers. As a result, we have discovered, analyzed, and exploited a novel side channel vector that propagates a leakage at large distance, called *Screaming Channels*. In addition, we have developed a method to shape electromagnetic noise in arbitrary radio signals that interact with other receivers, called *Noise-SDR*. Conceiving these attacks was the fruit of a creative research process, accompanied by extensive experimentation with off-the-shelf hardware, and open-source firmware. We hope that offensive and defensive research will further enhance our results and explore the many directions that are now open. We dedicated a particular effort to make our results easy to replicate. The code, data, and instructions that we have open sourced have already helped many researchers in industry and academia to replicate or even extend [291] our experiments.

Biot, September 2020

Giovanni Camurati

Part VI

APPENDIX

a APPENDIX

A.1 *screaming channels* ON WIFI CHIPS

In Chapter 4 we have shown the existence of *screaming-channel* effects on a BLE device. In this section, we explain why *screaming-channel* attacks on WiFi chips are challenging, but maybe possible.

A.1.1 Preliminaries

A.1.1.1 *The WiFi Protocol*

WiFi (*IEEE 802.11* [132]) is a popular technology used for wireless communications. There exist several variants of the protocol, operating at different frequencies, and with different modulations and rates. The main challenge for screaming channels is the use of OFDM subcarriers, modulated in both amplitude and phase. We focus on cards that support *IEEE 802.11g* (possibly with *IEEE 802.11b* legacy beacons). The main reason for our choice was their popularity and the availability of open-source drivers and firmware, useful for testing. In addition, an open-source *IEEE 802.11a/g/p* receiver [21, 307] based on SDR and *GNU Radio* is available to simplify research.

A.1.1.2 *An SDR Receiver*

Following [21, 307] closely, we briefly recall the structure of a packet at the physical layer and the algorithms used by the open-source receiver to process each part. A packet, sampled at 20 MHz, is made of frames. Each frame is composed of:

- **Preamble:** A cyclic sequence used to detect the beginning of a frame (*frame detection*). This sequence can be easily recognized thanks to its autocorrelation properties [157].
- **Short Training Sequence:** The cyclic property of the short training sequence can be used to estimate the frequency offset between transmitter and receiver [256], which can be then removed from the following samples (*frequency offset correction*).
- **Long Training Sequence:** The known long training sequence is used to detect the beginning of data symbols precisely (*symbol alignment*). The incoming signal is compared with the known training sequence using a matched filter. The position of the best match is then used to find the beginning of the following 64-sample data symbols (at fixed offsets from the training sequence).

- **Symbols:** Using the FFT algorithm, the samples of a symbol are transformed from the time domain to the frequency domain, where they are further processed. Each OFDM symbol is made of:
  - **Pilot Subcarriers (4):** The pilots are modulated with a known BPSK constellation. They can be used to estimate the phase error with linear regression (assuming that the error is linear with frequency). It is then possible to subtract the phase error from each of the data subcarriers (*phase offset correction*).
  - **Data Subcarriers (48):** Each of the subcarriers is modulated with BPSK, Quadrature Phase Shift Keying (QPSK), or QAM up to 64 QAM (supported types). For each of them, the effect of the channel compared to an ideal symbol (*channel estimation*) is computed using one one of the following algorithms: Least Square (LS), Least Mean Square (LMS), Spectral Temporal Averaging (STA), Linear Interpolation with the Comb Pilots (LC). The estimate of the channel (amplitude and phase) for each subcarrier is known as Channel State Information (CSI). Dividing the raw symbols by the CSI cancels the effect of the channel. Symbols are normalized and brought to the ideal quadrant, where they can be converted to bits by comparing them against fixed thresholds. The more advanced algorithms refine the estimation symbol by symbol using the error between the raw symbols and their ideal value.

  There are two types of symbol:
  - **Signal Symbol:** A BPSK symbol that communicates the length and encoding of the following symbols.
  - **Payload Symbols:** The OFDM symbols containing the actual data. After demodulation, the bits are deinterleaved, decoded and descrambled. Data is then encapsulated in a UDP packet. In this thesis we mostly focus on the physical layer, however, we use the MAC addresses to distinguish the packets sent by the target transmitter from other packets.

A.1.2 Common WiFi Physical Layer Impairments and Measurements

There exist several well-established methods to measure the physical layer of WiFi. Following [16], we present the most important ones, in particular, those that are related to analog/RF impairments in the transmitter.

- **Spectral Mask:** The emissions of the device can be compared (in the frequency domain) with the spectral mask, which defines the maximum allowed emission for each frequency. Commercial transmitters are supposed to meet the regulation requirements. In general, looking at the spectrum is helpful to identify *uncompensated frequency errors* and *spurious emissions*. For example, by simply looking at the emissions of the *nRF52832* we discovered the presence of intermodulation products between the system clock and the radio carrier. However, for OFDM systems, the following more advanced techniques are more suitable for investigating analog impairments.

- **Constellation Diagram:** For each subcarrier, the symbols in the frequency domain (amplitude, phase) can be plot on the complex plane. This is useful to visually pin-point phase and amplitude errors of the received symbols compared to the ideal ones. For example, *quadrature amplitude/phase imbalance* of the in-phase and quadrature components is clearly visible on the horizontal/vertical axis for the pilot tones. Since the other subcarriers are uncorrelated, quadrature imbalance and other impairments result in a noise-like spreading of the symbols in a cloud around their ideal value.

- **Error Vector Magnitude (EVM):** Given a possible ideal symbol R of the constellation, the corresponding measured symbols Z_i will fall in a noise-like cloud around it. Given M measurements, the EVM is a measure of the uncertainty of this cloud, defined as $\mathrm{EVM} = \sqrt{\frac{\sum |Z_i - R|^2}{\sum |R|^2}}$. The EVM on the entire constellation can be computed by averaging over all of its symbols. Plotting EVM over time is useful to pin-point problems in the time domain, such as the *settling time* of some components. Plotting the EVM for each subcarrier is useful to spot *mismatches* and *in-band spurious emissions*.

- **Cumulative Density Function (CDF) and Complementary Cumulative Density Function (CCDF):** The *nonlinear distortion* of a WiFi signal can be observed by plotting its CDF and CCDF against the ideal ones. The high Peak-to-Average Power Ratio (PAR) of a WiFi signal increases the chances of hitting the nonlinearity of some component, resulting in amplitude-to-amplitude (AM-AM) or amplitude-to-phase (AM-PM) distortions. For example, the power amplifier could exhibit lower gain for higher values of the input, resulting in a compressed signal.

Specific measurement at the physical layer were also explored to build covert channels (e.g.,[55, 69, 236]. In this case, they try to detect an error intentionally introduced by the transmitter to establish a covert communication channel.

A.1.3 Challenges

We identified five main reasons why exploiting *Screaming Channels* on WiFi transmissions might be complex compared to attacking a BLE device:

- **Signal and Circuit Quality:** BLE uses a simple frequency modulation scheme, whereas WiFi signals are modulated in amplitude and phase on several orthogonal subcarriers, achieving much higher data rates. Intuitively, a low-rate frequency modulation is much more tolerant to noise (especially on the amplitude) than a high-speed multi-level multi-carrier transmission. As a consequence, WiFi transmitters usually employ higher quality components and designs. For example, the power amplifier of WiFi transmitters has strong requirements in terms of linearity, because the transmitted signal has large amplitude variations (peak-to-average ratio) [16]. Given the higher quality of the radio front-end, WiFi chips are likely to be less impacted by problems such as screaming channels.

- **Orthogonality of Data and Leak:** The *screaming-channel* leakage on the *nRF52832* BLE chip shown in Chapter 4 consists in the unintentional amplitude modulation of the frequency-modulated BLE signal. The sensitive information is modulated on the harmonics of the clock which are themselves upconverted to the carrier frequency. Given the orthogonality of amplitude and frequency, the attacker can easily separate the unintentional leakage from the intentional data, without having to demodulate the data. On the contrary, it is unlikely for a leakage to be orthogonal to a WiFi packet modulated in amplitude and phase on many subcarriers. An attacker would have to first demodulate the intended data, and then compute the error between the received signal and the ideal one. At best, the attacker could focus on the pilot tones, which transmit a known phase-modulated signal for synchronization and channel estimation.

- **Reception Optimization:** When the leakage is orthogonal to the data, the attacker can optimize reception for the leakage. For the *nRF52832*, the radio can be tuned on the leakage instead of the center frequency of the packets. The biggest advantage of this method is that the receiver captures only the small leakage and not the strong intended signal with a small leakage on top of it. This allows measuring the small signal with good resolution, without saturating the ADC of the SDR. On the contrary, when the leakage is not orthogonal, the attacker has also to optimize the receiver for the reception of the intended packets. For WiFi, this means larger bandwidth, sampling rate, and clock accuracy than the ones used by the screaming channels receiver for the *nRF52832*. Moreover, the leakage will appear as a small variation on a big signal, making it harder to measure with good resolution.

- **Sampling Frequency:** Distinguishing a side channel trace corresponding to a certain encryption, or even a simple pattern in software, requires measuring a leakage signal for a sufficient duration and sampling frequency (see Chapter 4 for details about BLE). However, common physical layer measurements for WiFi happen at a very low frequency compared to side channels. For example, phase and magnitude errors of a symbol are computed over the entire duration of the symbol. In addition, packets are not sent continuously at a fixed frequency, instead, they arrive at random time instants. To recover a side channel trace at higher sampling frequency, the attacker would first have to demodulated a packet and generate an ideal signal, and then subtract it from the raw signal. Alternatively, the attacker could leverage undersampling and non-uniform sampling to reconstruct a useful side channel trace from error measurements arriving at low frequency at random points in time.
- **Variability and Complexity:** The physical layer of WiFi is more complex than the one of BLE. For example, a device might emit *IEEE 802.11b* beacons, but *IEEE 802.11g* packets. The transmission power, data rate, and hence the modulation type, might vary over time.

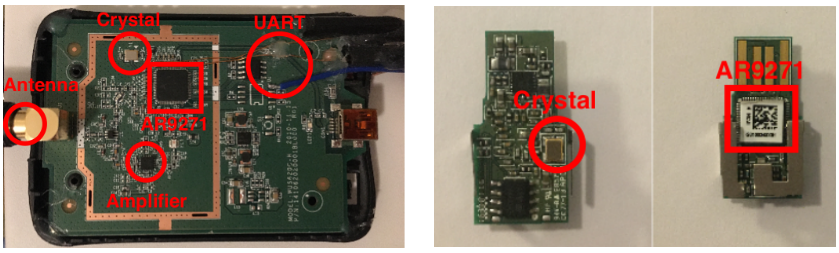
On the other hand, one advantage of WiFi is that devices usually transmit on a fixed channel, whereas BLE uses frequency hopping, which complicates reception.

A.1.4 Preliminary Experiments in Test Mode

Because of the previously mentioned challenges, we have conducted some experiments with the devices in test mode, sending a continuous wave.

A.1.4.1 *Qualcomm Atheros AR9271*

The *Qualcomm Atheros AR9271* is a mixed-signal chip featuring a WiFi transmitter (*IEEE 802.11 b/g/n*) and a 117 MHz CPU. It is used in commercial WiFi dongles, such as the *Alfa Network Long-Range USB Adapter AWUS036NHA* [9] and the *Penguin Wireless N USB Adapter (TPE-N150USB)* [203], shown in Figure a.1. In the *AWUS036NHA* case, an external amplifier (*SE2576L*), crystal (*T400PJ97*), and antenna are present on the board. On the device in the picture, we have connected



(a) *Alfa Network Long-Range USB Adapter AWUS036NHA*

(b) *Penguin Wireless N USB Adapter (TPE-N150USB)*

Figure a.1: Off-the-shelf WiFi dongles based on the *Qualcomm Atheros AR9271* mixed-signal chip. The *AWUS036NHA* in the picture was modified to add UART communication to the chip.

some additional cables in order to access the Universal Asynchronous Receiver Transmitter (UART) and communicate with the firmware.

We use a modified version of the *modwifi* driver for Linux [278] and the *modwifi-ath9k-htc* firmware [277] that were made available with [279]. In particular, we contacted the authors to obtain the patch to enable the constant jamming feature, which allows sending a constant stream of packets. We observed that with a particular sequence of commands we could trigger the transmission of a continuous wave as well. The firmware runs on the processor of the mixed signal chip and its execution might leak on the radio channel. We modify the main loop (which runs continuously and launches the tasks) in order to generate an easily identifiable pattern.

We observe that the radio carrier is modulated by software execution. We can clearly distinguish simple patterns, such as loops at different frequencies alternating every second. In this case, the intermodulation with the clock that was visible on the *nRF52832* is not present, and measuring the modulation directly on the carrier is difficult because of the small variations on a large signal. For this reason, we do not further investigate this device.

Previous work [45] has used the LO re-radiation [16] in WiFi cards to detect passive eavesdroppers. Using a loop probe close to the crystal of the *AR9271* and a *USRP B210* SDR, a leakage is clearly visible at frequency $f_{LO} = 4f_{rx}/3$, where f_{rx} is the frequency of the WiFi channel with the card in monitor mode. We observe that this leakage is modulated by simple patterns executing in software. In contrast to the transmission case, this signal is a continuous wave also when the card operates outside of the test mode, and it is always on when the device is active, waiting for packets to receive. Unfortunately, it is also a weaker signal than the intended transmission.

A.1.4.2 *Qualcomm Atheros AR9331*

The *Qualcomm Atheros AR9331* is a mixed-signal chip featuring a WiFi transceiver and a 400 MHz MIPS processor. It is used on the *Carambola 2* [1] WiFi-enabled Linux module by *8devices*.

The *AR9331* has a 400 MHz clock frequency and it is able to run *OpenWRT*, a Linux distribution that targets embedded devices. On this distribution we can load the same modified *modwifi* driver as for the *AR9271*, but in this case the operating system and the driver run on the chip itself and not on the host computer. We can also cross-compile programs to run on the device, in order to see if they produce a leak on the radio transmission.

We observe strong electromagnetic emissions for several harmonics of the 400 MHz clock, strongly modulated by software running on the processor. This chip is therefore a good target for conventional electromagnetic attacks. Similarly to the *AR9721*, a strong component is visible at $4f_{rx}/3$, where f_{rx} is the frequency of the channel, when the device is set in monitor mode or connected to a WiFi hotspot. However, to the best of our experimental setting, we were not able to observe any modulation of this component due to software execution. Setting the radio to transmit a continuous wave did not lead to better results, and we were not able to capture any modulation of the carrier due to software execution. Although these experiments do not fully exclude the presence of a screaming channel, they show that on this device conventional channels are easier to observe and exploit. For this reason we do not further investigate it.

A.1.4.3 *ExpressIF ESP32*

The *ExpressIF ESP32* [74] development kit features a mixed-signal chip with a WiFi and BLE transceiver and a dual-core Xtensa CPU, and a voltage regulator that also contains the USB-to-UART module.

To program the firmware running on the CPU we use the *ESP-IDF* [72]. Unfortunately, the continuous wave transmission mode is not supported. To activate it, we use the RF test firmware [73] provided in binary form by *ExpressIF*. Unfortunately, in this case we cannot run our own firmware on the device.

Without the ability to run software while transmitting in test mode, we were not able to observe any clear effect of software on the WiFi and BLE transmissions. However, with the test firmware we were able to observe another interesting *screaming-channel* effect. A leakage clearly visible at baseband close to the voltage regulator, is also visible at radio frequency, as it modulates the continuous wave sent by the transmitter in WiFi or BLE test mode. As we cannot easily run firmware tests while sending a continuous signal, we do not further investigate this chip.

A.1.4.4 *Real Transmissions*

We have conducted some preliminary experiments to detect the same effects when transmitting real WiFi packets. To this purpose, we have modified the open-source WiFi receiver [21, 307] to add several physical layer measurements, for example, EVM. Unfortunately, as of now we were not able to detect any *screaming-channel* effect on real packets.

A.1.5 Conclusion

We have identified at least one WiFi chip (*AR9271*) that shows some promising *screaming-channel* effect: a continuous carrier (test mode) and the LO re-radiation are modulated by simple software patterns running on the device. Compared to the effects observed on the *nRF52832*, these are much weaker.

We believe that further research on more devices and with more advanced measurement equipment is necessary to discover *Screaming Channels* on real WiFi transmissions. However, our preliminary analysis has already identified the main challenges and possible solutions.

A.2 *noise-sdr* DISCRETE TIME IMPLEMENTATION

A.2.1 Discrete Time

For clarity, we have explained our work with continuous time signals. In practice, the RF-PWM stage of *Noise-SDR* is implemented in software as a discrete time signal (sampled at discrete instants of time). Algorithm 1 is our discrete-time implementation of Figure 7.3. In the following, we discuss the constraints that arise from the sampling frequency.

A.2.2 The Importance of Time Resolution

The *time resolution* $T_{res} = 1/F_{res}$ at which we can generate the (high and low) pulses of the square wave impacts the *resolution* at which we control frequency, phase, and amplitude of the carrier. Given an integer q, the equations that describe the *quantization levels* (the possible discrete values that we can generate) are:

$$\begin{aligned} f_0 &= \frac{F_{res}}{q}, q \geqslant 2 \\ \theta_k &= 2k\pi f_0 \frac{q}{F_{res}}, q \in \left[- \left\lfloor \frac{F_{res}}{2kf_0} \right\rfloor, \left\lfloor \frac{F_{res}}{2kf_0} \right\rfloor \right) \\ a_k &= \sin(k\pi q \frac{f_0}{F_{res}}), q \in \left[0, \frac{1}{2k} \frac{F_{res}}{f_0} \right] \end{aligned} \tag{a.1}$$

Algorithm 1 Baseband to RF-PWM Modulation.

```
Input: Complex baseband signal $x_{bb}$ sampled at $T_{s_{bb}} = 1/F_{s_{bb}}$, fundamental frequency IF $= f_0$, time resolution $T_{res} = 1/F_{res}$, assuming $x_{bb}$ is normalized and $F_{res}$ is a multiple of $F_{s_{bb}}$
Output: Lists of RF-PWM pulse timings $T_{high}, T$ at resolution $T_{res}$
rep $\leftarrow T_s/T_{res}$
$T_{high} \leftarrow []$
$T \leftarrow []$
for $k = 0$ to $len(x_{bb}) - 1$ do
    $a_{bb}[k], \theta_{bb}[k] \leftarrow toPolar(x_{bb}[k])$
    $a_{bb}[k] \leftarrow \arcsin(a_{bb}[k])/\pi$
end for
for $i = 0$ to $len(x_{bb}) \cdot rep - 1$ do
    $x_{pwm}[i] \leftarrow \cos(2\pi f_0 i/F_{res} + \theta_{bb}[i/rep])$
end for
$i \leftarrow 0$
while $x_{pwm}(i) < 0$ do
    $i \leftarrow i + 1$
end while
while $i < len(x_{pwm})$ do
    $t \leftarrow 0$
    while $i + t < len(x_{pwm})$ and $x_{pwm}(i + t) \geqslant 0$ do
        $t \leftarrow t + 1$
    end while
    while $i + t < len(x_{pwm})$ and $x_{pwm}(i + t) < 0$ do
        $t \leftarrow t + 1$
    end while
    $T \leftarrow [T, t]$
    $T_{high} \leftarrow [T_{high}, a_{bb}[i/res] \cdot t]$
    $i \leftarrow i + t$
end while
```

A.2.3 Sampling

The baseband signal is sampled at $F_{s_{bb}}$ such that:

$$2B_{bb} < F_{s_{bb}} < f_0 - \Delta f_{max} \tag{a.2}$$

where the lower limit is set by the sampling theorem, and the upper limit is imposed by the fact that we can modulate the square wave one period at a time. In addition, the sampling theorem limits the square wave that we can generate to:

$$f_0 + B < F_{res}/2 \tag{a.3}$$

Indeed $F_{res}/2$ is the maximum frequency that we can generate having time resolution T_{res}.

BIBLIOGRAPHY

[1] 8Devices. *Carambola2*. https://fccid.io/Z9W-CM2/.

[2] *ANT Wireless Website*. https://www.thisisant.com/.

[3] David Adamy. *EW 101: A first course in electronic warfare*. Vol. 101. Artech house, 2001.

[4] S. Adibelli, C. Cheng, P. Juyal, and A. Zajic. “An Investigation of THz Backscattered Side-Channels Measurement at a Distance.” In: *2019 13th European Conference on Antennas and Propagation (EuCAP)*. Mar. 2019, pp. 1–5.

[5] Ali Afzali-Kusha, Makoto Nagata, Nishath K. Verghese, and David J. Allstot. “Substrate Noise Coupling in SoC Design: Modeling, Avoidance, and Validation.” In: *Proceedings of the IEEE* 94.12 (2006), pp. 2109–2138. DOI: 10.1109/JPROC.2006.886029. URL: https://doi.org/10.1109/JPROC.2006.886029.

[6] Dakshi Agrawal, Josyula R. Rao, and Pankaj Rohatgi. “Multi-channel Attacks.” In: *Cryptographic Hardware and Embedded Systems - CHES 2003, 5th International Workshop, Cologne, Germany, September 8-10, 2003, Proceedings*. Ed. by Colin D. Walter, Çetin Kaya Koç, and Christof Paar. Vol. 2779. Lecture Notes in Computer Science. Springer, 2003, pp. 2–16. DOI: 10.1007/978-3-540-45238-6_2. URL: https://doi.org/10.1007/978-3-540-45238-6_2.

[7] Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, and Pankaj Rohatgi. “The EM Side-Channel(s).” In: *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*. Ed. by Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar. Vol. 2523. Lecture Notes in Computer Science. Springer, 2002, pp. 29–45. DOI: 10.1007/3-540-36400-5_4. URL: https://doi.org/10.1007/3-540-36400-5_4.

[8] US Airforce. *Air Force Systems Security Memorandum 7011 - Emission Security Countermeasures Review*. Tech. rep. Document available at https://cryptome.org/afssm-7011.htm. 1998.

[9] Alfa Network. *Long-Range USB Adapter AWUS036NHA*. https://fccid.io/UQ23668.

[10] Ross J. Anderson. *Security engineering - a guide to building dependable distributed systems (2. ed.)* Wiley, 2008. ISBN: 978-0-470-06852-6.

[11] Ross Anderson and Markus G. Kuhn. *Soft Tempest - An Opportunity for NATO*. 1999.

[12] Daniele Antonioli, Nils Ole Tippenhauer, and Kasper Bonne Rasmussen. “The KNOB is Broken: Exploiting Low Entropy in the Encryption Key Negotiation Of Bluetooth BR/EDR.” In: *28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019*. Ed. by Nadia Heninger and Patrick Traynor. USENIX Association, 2019, pp. 1047–1061. URL: https://www.usenix.org/conference/usenixsecurity19/presentation/antonioli.

[13] Daniel Arp, Erwin Quiring, Christian Wressnegger, and Konrad Rieck. “Privacy Threats through Ultrasonic Side Channels on Mobile Devices.” In: *2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017*. IEEE, 2017, pp. 35–47. DOI: 10.1109/EuroSP.2017.33. URL: https://doi.org/10.1109/EuroSP.2017.33.

[14] Michael Backes, Tongbo Chen, Markus Dürmuth, Hendrik P. A. Lensch, and Martin Welk. “Tempest in a Teapot: Compromising Reflections Revisited.” In: *30th IEEE Symposium on Security and Privacy (S&P 2009), 17-20 May 2009, Oakland, California, USA*. IEEE Computer Society, 2009, pp. 315–327. DOI: 10.1109/SP.2009.20. URL: https://doi.org/10.1109/SP.2009.20.

[15] George Becker, J Cooper, Elke DeMulder, Gilbert Goodwill, Joshua Jaffe, G Kenworthy, T Kouzminov, A Leiserson, M Marson, Pankaj Rohatgi, et al. “Test vector leakage assessment (TVLA) methodology in practice.” In: *International Cryptographic Module Conference*. Vol. 1001. 2013, p. 13.

[16] Arya Behzad. *Wireless LAN Radios: System Definition to Transistor Design (IEEE Press Series on Microelectronic Systems)*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2008. ISBN: 978-0471-70964-0.

[17] Fabrice Bellard. *Analog and Digital TV (DVB-T) Signal Generation*. 2005. URL: https://bellard.org/dvbt/.

[18] Daniel J. Bernstein, Tanja Lange, and Christine van Vredendaal. “Tighter, faster, simpler side-channel security evaluations beyond computing power.” In: *IACR Cryptology ePrint Archive* 2015 (2015), p. 221. URL: http://eprint.iacr.org/2015/221.

[19] Liu Biao and Kong Jun-hui. “Practical template attacks based on pooled covariance matrix.” In: *2015 7th Asia-Pacific Conference on Environmental Electromagnetics (CEEM)*. Nov. 2015, pp. 184–188. DOI: 10.1109/CEEM.2015.7368661.

[20] Alex Biryukov, Daniel Dinu, and Yann Le Corre. “Side-Channel Attacks Meet Secure Network Protocols.” In: *Applied Cryptography and Network Security - 15th International Conference, ACNS 2017, Kanazawa, Japan, July 10-12, 2017, Proceedings*. Ed. by Dieter Gollmann, Atsuko Miyaji, and Hiroaki Kikuchi. Vol. 10355. Lecture Notes in Computer Science. Springer, 2017, pp. 435–

454. DOI: 10.1007/978-3-319-61204-1_22. URL: https://doi.org/10.1007/978-3-319-61204-1_22.

[21] Bastian Bloessl, Michele Segata, Christoph Sommer, and Falko Dressler. "An IEEE 802.11a/g/p OFDM receiver for GNU radio." In: *Proceedings of the second workshop on Software radio implementation forum, SRIF@SIGCOMM 2013, Hong Kong, China, August 12-16, 2013*. Ed. by Soung Chang Liew. ACM, 2013, pp. 9–16. DOI: 10.1145/2491246.2491248. URL: https://doi.org/10.1145/2491246.2491248.

[22] Andrey Bogdanov, Ilya Kizhvatov, Kamran Manzoor, Elmar Tischhauser, and Marc Witteman. "Fast and Memory-Efficient Key Recovery in Side-Channel Attacks." In: *Selected Areas in Cryptography - SAC 2015 - 22nd International Conference, Sackville, NB, Canada, August 12-14, 2015, Revised Selected Papers*. Ed. by Orr Dunkelman and Liam Keliher. Vol. 9566. Lecture Notes in Computer Science. Springer, 2015, pp. 310–327. DOI: 10.1007/978-3-319-31301-6_19. URL: https://doi.org/10.1007/978-3-319-31301-6_19.

[23] Daniele Borio and Ciro Gioia. "Real-time jamming detection using the sum-of-squares paradigm." In: *International Conference on Location and GNSS, ICL-GNSS 2015, Gothenburg, Sweden, June 22-24, 2015*. IEEE, 2015, pp. 1–6. DOI: 10.1109/ICL-GNSS.2015.7217161. URL: https://doi.org/10.1109/ICL-GNSS.2015.7217161.

[24] Kai Borre, Dennis M Akos, Nicolaj Bertelsen, Peter Rinder, and Søren Holdt Jensen. *A software-defined GPS and Galileo receiver: a single-frequency approach*. Springer Science & Business Media, 2007.

[25] Sergey Bratus, Travis Goodspeed, Ange Albertini, and Debanjum S. Solanky. "Fillory of PHY: Toward a Periodic Table of Signal Corruption Exploits and Polyglots in Digital Radio." In: *10th USENIX Workshop on Offensive Technologies, WOOT 16, Austin, TX, USA, August 8-9, 2016*. Ed. by Natalie Silvanovich and Patrick Traynor. USENIX Association, 2016. URL: https://www.usenix.org/conference/woot16/workshop-program/presentation/bratus.

[26] Donald G. Brennan. "Linear diversity combining techniques." In: *Proceedings of the IRE* 47.6 (1959), pp. 1075–1102. DOI: 10.1109/JRPROC.1959.287136.

[27] Stephane Bronckers, Geert Van der Plas, Gerd Vandersteen, and Yves Rolain. *Substrate Noise Coupling in Analog/RF Circuits*. Norwood, MA, USA: ARTECH HOUSE, 2009. ISBN: 978-1-59693-271-5.

[28] Andrew Burnside, Ahmet T. Erdogan, and Tughrul Arslan. "The Re-emission Side Channel." In: *2008 ECSIS Symposium on Bio-inspired, Learning, and Intelligent Systems for Security, BLISS 2008, Edinburgh, UK, 4-6 August 2008*. Ed. by Adrian Stoica, Tughrul Arslan, Daniel Howard, Tetsuya Higuchi, and Ahmed O. El-Rayis. IEEE Computer Society, 2008, pp. 154–159. DOI: 10.1109/BLISS.2008.22. URL: https://doi.org/10.1109/BLISS.2008.22.

[29] R. Callan, A. Zajic, and M. Prvulovic. "A Practical Methodology for Measuring the Side-Channel Signal Available to the Attacker for Instruction-Level Events." In: *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*. 2014, pp. 242–254.

[30] Robert Locke Callan, Alenka G. Zajic, and Milos Prvulovic. "FASE: finding amplitude-modulated side-channel emanations." In: *Proceedings of the 42nd Annual International Symposium on Computer Architecture, Portland, OR, USA, June 13-17, 2015*. Ed. by Deborah T. Marr and David H. Albonesi. ACM, 2015, pp. 592–603. DOI: 10.1145/2749469.2750394. URL: https://doi.org/10.1145/2749469.2750394.

[31] Giovanni Camurati. *Rejeu: Screaming Channels: quand un émetteur radio diffuse des canaux auxiliaires*. Invited talk (in French) at Rendez-Vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information (RESSI) 2019.

[32] Giovanni Camurati. *Screaming Channels: When Electromagnetic Side Channels Meet Radio Transceivers*. Invited talk and poster at Workshop on Practical Hardware Innovations in Security Implementation and Characterization (PHISIC) 2019.

[33] Giovanni Camurati. *Screaming Channels: quand un émetteur radio diffuse des canaux auxiliaires*. Invited talk (in French) at GDR Ondes, Journée thématique «Sécurité des systèmes électroniques et communicants» 2019.

[34] Giovanni Camurati. *Understanding Screaming Channels: From a Detailed Analysis to Improved Attacks*. Invited talk at CONFidence 2020.

[35] Giovanni Camurati, Aurélien Francillon, and François-Xavier Standaert. "Understanding Screaming Channels: From a Detailed Analysis to Improved Attacks." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2020.3 (2020), pp. 358–401. DOI: 10.13154/tches.v2020.i3.358-401. URL: https://doi.org/10.13154/tches.v2020.i3.358-401.

[36] Giovanni Camurati and Aurélien Francillon. "Noise-SDR: Shaping Arbitrary Radio Signals out of Noise on Modern Smartphones." 2020. Under submission.

[37] Giovanni Camurati and Marius Muench. *Screaming Channels: When Electromagnetic Side Channels Meet Radio Transceivers*. Presentation at Black Hat USA 2018.

[38] Giovanni Camurati, Sebastian Poeplau, Marius Muench, Tom Hayes, and Aurélien Francillon. "Screaming Channels: When Electromagnetic Side Channels Meet Radio Transceivers." In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*. Ed. by David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang. ACM, 2018, pp. 163–177. DOI: 10.1145/3243734.3243802. URL: https://doi.org/10.1145/3243734.3243802.

[39] Yinzhi Cao, Zhanhao Chen, Song Li, and Shujiang Wu. "Deterministic Browser." In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. Ed. by Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu. ACM, 2017, pp. 163–178. DOI: 10.1145/3133956.3133996. URL: https://doi.org/10.1145/3133956.3133996.

[40] Mathieu Carbone, Vincent Conin, Marie-Angela Cornelie, François Dassance, Guillaume Dufresne, Cécile Dumas, Emmanuel Prouff, and Alexandre Venelli. "Deep Learning to Evaluate Secure RSA Implementations." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2019.2 (2019), pp. 132–161. DOI: 10.13154/tches.v2019.i2.132-161. URL: https://doi.org/10.13154/tches.v2019.i2.132-161.

[41] Brent Carrara. "Air-Gap Covert Channels." In: 2016.

[42] Brent Carrara and Carlisle Adams. "On Acoustic Covert Channels Between Air-Gapped Systems." In: *Foundations and Practice of Security - 7th International Symposium, FPS 2014, Montreal, QC, Canada, November 3-5, 2014. Revised Selected Papers*. Ed. by Frédéric Cuppens, Joaquín García-Alfaro, A. Nur Zincir-Heywood, and Philip W. L. Fong. Vol. 8930. Lecture Notes in Computer Science. Springer, 2014, pp. 3–16. DOI: 10.1007/978-3-319-17040-4_1. URL: https://doi.org/10.1007/978-3-319-17040-4_1.

[43] Brent Carrara and Carlisle Adams. "Out-of-Band Covert Channels—A Survey." In: *ACM Comput. Surv.* 49.2 (June 2016). ISSN: 0360-0300. DOI: 10.1145/2938370. URL: https://doi.org/10.1145/2938370.

[44] GLONASS Coordination Scientific Information Center. *GLONASS Interface Control Document*. 1998. URL: https://www.unavco.org/help/glossary/docs/ICD_GLONASS_4.0_(1998)_en.pdf.

[45] Anadi Chaman, Jiaming Wang, Jiachen Sun, Haitham Hassanieh, and Romit Roy Choudhury. “Ghostbuster: Detecting the Presence of Hidden Eavesdroppers.” In: *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking, MobiCom 2018, New Delhi, India, October 29 - November 02, 2018*. Ed. by Rajeev Shorey, Rohan Murty, Yingying Jennifer Chen, and Kyle Jamieson. ACM, 2018, pp. 337–351. DOI: 10.1145/3241539.3241580. URL: https://doi.org/10.1145/3241539.3241580.

[46] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. “Template Attacks.” In: *Cryptographic Hardware and Embedded Systems - CHES 2002, 4th International Workshop, Redwood Shores, CA, USA, August 13-15, 2002, Revised Papers*. Ed. by Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar. Vol. 2523. Lecture Notes in Computer Science. Springer, 2002, pp. 13–28. DOI: 10.1007/3-540-36400-5_3. URL: https://doi.org/10.1007/3-540-36400-5_3.

[47] Chia-Lin Cheng, Luong N. Nguyen, Milos Prvulovic, and Alenka G. Zajić. “Exploiting Switching of Transistors in Digital Electronics for RFID Tag Design.” In: *IEEE Journal of Radio Frequency Identification* 3 (2018), pp. 67–76.

[48] Richard Chirgwin. *Boffins: Mixed-signal silicon can SCREAM your secrets to all*. News article. July 2018. URL: https://www.theregister.com/2018/07/27/screaming_channels_attack/.

[49] Jieun Choi, Hae-Yong Yang, and Dong-Ho Cho. “TEMPEST Comeback: A Realistic Audio Eavesdropping Threat on Mixed-signal SoCs.” In: *CCS ’20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*. Ed. by Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna. ACM, 2020, pp. 1085–1101. DOI: 10.1145/3372297.3417241. URL: https://doi.org/10.1145/3372297.3417241.

[50] Omar Choudary and Markus G. Kuhn. “Efficient Template Attacks.” In: *Smart Card Research and Advanced Applications - 12th International Conference, CARDIS 2013, Berlin, Germany, November 27-29, 2013. Revised Selected Papers*. Ed. by Aurélien Francillon and Pankaj Rohatgi. Vol. 8419. Lecture Notes in Computer Science. Springer, 2013, pp. 253–270. DOI: 10.1007/978-3-319-08302-5_17. URL: https://doi.org/10.1007/978-3-319-08302-5_17.

[51] Omar Choudary and Markus G. Kuhn. “Template Attacks on Different Devices.” In: *Constructive Side-Channel Analysis and Secure Design - 5th International Workshop, COSADE 2014, Paris, France, April 13-15, 2014. Revised Selected Papers*. Ed. by Emmanuel Prouff. Vol. 8622. Lecture Notes in Computer Science.

Springer, 2014, pp. 179–198. DOI: 10.1007/978-3-319-10175-0_13. URL: https://doi.org/10.1007/978-3-319-10175-0_13.

[52] Jiska Classen and Francesco Gringoli. *Spectra: Breaking Separation Between Wireless Chips*. Presentation at Black Hat USA 2020. Aug. 2020.

[53] Federal Communications Commission. *A Short History of Radio With an Inside Focus on Mobile Radio*. https://transition.fcc.gov/omd/history/radio/documents/short_history.pdf. 2004.

[54] *Construction guide for EM probe*. https://github.com/emsec/SCATools/wiki/BuildEMProbe.

[55] Emmanuel Cottais, José Lopes Esteves, and Chaouki Kasmi. "Second Order Soft-TEMPEST in RF Front-Ends: Design and Detection of Polyglot Modulations." In: *2018 International Symposium on Electromagnetic Compatibility (EMC EUROPE)* (2018), pp. 166–171.

[56] Alexandru Csete. *Gqrx SDR*. https://gqrx.dk/.

[57] Ang Cui. *Funtenna*. 2015. URL: https://github.com/funtenna.

[58] Mathieu Cunche and Leonardo S. Cardoso. "Analyzing Ultrasound-based Physical Tracking Systems." In: 2018.

[59] James T. Curran, Aiden Morrison, and Cillian O'Driscoll. *(In)Feasibility of Multi-Frequency Spoofing*. 2018. URL: https://insidegnss.com/infeasibility-of-multi-frequency-spoofing/.

[60] Boris Danev, Davide Zanetti, and Srdjan Capkun. "On physical-layer identification of wireless devices." In: *ACM Comput. Surv.* 45.1 (2012), 6:1–6:29. DOI: 10.1145/2379776.2379782. URL: https://doi.org/10.1145/2379776.2379782.

[61] Daniel Veilleux. *Simple Application-level Authentication*. https://devzone.nordicsemi.com/nordic/nordic-blog/b/blog/posts/simple-application-level-authentication. 2017.

[62] Debayan Das, Anupam Golder, Josef Danial, Santosh Ghosh, Arijit Raychowdhury, and Shreyas Sen. "X-DeepSCA: Cross-Device Deep Learning Side Channel Attack." In: *Proceedings of the 56th Annual Design Automation Conference 2019, DAC 2019, Las Vegas, NV, USA, June 02-06, 2019*. ACM, 2019, p. 134. DOI: 10.1145/3316781.3317934. URL: https://doi.org/10.1145/3316781.3317934.

[63] A. Dubey, D. Vohra, K. Vachhani, and A. Rao. "Demonstration of vulnerabilities in GSM security with USRP B200 and open-source penetration tools." In: *2016 22nd Asia-Pacific Conference on Communications (APCC)*. 2016, pp. 496–501.

[64] Emmanuel Duponchelle and Pierre-Michel Ricordel. *Risques associés aux signaux parasites compromettants : le cas des câbles DVI et HDMI*. Presentation (in French) at Symposium sur la sécurité des technologies de l'information et des communications (SSTIC) 2018. URL: https://www.sstic.org/2018/presentation/risques_spc_dvi_et_hdmi/.

[65] François Durvaux and François-Xavier Standaert. "From Improved Leakage Detection to the Detection of Points of Interests in Leakage Traces." In: *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*. Ed. by Marc Fischlin and Jean-Sébastien Coron. Vol. 9665. Lecture Notes in Computer Science. Springer, 2016, pp. 240–262. DOI: 10.1007/978-3-662-49890-3_10. URL: https://doi.org/10.1007/978-3-662-49890-3_10.

[66] Wim van Eck. "Electromagnetic radiation from video display units: An eavesdropping risk?" In: *Comput. Secur.* 4.4 (1985), pp. 269–286. DOI: 10.1016/0167-4048(85)90046-X. URL: https://doi.org/10.1016/0167-4048(85)90046-X.

[67] M. Abdelaziz Elaabid and Sylvain Guilley. "Portability of templates." In: *J. Cryptographic Engineering* 2.1 (2012), pp. 63–74. DOI: 10.1007/s13389-012-0030-6. URL: https://doi.org/10.1007/s13389-012-0030-6.

[68] William Entriken. *System Bus Radio*. 2013. URL: https://github.com/fulldecent/system-bus-radio.

[69] José Lopes Esteves, Emmanuel Cottais, and Chaouki Kasmi. "Second Order Soft Tempest: From Internal Cascaded Electromagnetic Interactions to Long Haul Covert Channels." In: *2019 URSI Asia-Pacific Radio Science Conference (AP-RASC)* (2019), pp. 1–3.

[70] Ettus Research. *USRP N210*. http://www.ettus.com/all-products/un210-kit/.

[71] Ettus Research. *USRP V210*. http://www.ettus.com/all-products/UB210-KIT/.

[72] ExpressIF. *ESP-IDF*. https://github.com/espressif/esp-idf.git.

[73] ExpressIF. *ESP RF Testing*. https://www.espressif.com/en/support/download/other-tools.

[74] ExpressIF. *ExpressIF ESP32*. https://fccid.io/2AC7Z-ESPWROOM32/.

[75] FLDigi. *FLDigi Modes*. URL: http://www.w1hkj.com/FldigiHelp-3.21/html/mode_table_page.html.

[76] *FLDigi*. http://www.w1hkj.com/.

[77] *FM Transmitter RPi3*. URL: https://github.com/PNPtutorials/FM_Transmitter_RPi3.

[78] Guillaume Ferré and Audrey Giremus. "LoRa Physical Layer Principle and Performance Analysis." In: *25th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2018, Bordeaux, France, December 9-12, 2018*. IEEE, 2018, pp. 65–68. DOI: 10.1109/ICECS.2018.8617880. URL: https://doi.org/10.1109/ICECS.2018.8617880.

[79] I. D. Flintoft, A. C. Marvin, M. P. Robinson, K. Fischer, and A. J. Rowell. "The re-emission spectrum of digital hardware subjected to EMI." In: *IEEE Transactions on Electromagnetic Compatibility* 45.4 (Nov. 2003), pp. 576–585. ISSN: 1558-187X. DOI: 10.1109/TEMC.2003.819058.

[80] Aurélien Francillon, Boris Danev, and Srdjan Capkun. "Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars." In: *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011*. The Internet Society, 2011. URL: https://www.ndss-symposium.org/ndss2011/relay-attacks-on-passive-keyless-entry-and-start-systems-in-modern-cars.

[81] Pietro Frigo, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. "Grand Pwning Unit: Accelerating Microarchitectural Attacks with the GPU." In: *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*. IEEE Computer Society, 2018, pp. 195–210. DOI: 10.1109/SP.2018.00022. URL: https://doi.org/10.1109/SP.2018.00022.

[82] *GNU Radio*. https://www.gnuradio.org/.

[83] G. Gandhi, V. Aggarwal, and L. O. Chua. "The First Radios Were Made Using Memristors!" In: *IEEE Circuits and Systems Magazine* 13.2 (2013), pp. 8–16.

[84] Karine Gandolfi, Christophe Mourtel, and Francis Olivier. "Electromagnetic Analysis: Concrete Results." In: *Cryptographic Hardware and Embedded Systems - CHES 2001, Third International Workshop, Paris, France, May 14-16, 2001, Proceedings*. Ed. by Çetin Kaya Koç, David Naccache, and Christof Paar. Vol. 2162. Lecture Notes in Computer Science Generators. Springer, 2001, pp. 251–261. DOI: 10.1007/3-540-44709-1_21. URL: https://doi.org/10.1007/3-540-44709-1_21.

[85] G.R.M. Garratt, Institution of Electrical Engineers, and Science Museum (Great Britain). *The Early History of Radio: From Faraday to Marconi*. History and Management of Technology. Institution

of Engineering and Technology, 1994. ISBN: 9780852968451. URL: https://books.google.fr/books?id=zhJTAAAAMAAJ.

[86] Daniel Genkin, Adi Shamir, and Eran Tromer. "RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis." In: *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. Lecture Notes in Computer Science. Springer, 2014, pp. 444–461. DOI: 10.1007/978-3-662-44371-2_25. URL: https://doi.org/10.1007/978-3-662-44371-2_25.

[87] Daniel Genkin, Lev Pachmanov, Itamar Pipman, and Eran Tromer. "Stealing Keys from PCs Using a Radio: Cheap Electromagnetic Attacks on Windowed Exponentiation." In: *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*. Ed. by Tim Güneysu and Helena Handschuh. Vol. 9293. Lecture Notes in Computer Science. Springer, 2015, pp. 207–228. DOI: 10.1007/978-3-662-48324-4_11. URL: https://doi.org/10.1007/978-3-662-48324-4_11.

[88] Daniel Genkin, Lev Pachmanov, Itamar Pipman, and Eran Tromer. "ECDH Key-Extraction via Low-Bandwidth Electromagnetic Attacks on PCs." In: *Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29 - March 4, 2016, Proceedings*. Ed. by Kazue Sako. Vol. 9610. Lecture Notes in Computer Science. Springer, 2016, pp. 219–235. DOI: 10.1007/978-3-319-29485-8_13. URL: https://doi.org/10.1007/978-3-319-29485-8_13.

[89] Daniel Genkin, Mihir Pattani, Roei Schuster, and Eran Tromer. "Synesthesia: Detecting Screen Content via Remote Acoustic Side Channels." In: *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*. IEEE, 2019, pp. 853–869. DOI: 10.1109/SP.2019.00074. URL: https://doi.org/10.1109/SP.2019.00074.

[90] Ilias Giechaskiel, Kasper Bonne Rasmussen, and Ken Eguro. "Leaky Wires: Information Leakage and Covert Communication Between FPGA Long Wires." In: *Proceedings of the 2018 on Asia Conference on Computer and Communications Security, AsiaCCS 2018, Incheon, Republic of Korea, June 04-08, 2018*. Ed. by Jong Kim, Gail-Joon Ahn, Seungjoo Kim, Yongdae Kim, Javier López, and Taesoo Kim. ACM, 2018, pp. 15–27. DOI: 10.1145/3196494.3196518. URL: https://doi.org/10.1145/3196494.3196518.

[91] Benjamin Jun Gilbert Goodwill, Josh Jaffe, Pankaj Rohatgi, et al. "A testing methodology for side-channel resistance valida-

tion." In: *NIST non-invasive attack testing workshop*. Vol. 7. 2011, pp. 115–136.

[92] Ognjen Glamocanin, Louis Coulon, Francesco Regazzoni, and Mirjana Stojilovic. "Are Cloud FPGAs Really Vulnerable to Power Analysis Attacks?" In: *2020 Design, Automation & Test in Europe Conference & Exhibition, DATE 2020, Grenoble, France, March 9-13, 2020*. IEEE, 2020, pp. 1007–1010. DOI: 10.23919/DATE48585.2020.9116481. URL: https://doi.org/10.23919/DATE48585.2020.9116481.

[93] Cezary Glowacz, Vincent Grosso, Romain Poussier, Joachim Schüth, and François-Xavier Standaert. "Simpler and More Efficient Rank Estimation for Side-Channel Security Assessment." In: *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers*. Ed. by Gregor Leander. Vol. 9054. Lecture Notes in Computer Science. Springer, 2015, pp. 117–129. ISBN: 978-3-662-48115-8. DOI: 10.1007/978-3-662-48116-5_6. URL: https://doi.org/10.1007/978-3-662-48116-5_6.

[94] Dennis R. E. Gnad, Jonas Krautter, and Mehdi Baradaran Tahoori. "Leaky Noise: New Side-Channel Attack Vectors in Mixed-Signal IoT Devices." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2019.3 (2019), pp. 305–339. DOI: 10.13154/tches.v2019.i3.305-339. URL: https://doi.org/10.13154/tches.v2019.i3.305-339.

[95] Anupam Golder, Debayan Das, Josef Danial, Santosh Ghosh, Shreyas Sen, and Arijit Raychowdhury. "Practical Approaches Toward Deep-Learning-Based Cross-Device Power Side-Channel Attack." In: *IEEE Trans. Very Large Scale Integr. Syst.* 27.12 (2019), pp. 2720–2733. DOI: 10.1109/TVLSI.2019.2926324. URL: https://doi.org/10.1109/TVLSI.2019.2926324.

[96] Nina Golyandina and Anatoly Zhigljavsky. *Singular Spectrum Analysis for Time Series*. Springer Science & Business Media, Jan. 2013. ISBN: 978-3-642-34913-3. DOI: 10.1007/978-3-642-34913-3.

[97] Travis Goodspeed, Sergey Bratus, Ricky Melgares, Rebecca Shapiro, and Ryan Speers. "Packets in Packets: Orson Welles' In-Band Signaling Attacks for Modern Radios." In: *5th USENIX Workshop on Offensive Technologies, WOOT'11, August 8, 2011, San Francisco, CA, USA, Proceedings*. Ed. by David Brumley and Michal Zalewski. USENIX Association, 2011, pp. 54–61. URL: http://static.usenix.org/event/woot11/tech/final_files/Goodspeed.pdf.

[98] Google. *Eddystone*. https://github.com/google/eddystone.

[99] Google. *Physical Web*. https://google.github.io/physical-web/.

[100] Ben Gras, Kaveh Razavi, Erik Bosman, Herbert Bos, and Cristiano Giuffrida. "ASLR on the Line: Practical Cache Attacks on the MMU." In: *24th Annual Network and Distributed System Security Symposium, NDSS 2017, San Diego, California, USA, February 26 - March 1, 2017*. The Internet Society, 2017. URL: https://www.ndss-symposium.org/ndss2017/ndss-2017-programme/aslrcache-practical-cache-attacks-mmu/.

[101] Joseph Gravellier, Jean-Max Dutertre, Yannick Teglia, and Philippe Loubet-Moundi. "SideLine: How Delay-Lines (May) Leak Secrets from your SoC." In: *CoRR* abs/2009.07773 (2020). arXiv: 2009.07773. URL: https://arxiv.org/abs/2009.07773.

[102] Great Scott Gadgets. *HackRF One*. https://greatscottgadgets.com/hackrf/.

[103] Great Scott Gadgets. *Ubertooth One*. https://greatscottgadgets.com/ubertoothone/.

[104] David Alan Grier. *What the Count of Monte Cristo Can Teach Us About Cybersecurity*. IEEE Spectrum. https://spectrum.ieee.org/tech-talk/telecom/security/what-the-count-of-monte-cristo-can-teach-us-about-cybersecurity. 2018.

[105] M. Grozing, J. Digel, Thomas Veigel, R. Bieg, Jianxiong Zhang, S. Brandl, M. Schmidt, C. Haslach, D. Markert, and W. Templ. "A RF Pulse-Width and Pulse-Position Modulator IC in 28 nm FDSOI CMOS." In: *2018 IEEE Nordic Circuits and Systems Conference (NORCAS): NORCHIP and International Symposium of System-on-Chip (SoC)* (2018), pp. 1–4.

[106] Daniel Gruss. "Software-based Microarchitectural Attacks." In: *CoRR* abs/1706.05973 (2017). arXiv: 1706.05973. URL: http://arxiv.org/abs/1706.05973.

[107] Daniel Gruss, Clémentine Maurice, and Stefan Mangard. "Rowhammer.js: A Remote Software-Induced Fault Attack in JavaScript." In: *Detection of Intrusions and Malware, and Vulnerability Assessment - 13th International Conference, DIMVA 2016, San Sebastián, Spain, July 7-8, 2016, Proceedings*. Ed. by Juan Caballero, Urko Zurutuza, and Ricardo J. Rodríguez. Vol. 9721. Lecture Notes in Computer Science. Springer, 2016, pp. 300–321. DOI: 10.1007/978-3-319-40667-1_15. URL: https://doi.org/10.1007/978-3-319-40667-1_15.

[108] M. Guri, M. Monitz, Y. Mirski, and Y. Elovici. "BitWhisper: Covert Signaling Channel between Air-Gapped Computers Using Thermal Manipulations." In: *2015 IEEE 28th Computer Security Foundations Symposium*. 2015, pp. 276–289.

[109] Mordechai Guri. "AiR-ViBeR: Exfiltrating Data from Air-Gapped Computers via Covert Surface ViBrAtIoNs." In: *CoRR* abs/2004.06195 (2020). arXiv: 2004.06195. URL: https://arxiv.org/abs/2004.06195.

[110] Mordechai Guri. "POWER-SUPPLaY: Leaking Data from Air-Gapped Systems by Turning the Power-Supplies Into Speakers." In: *IACR Cryptol. ePrint Arch.* 2020 (2020), p. 516. URL: https://eprint.iacr.org/2020/516.

[111] Mordechai Guri and Dima Bykhovsky. "aIR-Jumper: Covert air-gap exfiltration/infiltration via security cameras & infrared (IR)." In: *Comput. Secur.* 82 (2019), pp. 15–29. DOI: 10.1016/j.cose.2018.11.004. URL: https://doi.org/10.1016/j.cose.2018.11.004.

[112] Mordechai Guri, Andrey Daidakulov, and Yuval Elovici. "MAGNETO: Covert Channel between Air-Gapped Systems and Nearby Smartphones via CPU-Generated Magnetic Fields." In: *CoRR* abs/1802.02317 (2018). arXiv: 1802.02317. URL: http://arxiv.org/abs/1802.02317.

[113] Mordechai Guri, Matan Monitz, and Yuval Elovici. "USBee: Air-gap covert-channel via electromagnetic emission from USB." In: *14th Annual Conference on Privacy, Security and Trust, PST 2016, Auckland, New Zealand, December 12-14, 2016*. IEEE, 2016, pp. 264–268. DOI: 10.1109/PST.2016.7906972. URL: https://doi.org/10.1109/PST.2016.7906972.

[114] Mordechai Guri, Matan Monitz, and Yuval Elovici. "Bridging the Air Gap between Isolated Networks and Mobile Phones in a Practical Cyber-Attack." In: *ACM Trans. Intell. Syst. Technol.* 8.4 (May 2017). ISSN: 2157-6904. DOI: 10.1145/2870641. URL: https://doi.org/10.1145/2870641.

[115] Mordechai Guri, Yosef A. Solewicz, and Yuval Elovici. "MOSQUITO: Covert Ultrasonic Transmissions Between Two Air-Gapped Computers Using Speaker-to-Speaker Communication." In: *IEEE Conference on Dependable and Secure Computing, DSC 2018, Kaohsiung, Taiwan, December 10-13, 2018*. IEEE, 2018, pp. 1–8. DOI: 10.1109/DESEC.2018.8625124. URL: https://doi.org/10.1109/DESEC.2018.8625124.

[116] Mordechai Guri, Boris Zadov, and Yuval Elovici. "LED-it-GO: Leaking (A Lot of) Data from Air-Gapped Computers via the (Small) Hard Drive LED." In: *Detection of Intrusions and Malware, and Vulnerability Assessment - 14th International Conference, DIMVA 2017, Bonn, Germany, July 6-7, 2017, Proceedings*. Ed. by Michalis Polychronakis and Michael Meier. Vol. 10327. Lecture Notes in Computer Science. Springer, 2017, pp. 161–184. DOI: 10.1007/978-3-319-60876-1_8. URL: https://doi.org/10.1007/978-3-319-60876-1_8.

[117] Mordechai Guri, Boris Zadov, and Yuval Elovici. "ODINI: Escaping Sensitive Data From Faraday-Caged, Air-Gapped Computers via Magnetic Fields." In: *IEEE Trans. Inf. Forensics Secur.* 15 (2020), pp. 1190–1203. DOI: 10.1109/TIFS.2019.2938404. URL: https://doi.org/10.1109/TIFS.2019.2938404.

[118] Mordechai Guri, Gabi Kedma, Assaf Kachlon, and Yuval Elovici. "AirHopper: Bridging the air-gap between isolated networks and mobile phones using radio frequencies." In: *9th International Conference on Malicious and Unwanted Software: The Americas MALWARE 2014, Fajardo, PR, USA, October 28-30, 2014*. IEEE Computer Society, 2014, pp. 58–67. DOI: 10.1109/MALWARE.2014.6999418. URL: https://doi.org/10.1109/MALWARE.2014.6999418.

[119] Mordechai Guri, Assaf Kachlon, Ofer Hasson, Gabi Kedma, Yisroel Mirsky, and Yuval Elovici. "GSMem: Data Exfiltration from Air-Gapped Computers over GSM Frequencies." In: *24th USENIX Security Symposium, USENIX Security 15, Washington, D.C., USA, August 12-14, 2015*. Ed. by Jaeyeon Jung and Thorsten Holz. USENIX Association, 2015, pp. 849–864. URL: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/guri.

[120] Mordechai Guri, Yosef A. Solewicz, Andrey Daidakulov, and Yuval Elovici. "Fansmitter: Acoustic Data Exfiltration from (Speakerless) Air-Gapped Computers." In: *CoRR* abs/1606.05915 (2016). arXiv: 1606.05915. URL: http://arxiv.org/abs/1606.05915.

[121] Mordechai Guri, Yosef A. Solewicz, Andrey Daidakulov, and Yuval Elovici. "Acoustic Data Exfiltration from Speakerless Air-Gapped Computers via Covert Hard-Drive Noise ('DiskFiltration')." In: *Computer Security - ESORICS 2017 - 22nd European Symposium on Research in Computer Security, Oslo, Norway, September 11-15, 2017, Proceedings, Part II*. Ed. by Simon N. Foley, Dieter Gollmann, and Einar Snekkenes. Vol. 10493. Lecture Notes in Computer Science. Springer, 2017, pp. 98–115. DOI: 10.1007/978-3-319-66399-9_6. URL: https://doi.org/10.1007/978-3-319-66399-9_6.

[122] Mordechai Guri, Boris Zadov, Andrey Daidakulov, and Yuval Elovici. "xLED: Covert Data Exfiltration from Air-Gapped Networks via Switch and Router LEDs." In: *16th Annual Conference on Privacy, Security and Trust, PST 2018, Belfast, Northern Ireland, Uk, August 28-30, 2018*. Ed. by Kieran McLaughlin, Ali A. Ghorbani, Sakir Sezer, Rongxing Lu, Liqun Chen, Robert H. Deng, Paul Miller, Stephen Marsh, and Jason R. C. Nurse. IEEE Computer Society, 2018, pp. 1–12. DOI: 10.1109/PST.2018.8514196. URL: https://doi.org/10.1109/PST.2018.8514196.

[123] Mordechai Guri, Boris Zadov, Dima Bykhovsky, and Yuval Elovici. "PowerHammer: Exfiltrating Data From Air-Gapped Computers Through Power Lines." In: *IEEE Trans. Information Forensics and Security* 15 (2020), pp. 1879–1890. DOI: 10.1109/TIFS.2019.2952257. URL: https://doi.org/10.1109/TIFS.2019.2952257.

[124] Neil Hanley, Máire O'Neill, Michael Tunstall, and William P. Marnane. "Empirical evaluation of multi-device profiling side-channel attacks." In: *2014 IEEE Workshop on Signal Processing Systems, SiPS 2014, Belfast, United Kingdom, October 20-22, 2014*. IEEE, 2014, pp. 226–231. DOI: 10.1109/SiPS.2014.6986091. URL: https://doi.org/10.1109/SiPS.2014.6986091.

[125] Ragib Hasan, Nitesh Saxena, Tzipora Halevi, Shams Zawoad, and Dustin Rinehart. "Sensing-enabled channels for hard-to-detect command and control of mobile devices." In: *8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '13, Hangzhou, China - May 08 - 10, 2013*. Ed. by Kefei Chen, Qi Xie, Weidong Qiu, Ninghui Li, and Wen-Guey Tzeng. ACM, 2013, pp. 469–480. DOI: 10.1145/2484313.2484373. URL: https://doi.org/10.1145/2484313.2484373.

[126] Yu-ichi Hayashi, Naofumi Homma, Mamoru Miura, Takafumi Aoki, and Hideaki Sone. "A Threat for Tablet PCs in Public Space: Remote Visualization of Screen Images Using EM Emanation." In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*. Ed. by Gail-Joon Ahn, Moti Yung, and Ninghui Li. ACM, 2014, pp. 954–965. DOI: 10.1145/2660267.2660292. URL: https://doi.org/10.1145/2660267.2660292.

[127] Grant Hernandez and Kevin R. B. Butler. "Basebads: Automated security analysis of baseband firmware: poster." In: *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks, WiSec 2019, Miami, Florida, USA, May 15-17, 2019*. ACM, 2019, pp. 318–319. DOI: 10.1145/3317549.3326310. URL: https://doi.org/10.1145/3317549.3326310.

[128] Grant Hernandez and Marius Muench. *Emulating Samsung's Baseband for Security Testing*. Presentation at Black Hat USA 2020. Aug. 2020.

[129] Philip Hodgers, Francesco Regazzoni, Richard Gilmore, Ciara Moore, and Tobias Oder. "Secure Architectures of Future Emerging cryptography: State-of-the-Art in Physical Side-Channel Attacks and Resistant Technologies." In: (2016). URL: http://www.safecrypto.eu/wp-content/uploads/2015/02/SAFEcrypto_D7.1-Approved.pdf.

[130] Lin Huang and Qing Yang. *Low-cost GPS simulator – GPS spoofing by SDR*. Aug. 2015.

[131] Todd E. Humphreys, Brent M. Ledvina, Mark L. Psiaki, Brady W. O'Hanlon, and Paul M. Kintner. "Assessing the Spoofing Threat: Development of a Portable GPS Civilian Spoofer." In: 2008.

[132] "IEEE Standard for Information technology-Telecommunications and information exchange between systems Local and metropolitan area networks-Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications." In: *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)* (Dec. 2016), pp. 1–3534. ISSN: null. DOI: 10.1109/IEEESTD.2016.7786995.

[133] Omar Adel Ibrahim, Savio Sciancalepore, Gabriele Oligeri, and Roberto Di Pietro. "MAGNETO: Fingerprinting USB Flash Drives via Unintentional Magnetic Emissions." In: *CoRR* abs/2002.05905 (2020). arXiv: 2002.05905. URL: https://arxiv.org/abs/2002.05905.

[134] Samy Kamkar. "Drive it like you hacked it: New attacks and tools to wirelessly steal cars." In: *Presentation at DEFCON* 23 (2015).

[135] Bartek Kania. *VGASIGFM radio transmitter using a VGA graphics card*. 2009. URL: https://bk.gnarf.org/creativity/vgasig/vgasig.pdf.

[136] ED Kaplan and CJ Hegarty. "Understanding GPS/GNSS: principles and applications." In: *GNSS technology and applications series*. Artech House Publisher, 2017.

[137] C. Kasmi and J. Lopes Esteves. "IEMI Threats for Information Security: Remote Command Injection on Modern Smartphones." In: *IEEE Transactions on Electromagnetic Compatibility* 57.6 (2015), pp. 1752–1755.

[138] Bryce Kellogg, Aaron N. Parks, Shyamnath Gollakota, Joshua R. Smith, and David Wetherall. "Wi-fi backscatter: internet connectivity for RF-powered devices." In: *ACM SIGCOMM 2014 Conference, SIGCOMM'14, Chicago, IL, USA, August 17-22, 2014*. Ed. by Fabián E. Bustamante, Y. Charlie Hu, Arvind Krishnamurthy, and Sylvia Ratnasamy. ACM, 2014, pp. 607–618. DOI: 10.1145/2619239.2626319. URL: https://doi.org/10.1145/2619239.2626319.

[139] John Kelsey, Bruce Schneier, David A. Wagner, and Chris Hall. "Side Channel Cryptanalysis of Product Ciphers." In: *J. Comput. Secur.* 8.2/3 (2000), pp. 141–158. URL: http://content.iospress.com/articles/journal-of-computer-security/jcs133.

[140] Haider A. Khan, Nader Sehatbakhsh, Luong N. Nguyen, Milos Prvulovic, and Alenka G. Zajic. "Malware Detection in Embedded Systems Using Neural Network Model for Electromagnetic Side-Channel Signals." In: *J. Hardw. Syst. Secur.* 3.4 (2019), pp. 305–318. DOI: 10.1007/s41635-019-00074-w. URL: https://doi.org/10.1007/s41635-019-00074-w.

[141] Christopher David Kilgour. "A Bluetooth low-energy capture and analysis tool using software-defined radio." In: (2013).

[142] Paul C. Kocher. "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems." In: *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*. Ed. by Neal Koblitz. Vol. 1109. Lecture Notes in Computer Science. Springer, 1996, pp. 104–113. DOI: 10.1007/3-540-68697-5_9. URL: https://doi.org/10.1007/3-540-68697-5_9.

[143] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. "Differential Power Analysis." In: *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*. Ed. by Michael J. Wiener. Vol. 1666. Lecture Notes in Computer Science. Springer, 1999, pp. 388–397. DOI: 10.1007/3-540-48405-1_25. URL: https://doi.org/10.1007/3-540-48405-1_25.

[144] David Kohlbrenner and Hovav Shacham. "Trusted Browsers for Uncertain Times." In: *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016*. Ed. by Thorsten Holz and Stefan Savage. USENIX Association, 2016, pp. 463–480. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/kohlbrenner.

[145] T. Konefal and A. C. Marvin. "Prediction and measurement of EMC radiated immunity problems in interconnected digital electronic systems." In: *IEE Proceedings - Science, Measurement and Technology* 141.6 (Nov. 1994), pp. 464–470. ISSN: 1350-2344. DOI: 10.1049/ip-smt:19941453.

[146] Thomas Korak and Thomas Plos. "Applying Remote Side-Channel Analysis Attacks on a Security-Enabled NFC Tag." In: *Topics in Cryptology - CT-RSA 2013 - The Cryptographers' Track at the RSA Conference 2013, San Francisco,CA, USA, February 25-March 1, 2013. Proceedings*. Ed. by Ed Dawson. Vol. 7779. Lecture Notes in Computer Science. Springer, 2013, pp. 207–222. DOI: 10.1007/978-3-642-36095-4_14. URL: https://doi.org/10.1007/978-3-642-36095-4_14.

[147] Markus G. Kuhn. “Optical Time-Domain Eavesdropping Risks of CRT Displays.” In: *2002 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 12-15, 2002*. IEEE Computer Society, 2002, pp. 3–18. DOI: 10.1109/SECPRI.2002.1004358. URL: https://doi.org/10.1109/SECPRI.2002.1004358.

[148] Markus G. Kuhn. “Electromagnetic Eavesdropping Risks of Flat-Panel Displays.” In: *Privacy Enhancing Technologies, 4th International Workshop, PET 2004, Toronto, Canada, May 26-28, 2004, Revised Selected Papers*. Ed. by David M. Martin Jr. and Andrei Serjantov. Vol. 3424. Lecture Notes in Computer Science. Springer, 2004, pp. 88–107. DOI: 10.1007/11423409_7. URL: https://doi.org/10.1007/11423409_7.

[149] Markus G. Kuhn and Ross J. Anderson. “Soft Tempest: Hidden Data Transmission Using Electromagnetic Emanations.” In: *Information Hiding*. Ed. by David Aucsmith. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 124–142. ISBN: 978-3-540-49380-8.

[150] S. Kulkarni, I. Kazi, D. Seebacher, P. Singerl, F. Dielacher, W. Dehaene, and P. Reynaert. “Multi-standard wideband OFDM RF-PWM transmitter in 40nm CMOS.” In: *ESSCIRC Conference 2015 - 41st European Solid-State Circuits Conference (ESSCIRC)*. 2015, pp. 88–91.

[151] Joy Laskar, Babak Matinpour, and Sudipto Chakraborty. “Modern Receiver Front-Ends. Chapter 7: Design and Integration of Passive Components.” In: (Feb. 2004), pp. 143–190. DOI: 10.1002/0471474851.ch7. URL: http://dx.doi.org/10.1002/0471474851.ch7.

[152] Mengyuan Li, Yan Meng, Junyi Liu, Haojin Zhu, Xiaohui Liang, Yao Liu, and Na Ruan. “When CSI Meets Public WiFi: Inferring Your Mobile Phone Password via WiFi Signals.” In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*. Ed. by Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi. ACM, 2016, pp. 1068–1079. DOI: 10.1145/2976749.2978397. URL: https://doi.org/10.1145/2976749.2978397.

[153] H. Lin, C. Lu, H. Tsai, and T. Kung. “The analysis of EMI noise coupling mechanism for GPS reception performance degradation from SSD/USB module.” In: *2014 International Symposium on Electromagnetic Compatibility, Tokyo*. 2014, pp. 350–353.

[154] Han-Nien Lin. “Analysis of Platform Noise Effect on Performance of Wireless Communication Devices.” In: 2012.

[155] Moritz Lipp, Daniel Gruss, Raphael Spreitzer, Clémentine Maurice, and Stefan Mangard. "ARMageddon: Cache Attacks on Mobile Devices." In: *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016*. Ed. by Thorsten Holz and Stefan Savage. USENIX Association, 2016, pp. 549–564. URL: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/lipp.

[156] Moritz Lipp, Misiker Tadesse Aga, Michael Schwarz, Daniel Gruss, Clémentine Maurice, Lukas Raab, and Lukas Lamster. "Nethammer: Inducing Rowhammer Faults through Network Requests." In: *CoRR* abs/1805.04956 (2018). arXiv: 1805.04956. URL: http://arxiv.org/abs/1805.04956.

[157] Chia-Horng Liu. "On the design of OFDM signal detection algorithms for hardware implementation." In: *Proceedings of the Global Telecommunications Conference, 2003. GLOBECOM '03, San Francisco, CA, USA, 1-5 December 2003*. IEEE, 2003, pp. 596–599. DOI: 10.1109/GLOCOM.2003.1258308. URL: https://doi.org/10.1109/GLOCOM.2003.1258308.

[158] K. Liu, J. Lee, H. Hsu, and J. Chen. "An experimental study of WiFi performance impact due to SSC spread-percentage and modulation frequency." In: *2018 IEEE International Symposium on Electromagnetic Compatibility and 2018 IEEE Asia-Pacific Symposium on Electromagnetic Compatibility (EMC/APEMC)*. 2018, pp. 104–108.

[159] Vincent Liu, Aaron N. Parks, Vamsi Talla, Shyamnath Gollakota, David Wetherall, and Joshua R. Smith. "Ambient backscatter: wireless communication out of thin air." In: *ACM SIGCOMM 2013 Conference, SIGCOMM'13, Hong Kong, China, August 12-16, 2013*. Ed. by Dah Ming Chiu, Jia Wang, Paul Barford, and Srinivasan Seshan. ACM, 2013, pp. 39–50. DOI: 10.1145/2486001.2486015. URL: https://doi.org/10.1145/2486001.2486015.

[160] Joe Loughry. "Optical TEMPEST." In: *2018 International Symposium on Electromagnetic Compatibility (EMC EUROPE)*. IEEE. 2018, pp. 172–177.

[161] Joe Loughry and David A. Umphress. "Information leakage from optical emanations." In: *ACM Trans. Inf. Syst. Secur.* 5.3 (2002), pp. 262–289. DOI: 10.1145/545186.545189. URL: https://doi.org/10.1145/545186.545189.

[162] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007. ISBN: 978-0-387-30857-9.

[163] Martin Marinov. *TempestSDR*. URL: https://github.com/martinmarinov/TempestSDR.

[164] Athanasios Theodore Markettos. "Active electromagnetic attacks on secure hardware." PhD thesis. University of Cambridge, UK, 2011. URL: http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.609203.

[165] Steve Markgraf. *osmo-fl2k for using USB 3.0 VGA adapters as SDR transmitters*. 2018. URL: https://osmocom.org/projects/osmo-fl2k/wiki.

[166] Daniel P. Martin, Jonathan F. O'Connell, Elisabeth Oswald, and Martijn Stam. "Counting Keys in Parallel After a Side Channel Attack." In: *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*. Ed. by Tetsu Iwata and Jung Hee Cheon. Vol. 9453. Lecture Notes in Computer Science. Springer, 2015, pp. 313–337. DOI: 10.1007/978-3-662-48800-3_13. URL: https://doi.org/10.1007/978-3-662-48800-3_13.

[167] Luke Mather, Elisabeth Oswald, Joe Bandenburg, and Marcin Wójcik. "Does My Device Leak Information? An a priori Statistical Power Analysis of Leakage Detection Tests." In: *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*. Ed. by Kazue Sako and Palash Sarkar. Vol. 8269. Lecture Notes in Computer Science. Springer, 2013, pp. 486–505. DOI: 10.1007/978-3-642-42033-7_25. URL: https://doi.org/10.1007/978-3-642-42033-7_25.

[168] Célestin Matte, Mathieu Cunche, and Vincent Toubiana. "Does disabling Wi-Fi prevent my smartphone from sending Wi-Fi frames?" In: (2017).

[169] Oliver Mattos and Oskar Weigl. *Turning the Raspberry Pi Into an FM Transmitter*. URL: http://icrobotics.co.uk/wiki/index.php/Turning_the_Raspberry_Pi_Into_an_FM_Transmitter.

[170] N. Matyunin, J. Szefer, S. Biedermann, and S. Katzenbeisser. "Covert channels using mobile device's magnetic field sensors." In: *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*. 2016, pp. 525–532.

[171] Nikolay Matyunin, Nikolaos Athanasios Anagnostopoulos, Spyros Boukoros, Markus Heinrich, André Schaller, Maksim Kolinichenko, and Stefan Katzenbeisser. "Tracking Private Browsing Sessions using CPU-based Covert Channels." In: *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks, WiSec 2018, Stockholm, Sweden, June 18-20, 2018*. Ed. by Panos Papadimitratos, Kevin R. B. Butler, and Christina

Pöpper. ACM, 2018, pp. 63–74. DOI: 10.1145/3212480.3212489. URL: https://doi.org/10.1145/3212480.3212489.

[172] Nikolay Matyunin, Yujue Wang, Tolga Arul, Kristian Kullmann, Jakub Szefer, and Stefan Katzenbeisser. “MagneticSpy: Exploiting Magnetometer in Mobile Devices for Website and Application Fingerprinting.” In: *Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society*. 2019, pp. 135–149.

[173] Vasilios Mavroudis, Shuang Hao, Yanick Fratantonio, Federico Maggi, Christopher Kruegel, and Giovanni Vigna. “On the Privacy and Security of the Ultrasound Ecosystem.” In: *Proceedings on Privacy Enhancing Technologies* 2017.2 (2017), pp. 95–112.

[174] Olivier Meynard, Sylvain Guilley, Jean-Luc Danger, and Laurent Sauvage. “Far Correlation-based EMA with a precharacterized leakage model.” In: *Design, Automation and Test in Europe, DATE 2010, Dresden, Germany, March 8-12, 2010*. Ed. by Giovanni De Micheli, Bashir M. Al-Hashimi, Wolfgang Müller, and Enrico Macii. IEEE Computer Society, 2010, pp. 977–980. DOI: 10.1109/DATE.2010.5456906. URL: https://doi.org/10.1109/DATE.2010.5456906.

[175] Minicircuits. *ZEL-1724LN*. https://www.minicircuits.com/pdfs/ZEL-1724LN+.pdf.

[176] Minicircuits. *ZKL-1R5*. https://www.minicircuits.com/pdfs/ZKL-1R5+.pdf.

[177] Minicircuits. *ZX60-272LN*. https://ww2.minicircuits.com/pdfs/ZX60-272LN+.pdf.

[178] David P. Montminy, Rusty O. Baldwin, Michael A. Temple, and Eric D. Laspe. “Improving cross-device attacks using zero-mean unit-variance normalization.” In: *J. Cryptographic Engineering* 3.2 (2013), pp. 99–110. DOI: 10.1007/s13389-012-0038-y. URL: https://doi.org/10.1007/s13389-012-0038-y.

[179] S. Moon, S. Kim, D. Kim, D. Yi, S. Lee, and J. Shin. “Analysis and Estimation on EMI Effects in AP-DRAM Interface for a Mobile Platform.” In: *2016 IEEE 66th Electronic Components and Technology Conference (ECTC)*. 2016, pp. 1329–1334.

[180] A. B. M. Musa and Jakob Eriksson. “Tracking unmodified smartphones using wi-fi monitors.” In: *The 10th ACM Conference on Embedded Network Sensor Systems, SenSys ’12, Toronto, ON, Canada, November 6-9, 2012*. Ed. by M. Rasit Eskicioglu, Andrew Campbell, and Koen Langendoen. ACM, 2012, pp. 281–294. DOI: 10.1145/2426656.2426685. URL: https://doi.org/10.1145/2426656.2426685.

[181] *NSA ANT Catalog*. Tech. rep. Available at:https://www.eff.org/files/2014/01/06/20131230-appelbaum-nsa_ant_catalog.pdf. NSA, 2008.

[182] NSA. *NACSIM 5000, Tempest fundamentals*. Tech. rep. Document declassified in 2000 and available at https://cryptome.org/jya/nacsim-5000/nacsim-5000.htm. 1982.

[183] Muhammad Nabeel, Bastian Bloessl, and Falko Dressler. "On using BOC modulation in ultra-low power sensor networks for wildlife tracking." In: *IEEE Wireless Communications and Networking Conference, WCNC 2016, Doha, Qatar, April 3-6, 2016*. IEEE, 2016, pp. 1–6. DOI: 10.1109/WCNC.2016.7564858. URL: https://doi.org/10.1109/WCNC.2016.7564858.

[184] Rajalakshmi Nandakumar, Alex Takakuwa, Tadayoshi Kohno, and Shyamnath Gollakota. "CovertBand: Activity Information Leakage using Music." In: *IMWUT* 1.3 (2017), 87:1–87:24. DOI: 10.1145/3131897. URL: https://doi.org/10.1145/3131897.

[185] Ben Nassi, Yaron Pirutin, Adi Shamir, Yuval Elovici, and Boris Zadov. "Lamphone: Real-Time Passive Sound Recovery from Light Bulb Vibrations." In: *IACR Cryptol. ePrint Arch.* 2020 (2020), p. 708. URL: https://eprint.iacr.org/2020/708.

[186] NewAE Technology Inc. *NAE-HPROBE-15*. https://www.newae.com/products-1/NAE-HPROBE-15.

[187] Luong N. Nguyen, Chia-Lin Cheng, Milos Prvulovic, and Alenka G. Zajic. "Creating a Backscattering Side Channel to Enable Detection of Dormant Hardware Trojans." In: *IEEE Trans. Very Large Scale Integr. Syst.* 27.7 (2019), pp. 1561–1574. DOI: 10.1109/TVLSI.2019.2906547. URL: https://doi.org/10.1109/TVLSI.2019.2906547.

[188] Tyler Nighswander, Brent M. Ledvina, Jonathan Diamond, Robert Brumley, and David Brumley. "GPS software attacks." In: *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*. Ed. by Ting Yu, George Danezis, and Virgil D. Gligor. ACM, 2012, pp. 450–461. DOI: 10.1145/2382196.2382245. URL: https://doi.org/10.1145/2382196.2382245.

[189] Nordic Semiconductor. *PCA10040 Development Kit*. https://www.nordicsemi.com/Software-and-tools/Development-Kits/nRF52-DK.

[190] Nordic Semiconductor. *nRF5*. https://developer.nordicsemi.com/nRF5_SDK/nRF5_SDK_v14.x.x/nRF5_SDK_14.2.0_17b948a.zip.

[191] Nordic Semiconductor. *nRF52832*. https://www.nordicsemi.com/Products/Low-power-short-range-wireless/nRF52832.

[192] Nordic Semiconductor. *nRF52840*. https://www.nordicsemi.com/Software-and-Tools/Development-Kits/nRF52840-DK.

[193] Pieter AJ Nuyts, Patrick Reynaert, and Wim Dehaene. *Continuous-time digital front-ends for multistandard wireless transmission*. Springer, 2014.

[194] Colin O'Flynn and Zhizhang (David) Chen. "ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research." In: *Constructive Side-Channel Analysis and Secure Design - 5th International Workshop, COSADE 2014, Paris, France, April 13-15, 2014. Revised Selected Papers*. Ed. by Emmanuel Prouff. Vol. 8622. Lecture Notes in Computer Science. Springer, 2014, pp. 243–260. DOI: 10.1007/978-3-319-10175-0_17. URL: https://doi.org/10.1007/978-3-319-10175-0_17.

[195] Colin O'Flynn and Zhizhang Chen. "Power Analysis Attacks Against IEEE 802.15.4 Nodes." In: *Constructive Side-Channel Analysis and Secure Design - 7th International Workshop, COSADE 2016, Graz, Austria, April 14-15, 2016, Revised Selected Papers*. Ed. by François-Xavier Standaert and Elisabeth Oswald. Vol. 9689. Lecture Notes in Computer Science. Springer, 2016, pp. 55–70. DOI: 10.1007/978-3-319-43283-0_4. URL: https://doi.org/10.1007/978-3-319-43283-0_4.

[196] Colin O'Flynn and Alex Dewar. "On-Device Power Analysis Across Hardware Security Domains. Stop Hitting Yourself." In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2019.4 (2019), pp. 126–153. DOI: 10.13154/tches.v2019.i4.126-153. URL: https://doi.org/10.13154/tches.v2019.i4.126-153.

[197] David Oswald and Christof Paar. "Breaking Mifare DESFire MF3ICD40: Power Analysis and Templates in the Real World." In: *Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings*. Ed. by Bart Preneel and Tsuyoshi Takagi. Vol. 6917. Lecture Notes in Computer Science. Springer, 2011, pp. 207–222. DOI: 10.1007/978-3-642-23951-9_14. URL: https://doi.org/10.1007/978-3-642-23951-9_14.

[198] Elisabeth Oswald and Stefan Mangard. "Template Attacks on Masking - Resistance Is Futile." In: *Topics in Cryptology - CT-RSA 2007, The Cryptographers' Track at the RSA Conference 2007, San Francisco, CA, USA, February 5-9, 2007, Proceedings*. Ed. by Masayuki Abe. Vol. 4377. Lecture Notes in Computer Science. Springer, 2007, pp. 243–256. DOI: 10.1007/11967668_16. URL: https://doi.org/10.1007/11967668_16.

[199] Y. Papananos, K. Galanopoulos, N. Alexiou, and F. Dielacher. "A Mixed-Signal RF Pulse Width Modulator with High Time Resolution Utilizing Passive Delay Lines." In: *IEEE Transactions on Circuits and Systems II: Express Briefs* (2020), pp. 1–1.

[200] Thierry Parmentelat, Thierry Turletti, Walid Dabbous, Mohamed Naoufal Mahfoudi, and Francesco Bronzino. "Nepi-ng: An Efficient Experiment Control Tool in R2Lab." In: *Proceedings of the 12th International Workshop on Wireless Network Testbeds, Experimental Evaluation &#38; Characterization*. WiNTECH '18. New Delhi, India: ACM, 2018, pp. 56–65. ISBN: 978-1-4503-5930-6. DOI: 10.1145/3267204.3267213. URL: http://doi.acm.org/10.1145/3267204.3267213.

[201] Particle. *Particle Debugger*. https://store.particle.io/products/particle-debugger.

[202] Muhammad Touqir Pasha. "All-Digital PWM Transmitters." In: 2019.

[203] Penguin. *Penguin Wireless N USB Adapter for GNU / Linux (TPE-N150USB)*. https://fccid.io/NKR-DNUA93F.

[204] Thomas Plos. "Susceptibility of UHF RFID Tags to Electromagnetic Analysis." In: *Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008, San Francisco, CA, USA, April 8-11, 2008. Proceedings*. Ed. by Tal Malkin. Vol. 4964. Lecture Notes in Computer Science. Springer, 2008, pp. 288–300. DOI: 10.1007/978-3-540-79263-5_18. URL: https://doi.org/10.1007/978-3-540-79263-5_18.

[205] Thomas Plos. "Evaluation of the Detached Power Supply as Side-Channel Analysis Countermeasure for Passive UHF RFID Tags." In: *Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009, San Francisco, CA, USA, April 20-24, 2009. Proceedings*. Ed. by Marc Fischlin. Vol. 5473. Lecture Notes in Computer Science. Springer, 2009, pp. 444–458. DOI: 10.1007/978-3-642-00862-7_30. URL: https://doi.org/10.1007/978-3-642-00862-7_30.

[206] Thomas Plos and Christian Maierhofer. "On measuring the parasitic backscatter of sensor-enabled UHF RFID tags." In: *Inf. Secur. Tech. Rep.* 17.4 (2013), pp. 239–252. DOI: 10.1016/j.istr.2013.02.004. URL: https://doi.org/10.1016/j.istr.2013.02.004.

[207] Johannes Pohl and Andreas Noack. "Universal Radio Hacker: A Suite for Analyzing and Attacking Stateful Wireless Protocols." In: *12th USENIX Workshop on Offensive Technologies, WOOT 2018, Baltimore, MD, USA, August 13-14, 2018*. Ed. by Christian Rossow and Yves Younan. USENIX Association, 2018. URL: https://www.usenix.org/conference/woot18/presentation/pohl.

[208] Romain Poussier, François-Xavier Standaert, and Vincent Grosso. "Simple Key Enumeration (and Rank Estimation) Using Histograms: An Integrated Approach." In: *Cryptographic Hard-*

ware and Embedded Systems - CHES 2016 - 18th International Conference, Santa Barbara, CA, USA, August 17-19, 2016, Proceedings. Ed. by Benedikt Gierlichs and Axel Y. Poschmann. Vol. 9813. Lecture Notes in Computer Science. Springer, 2016, pp. 61–81. DOI: 10.1007/978-3-662-53140-2_4. URL: https://doi.org/10.1007/978-3-662-53140-2_4.

[209] Santos Merino Del Pozo and François-Xavier Standaert. "Blind Source Separation from Single Measurements Using Singular Spectrum Analysis." In: *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*. Ed. by Tim Güneysu and Helena Handschuh. Vol. 9293. Lecture Notes in Computer Science. Springer, 2015, pp. 42–59. DOI: 10.1007/978-3-662-48324-4_3. URL: https://doi.org/10.1007/978-3-662-48324-4_3.

[210] M. Prvulovic, A. Zajić, R. L. Callan, and C. J. Wang. "A Method for Finding Frequency-Modulated and Amplitude-Modulated Electromagnetic Emanations in Computer Systems." In: *IEEE Transactions on Electromagnetic Compatibility* 59.1 (2017), pp. 34–42.

[211] NF Pub. "Specification for the advanced encryption standard (aes)." In: *Tech. Rep. FIPS PUB 197, Federal Information Processing Standards* (2001).

[212] *Qsstv*. http://users.telenet.be/on4qz/.

[213] Jean-Jacques Quisquater and David Samyde. "ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards." In: *Smart Card Programming and Security, International Conference on Research in Smart Cards, E-smart 2001, Cannes, France, September 19-21, 2001, Proceedings*. Ed. by Isabelle Attali and Thomas P. Jensen. Vol. 2140. Lecture Notes in Computer Science. Springer, 2001, pp. 200–210. DOI: 10.1007/3-540-45418-7_17. URL: https://doi.org/10.1007/3-540-45418-7_17.

[214] *R2lab, an open tested located in an anechoic chamber for reproducible research in wireless networks*. http://fit-r2lab.inria.fr/. Accessed: 2018-05-07.

[215] RTL-SDR.com. *RTL-SDR*. URL: https://www.rtl-sdr.com/update-rtl-sdr-transmitting-1270-mhz/.

[216] *RTL TRX*. URL: https://github.com/tejeez/rtl-trx.

[217] F. Raab. "Radio Frequency Pulsewidth Modulation." In: *IEEE Transactions on Communications* 21.8 (Aug. 1973), pp. 958–966. ISSN: 1558-0857. DOI: 10.1109/TCOM.1973.1091763.

[218] Chethan Ramesh, Shivukumar B. Patil, Siva Nishok Dhanuskodi, George Provelengios, Sébastien Pillement, Daniel E. Holcomb, and Russell Tessier. "FPGA Side Channel Attacks without Physical Access." In: *26th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines, FCCM 2018, Boulder, CO, USA, April 29 - May 1, 2018*. IEEE Computer Society, 2018, pp. 45–52. DOI: 10.1109/FCCM.2018.00016. URL: https://doi.org/10.1109/FCCM.2018.00016.

[219] Craig Ramsay and Jasper Lohuis. *TEMPEST attacks against AES*. Fox-IT whitepaper. Available at: https://www.fox-it.com/en/insights/blogs/blog/tempest-attacks-aes/. June 2017.

[220] *Rasberry Pirate Radio FM Transmitter*. URL: https://www.rtl-sdr.com/raspberry-pirate-radio-fm-transmitter/.

[221] Redbear. *BLE Nano v2*. https://fccid.io/2AKGS-MBN2.

[222] Mathieu Renauld, François-Xavier Standaert, Nicolas Veyrat-Charvillon, Dina Kamel, and Denis Flandre. "A Formal Study of Power Variability Issues and Side-Channel Attacks for Nanoscale Devices." In: *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*. Ed. by Kenneth G. Paterson. Vol. 6632. Lecture Notes in Computer Science. Springer, 2011, pp. 109–128. DOI: 10.1007/978-3-642-20465-4_8. URL: https://doi.org/10.1007/978-3-642-20465-4_8.

[223] Rigado. *Rigado BDM301 Evaluation Board*. https://fccid.io/2AA9B04.

[224] Federico La Rocca. *gr-tempest*. URL: https://github.com/git-artes/gr-tempest.

[225] Eyal Ronen, Adi Shamir, Achi-Or Weingarten, and Colin O'Flynn. "IoT Goes Nuclear: Creating a Zigbee Chain Reaction." In: *IEEE Secur. Priv.* 16.1 (2018), pp. 54–62. DOI: 10.1109/MSP.2018.1331033. URL: https://doi.org/10.1109/MSP.2018.1331033.

[226] Yoann Roth, Jean-Baptiste Doré, Laurent Ros, and Vincent Berg. "The Physical Layer of Low Power Wide Area Networks: Strategies, Information Theory's Limit and Existing Solutions." In: *Advances in Signal Processing: Reviews, Vol. 1, Book Series*. Aug. 2018. URL: https://hal.archives-ouvertes.fr/hal-01872023.

[227] Jan Ruge, Jiska Classen, Francesco Gringoli, and Matthias Hollick. "Frankenstein: Advanced Wireless Fuzzing to Exploit New Bluetooth Escalation Targets." In: *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*. Ed. by Srdjan Capkun and Franziska Roesner. USENIX Association,

2020, pp. 19–36. URL: https://www.usenix.org/conference/usenixsecurity20/presentation/ruge.

[228] Bluetooth SIG. *Bluetooth 5.0 Core Specification*. https://www.bluetooth.com/specifications/archived-specifications/. 2016.

[229] Asanka Sayakkara, Luis Miralles-Pechuán, Nhien-An Le-Khac, and Mark Scanlon. "Cutting Through the Emissions: Feature Selection from Electromagnetic Side-Channel Data for Activity Detection." In: *Forensic Science International: Digital Investigation* 32 (2020), p. 300927.

[230] Falk Schellenberg, Dennis R. E. Gnad, Amir Moradi, and Mehdi Baradaran Tahoori. "An inside job: Remote power analysis attacks on FPGAs." In: *2018 Design, Automation & Test in Europe Conference & Exhibition, DATE 2018, Dresden, Germany, March 19-23, 2018*. Ed. by Jan Madsen and Ayse K. Coskun. IEEE, 2018, pp. 1111–1116. DOI: 10.23919/DATE.2018.8342177. URL: https://doi.org/10.23919/DATE.2018.8342177.

[231] Werner Schindler, Kerstin Lemke, and Christof Paar. "A Stochastic Model for Differential Side Channel Cryptanalysis." In: *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*. Ed. by Josyula R. Rao and Berk Sunar. Vol. 3659. Lecture Notes in Computer Science. Springer, 2005, pp. 30–46. DOI: 10.1007/11545262_3. URL: https://doi.org/10.1007/11545262_3.

[232] Bruce Schneier. *1834: The First Cyberattack*. Shneier on Security. https://www.schneier.com/blog/archives/2018/05/1834_the_first_.html. 2018.

[233] Matthias Schulz. "Teaching Your Wireless Card New Tricks: Smartphone Performance and Security Enhancements Through Wi-Fi Firmware Modifications." PhD thesis. Darmstadt University of Technology, Germany, 2018. URL: http://tuprints.ulb.tu-darmstadt.de/7243/.

[234] Matthias Schulz, Daniel Wegemer, and Matthias Hollick. *Nexmon: The C-based Firmware Patching Framework*. 2017. URL: https://nexmon.org.

[235] Matthias Schulz, Francesco Gringoli, Daniel Steinmetzer, Michael Koch, and Matthias Hollick. "Massive Reactive Smartphone-Based Jamming using Arbitrary Waveforms and Adaptive Power Control." In: *ACM Conference on Security and Privacy in Wireless & Mobile Networks (WiSec) 2017*. Boston, USA, July 2017, pp. 111–121.

[236] Matthias Schulz, Jakob Link, Francesco Gringoli, and Matthias Hollick. "Shadow Wi-Fi: Teaching Smartphones to Transmit Raw Signals and to Extract Channel State Information to Implement Practical Covert Channels over Wi-Fi." In: *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services, MobiSys 2018, Munich, Germany, June 10-15, 2018*. Ed. by Jörg Ott, Falko Dressler, Stefan Saroiu, and Prabal Dutta. ACM, 2018, pp. 256–268. DOI: 10.1145/3210240.3210333. URL: https://doi.org/10.1145/3210240.3210333.

[237] Michael Schwarz, Moritz Lipp, and Daniel Gruss. "JavaScript Zero: Real JavaScript and Zero Side-Channel Attacks." In: *NDSS*. 2018.

[238] Michael Schwarz, Clémentine Maurice, Daniel Gruss, and Stefan Mangard. "Fantastic Timers and Where to Find Them: High-Resolution Microarchitectural Attacks in JavaScript." In: *Financial Cryptography and Data Security - 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers*. Ed. by Aggelos Kiayias. Vol. 10322. Lecture Notes in Computer Science. Springer, 2017, pp. 247–267. DOI: 10.1007/978-3-319-70972-7_13. URL: https://doi.org/10.1007/978-3-319-70972-7_13.

[239] A Ciccomancini Scogna, H Shim, J Yu, CY Oh, S Cheon, N Oh, and DS Kim. "RFI and receiver sensitivity analysis in mobile electronic devices." In: *DesignCon*. Vol. 7. 2017, pp. 1–6.

[240] Manzel Seet. *PAL-Streamer*. 2020. URL: https://hackaday.io/project/171977-pal-streamer.

[241] Nordic Semiconductor. *Electronic ShockBurst*. https://infocenter.nordicsemi.com/topic/com.nordic.infocenter.sdk5.v15.0.0/group__nrf__esb.html.

[242] Nordic Semiconductor. *Gazel*. https://infocenter.nordicsemi.com/topic/com.nordic.infocenter.sdk5.v15.0.0/gzll_02_user_guide.html.

[243] Vitali Sepetnitsky, Mordechai Guri, and Yuval Elovici. "Exfiltration of Information from Air-Gapped Machines Using Monitor's LED Indicator." In: *IEEE Joint Intelligence and Security Informatics Conference, JISIC 2014, The Hague, The Netherlands, 24-26 September, 2014*. IEEE, 2014, pp. 264–267. DOI: 10.1109/JISIC.2014.51. URL: https://doi.org/10.1109/JISIC.2014.51.

[244] Ben Seri and Gregory Vishnepolsky. "Blueborne on android: Exploiting an rce over the air." In: *Tech. Rep.* (2017).

[245] A. Shamir and Y. Oren. "Remote Password Extraction from RFID Tags." In: *IEEE Transactions on Computers* 56 (June 2007), pp. 1292–1296. ISSN: 0018-9340. DOI: 10.1109/TC.2007.1050. URL: doi.ieeecomputersociety.org/10.1109/TC.2007.1050.

[246] Zhihui Shao, Mohammad A. Islam, and Shaolei Ren. "Your Noise, My Signal: Exploiting Switching Noise for Stealthy Data Exfiltration from Desktop Computers." In: *Proc. ACM Meas. Anal. Comput. Syst.* 4.1 (2020), 07:1–07:39. DOI: 10.1145/3379473. URL: https://doi.org/10.1145/3379473.

[247] Sigidwiki. *AM*. URL: https://www.sigidwiki.com/wiki/Amplitude_Modulation_(AM).

[248] Sigidwiki. *HamDRM*. URL: https://www.sigidwiki.com/wiki/WinDRM.

[249] Sigidwiki. *NBFM*. URL: https://www.sigidwiki.com/wiki/NFM_Voice.

[250] Sigidwiki. *OLIVIA*. URL: https://www.sigidwiki.com/wiki/Olivia.

[251] Sigidwiki. *PSK31*. URL: https://www.sigidwiki.com/wiki/PSK31.

[252] Sigidwiki. *RTTY50*. URL: https://www.sigidwiki.com/wiki/Radio_Teletype_(RTTY).

[253] Sigidwiki. *SSTV*. URL: https://www.sigidwiki.com/wiki/SSTV.

[254] Mridula Singh, Patrick Leu, and Srdjan Capkun. "UWB with Pulse Reordering: Securing Ranging against Relay and Physical-Layer Attacks." In: *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society, 2019. URL: https://www.ndss-symposium.org/ndss-paper/uwb-with-pulse-reordering-securing-ranging-against-relay-and-physical-layer-attacks/.

[255] Kevin Slattery and Harry Skinner. *Platform interference in wireless systems: Models, measurement, and mitigation*. Newnes, 2011.

[256] E. Sourour, H. El-Ghoroury, and D. McNeill. "Frequency offset estimation and correction in the IEEE 802.11a WLAN." In: *IEEE 60th Vehicular Technology Conference, 2004. VTC2004-Fall. 2004*. Vol. 7. Sept. 2004, 4923–4927 Vol. 7. DOI: 10.1109/VETECF.2004.1405033.

[257] Dominic Spill. *Ridiculous Radios*. 2018. URL: https://hackaday.com/wp-content/uploads/2018/12/Ridiculous-Radios-Supercon-2018.pdf.

[258] Luigi Sportiello. "'Internet of Smart Cards': A pocket attacks scenario." In: *Int. J. Crit. Infrastructure Prot.* 26 (2019). DOI: 10.1016/j.ijcip.2019.05.005. URL: https://doi.org/10.1016/j.ijcip.2019.05.005.

[259] Colin Stagner. “Detecting and locating electronic devices using their unintended electromagnetic emissions.” PhD thesis. Missouri University of Science and Technology, 2013.

[260] François-Xavier Standaert, Eric Peeters, Gaël Rouvroy, and Jean-Jacques Quisquater. “An Overview of Power Analysis Attacks Against Field Programmable Gate Arrays.” In: *Proceedings of the IEEE* 94.2 (2006), pp. 383–394. DOI: 10.1109/JPROC.2005.862437. URL: https://doi.org/10.1109/JPROC.2005.862437.

[261] Venkatachalam Subramanian, A. Selcuk Uluagac, Hasan Cam, and Raheem A. Beyah. “Examining the characteristics and implications of sensor side channels.” In: *Proceedings of IEEE International Conference on Communications, ICC 2013, Budapest, Hungary, June 9-13, 2013*. IEEE, 2013, pp. 2205–2210. DOI: 10.1109/ICC.2013.6654855. URL: https://doi.org/10.1109/ICC.2013.6654855.

[262] Y. Sun and P. Willett. “Hough transform for long chirp detection.” In: *IEEE Transactions on Aerospace and Electronic Systems* 38.2 (2002), pp. 553–569.

[263] TEKBOX. *TBWA2*. https://www.tekbox.com/product/tbwa2-wideband-rf-amplifiers/.

[264] TEKBOX. *TEKBOX TBWA2*. https://www.tekbox.com/product/TBWA2_Manual.pdf.

[265] *TEMPEST: A Signal Problem*. Tech. rep. Available at:https://www.nsa.gov/news-features/declassified-documents/cryptologic-spectrum/assets/files/tempest.pdf. NSA, 1972.

[266] TP-LINK. *TL-ANT2424B*. https://www.tp-link.com/us/business-networking/antenna-and-accessory/tl-ant2424b/.

[267] Joachim Tapparel, Orion Afisiadis, Paul Mayoraz, Alexios Balatsoukas-Stimming, and Andreas Burg. *An Open-Source LoRa Physical Layer Prototype on GNU Radio*. 2020. arXiv: 2002.08208 [eess.SP].

[268] Joe Taylor. *FT4*. URL: https://physics.princeton.edu/pulsar/k1jt/FT4_Protocol.pdf.

[269] Joe Taylor. *WSPR Instructions*. URL: https://physics.princeton.edu/pulsar/K1JT/WSPR_Instructions.TXT.

[270] Joe Taylor. *WSPR*. URL: https://physics.princeton.edu/pulsar/K1JT/WSPR_QST_Nov_2010.pdf.

[271] Pietro Tedeschi, Gabriele Oligeri, and Roberto Di Pietro. “Leveraging Jamming to Help Drones Complete Their Mission.” In: *IEEE Access* 8 (2020), pp. 5049–5064. DOI: 10.1109/ACCESS.2019.2963105. URL: https://doi.org/10.1109/ACCESS.2019.2963105.

[272] *The Thing*. URL: https://www.cryptomuseum.com/covert/bugs/thing/index.htm.

[273] Erik Thiele. *Tempest for Eliza*. 2001. URL: http://www.erikyyy.de/tempest/.

[274] Nils Ole Tippenhauer, Christina Pöpper, Kasper Bonne Rasmussen, and Srdjan Capkun. "On the requirements for successful GPS spoofing attacks." In: *Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, Chicago, Illinois, USA, October 17-21, 2011*. Ed. by Yan Chen, George Danezis, and Vitaly Shmatikov. ACM, 2011, pp. 75–86. DOI: 10.1145/2046707.2046719. URL: https://doi.org/10.1145/2046707.2046719.

[275] Martin Untersinger. *Les très indiscrètes puces des objets connectés*. News article. Sept. 2018. URL: https://www.lemonde.fr/pixels/article/2018/07/25/les-tres-indiscretes-puces-des-objets-connectes_5335566_4408996.html.

[276] Jo Van Bulck. *Microarchitectural Side-Channel Attacks for Privileged Software Adversaries*. eng. 2020. URL: $$Uhttps://lirias.kuleuven.be/retrieve/585107$$Dvanbulck-thesis-final.pdf[freelyavailable].

[277] Mathy Vanhoef. *modwifi-ath9k-htc*. https://github.com/vanhoefm/modwifi-ath9k-htc.

[278] Mathy Vanhoef. *modwifi*. https://github.com/vanhoefm/modwifi.

[279] Mathy Vanhoef and Frank Piessens. "Advanced Wi-Fi attacks using commodity hardware." In: *Proceedings of the 30th Annual Computer Security Applications Conference, ACSAC 2014, New Orleans, LA, USA, December 8-12, 2014*. Ed. by Charles N. Payne Jr., Adam Hahn, Kevin R. B. Butler, and Micah Sherr. ACM, 2014, pp. 256–265. DOI: 10.1145/2664243.2664260. URL: https://doi.org/10.1145/2664243.2664260.

[280] Mathy Vanhoef and Eyal Ronen. "Dragonblood: Analyzing the Dragonfly Handshake of WPA3 and EAP-pwd." In: *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*. IEEE, 2020, pp. 517–533. DOI: 10.1109/SP40000.2020.00031. URL: https://doi.org/10.1109/SP40000.2020.00031.

[281] Robert Vautard, Pascal Yiou, and Michael Ghil. "Singular-spectrum analysis: A toolkit for short, noisy chaotic signals." In: *Physica D: Nonlinear Phenomena* 58.1-4 (1992), pp. 95–126.

[282] Victor van der Veen, Yanick Fratantonio, Martina Lindorfer, Daniel Gruss, Clementine Maurice, Giovanni Vigna, Herbert Bos, Kaveh Razavi, and Cristiano Giuffrida. "Drammer: Deterministic Rowhammer Attacks on Mobile Platforms." In: *CCS*. Pwnie Award for Best Privilege Escalation Bug, Android Security Reward, CSAW Best Paper Award, DCSR Paper Award. Oct. 2016. URL: Paper=https://vvdveen.com/publications/drammer.pdfWeb=https://www.vusec.net/projects/drammerCode=https://github.com/vusec/drammer.

[283] Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. "Security Evaluations beyond Computing Power." In: *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*. Ed. by Thomas Johansson and Phong Q. Nguyen. Vol. 7881. Lecture Notes in Computer Science. Springer, 2013, pp. 126–141. DOI: 10.1007/978-3-642-38348-9_8. URL: https://doi.org/10.1007/978-3-642-38348-9_8.

[284] Nicolas Veyrat-Charvillon, Benoît Gérard, Mathieu Renauld, and François-Xavier Standaert. "An Optimal Key Enumeration Algorithm and Its Application to Side-Channel Attacks." In: *Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers*. Ed. by Lars R. Knudsen and Huapeng Wu. Vol. 7707. Lecture Notes in Computer Science. Springer, 2012, pp. 390–406. DOI: 10.1007/978-3-642-35999-6_25. URL: https://doi.org/10.1007/978-3-642-35999-6_25.

[285] Tien Dang Vo-Huu, Triet Dang Vo-Huu, and Guevara Noubir. "Fingerprinting Wi-Fi Devices Using Software Defined Radios." In: *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks, WISEC 2016, Darmstadt, Germany, July 18-22, 2016*. Ed. by Matthias Hollick, Panos Papadimitratos, and William Enck. ACM, 2016, pp. 3–14. DOI: 10.1145/2939918.2939936. URL: https://doi.org/10.1145/2939918.2939936.

[286] Satohiro Wakabayashi, Seita Maruyama, Tatsuya Mori, Shigeki Goto, Masahiro Kinugawa, and Yu-ichi Hayashi. "A Feasibility Study of Radio-frequency Retroreflector Attack." In: *12th USENIX Workshop on Offensive Technologies, WOOT 2018, Baltimore, MD, USA, August 13-14, 2018*. Ed. by Christian Rossow and Yves Younan. USENIX Association, 2018. URL: https://www.usenix.org/conference/woot18/presentation/wakabayashi.

[287] J. S. Walling, H. Lakdawala, Y. Palaskas, A. Ravi, O. Degani, K. Soumyanath, and D. J. Allstot. "A Class-E PA With Pulse-Width and Pulse-Position Modulation in 65 nm CMOS." In: *IEEE Journal of Solid-State Circuits* 44.6 (2009), pp. 1668–1678.

[288] Anran Wang, Vikram Iyer, Vamsi Talla, Joshua R. Smith, and Shyamnath Gollakota. “FM Backscatter: Enabling Connected Cities and Smart Fabrics.” In: *14th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2017, Boston, MA, USA, March 27-29, 2017*. Ed. by Aditya Akella and Jon Howell. USENIX Association, 2017, pp. 243–258. URL: https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/wang-anran.

[289] Kang Wang. “Time and Position Spoofing with Open Source Projects.” In: 2015.

[290] Ruize Wang. “Deep Learning Based Side-Channel Analysis of AES Based on Far Field.” 2020. URL: https://www.diva-portal.org/smash/get/diva2:1449832/FULLTEXT01.pdf.

[291] Ruize Wang, Huanyu Wang, and Elena Dubrova. “Far Field EM Side-Channel Attack on AES Using Deep Learning.” In: *Proceedings of the 4th ACM Workshop on Attacks and Solutions in Hardware Security*. ASHES’20. Virtual Event, USA: Association for Computing Machinery, 2020, pp. 35–44. ISBN: 9781450380904. DOI: 10.1145/3411504.3421214. URL: https://doi.org/10.1145/3411504.3421214.

[292] Bernard L. Welch. “The generalization of ‘student’s’ problem when several different population variances are involved.” In: *Biometrika* 34.1-2 (Jan. 1947), pp. 28–35. ISSN: 0006-3444. DOI: 10.1093/biomet/34.1-2.28. eprint: http://oup.prod.sis.lan/biomet/article-pdf/34/1-2/28/553093/34-1-2-28.pdf. URL: https://doi.org/10.1093/biomet/34.1-2.28.

[293] Jasper G. J. van Woudenberg, Marc F. Witteman, and Bram Bakker. “Improving Differential Power Analysis by Elastic Alignment.” In: *Topics in Cryptology - CT-RSA 2011 - The Cryptographers’ Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*. Ed. by Aggelos Kiayias. Vol. 6558. Lecture Notes in Computer Science. Springer, 2011, pp. 104–119. DOI: 10.1007/978-3-642-19074-2_8. URL: https://doi.org/10.1007/978-3-642-19074-2_8.

[294] Chen Yan, Hocheol Shin, Connor Bolton, Wenyuan Xu, Yongdae Kim, and Kevin Fu. “SoK: A Minimalist Approach to Formalizing Analog Sensor Security.” In: *2020 IEEE Symposium on Security and Privacy, SP 2020, San Francisco, CA, USA, May 18-21, 2020*. IEEE, 2020, pp. 233–248. DOI: 10.1109/SP40000.2020.00026. URL: https://doi.org/10.1109/SP40000.2020.00026.

[295] Zhice Yang, Qianyi Huang, and Qian Zhang. “NICScatter: Backscatter as a Covert Channel in Mobile Devices.” In: *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking, MobiCom 2017, Snowbird, UT, USA,*

October 16 - 20, 2017. Ed. by Kobus van der Merwe, Ben Greenstein, and Kannan Srinivasan. ACM, 2017, pp. 356–367. DOI: 10.1145/3117811.3117814. URL: https://doi.org/10.1145/3117811.3117814.

[296] Zhice Yang, Qianyi Huang, and Qian Zhang. "Backscatter as a Covert Channel in Mobile Devices." In: *GetMobile Mob. Comput. Commun.* 22.1 (2018), pp. 31–34. DOI: 10.1145/3229316.3229327. URL: https://doi.org/10.1145/3229316.3229327.

[297] Ted Yapo. *Experimental software-defined radio using a serial port*. 2018. URL: https://github.com/tedyapo/serial-port-sdr.

[298] Xin Ye, Thomas Eisenbarth, and William Martin. "Bounded, yet Sufficient? How to Determine Whether Limited Side Channel Information Enables Key Recovery." In: *Smart Card Research and Advanced Applications - 13th International Conference, CARDIS 2014, Paris, France, November 5-7, 2014. Revised Selected Papers*. Ed. by Marc Joye and Amir Moradi. Vol. 8968. Lecture Notes in Computer Science. Springer, 2014, pp. 215–232. DOI: 10.1007/978-3-319-16763-3_13. URL: https://doi.org/10.1007/978-3-319-16763-3_13.

[299] Yepkit. *YKUSH*. https://www.yepkit.com/products/ykush.

[300] Alenka Zajic, Milos Prvulovic, and Derrick Chu. "Path loss prediction for electromagnetic side-channel signals." In: *Antennas and Propagation (EUCAP), 2017 11th European Conference on*. IEEE. 2017, pp. 3877–3881.

[301] A. Zajić and M. Prvulovic. "Experimental Demonstration of Electromagnetic Information Leakage From Modern Processor-Memory Systems." In: *IEEE Transactions on Electromagnetic Compatibility* 56.4 (2014), pp. 885–893.

[302] Zeptobars. *nRF51822 - Bluetooth LE SoC : weekend die-shot*. https://zeptobars.com/en/read/nRF51822-Bluetooth-LE-SoC-Cortex-M0. Aug. 2014.

[303] Zihao Zhan, Zhenkai Zhang, and Xenofon Koutsoukos. "BitJabber: The World's Fastest Electromagnetic Covert Channel." In: *2020 IEEE International Test Conference (ITC)*. IEEE. 2020.

[304] Zhenkai Zhang, Zihao Zhan, Daniel Balasubramanian, Xenofon D. Koutsoukos, and Gabor Karsai. "Triggering Rowhammer Hardware Faults on ARM: A Revisit." In: *Proceedings of the 2018 Workshop on Attacks and Solutions in Hardware Security, ASHES@CCS 2018, Toronto, ON, Canada, October 19, 2018*. Ed. by Chip-Hong Chang, Ulrich Rührmair, Daniel Holcomb, and Jorge Guajardo. ACM, 2018, pp. 24–33. DOI: 10.1145/3266444.3266454. URL: https://doi.org/10.1145/3266444.3266454.

[305] Zhenkai Zhang, Zihao Zhan, Daniel Balasubramanian, Bo Li, Peter Volgyesi, and Xenofon Koutsoukos. "Leveraging EM Side-Channel Information to Detect Rowhammer Attacks." In: *2020 IEEE Symposium on Security and Privacy (S&P'20)*. 2020, pp. 862–879.

[306] Mark Zhao and G. Edward Suh. "FPGA-Based Remote Power Side-Channel Attacks." In: *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*. IEEE Computer Society, 2018, pp. 229–244. DOI: 10.1109/SP.2018.00049. URL: https://doi.org/10.1109/SP.2018.00049.

[307] bastibl. https://github.com/bastibl/gr-ieee802-11.

[308] bolek42. *bolek42/rsa-sdr*. 2017. URL: https://github.com/bolek42/rsa-sdr.

[309] cnlhor. *ESP8266 Analog Broadcast Television Interface*. 2016. URL: https://github.com/cnlohr/channel3.

[310] *gnss-sdr*. URL: https://gnss-sdr.org/.

[311] eurecom s3 group. *screaming channels open-source code*. https://github.com/eurecom-s3/screaming_channels. 2018.

[312] siro@das labor.org. *VGAtoIQBaseband*. 2013. URL: https://wiki.das-labor.org/w/VGAtoBaseband.

COLOPHON

This thesis was typeset in LaTeX using the `classicthesis` template. The author would like to thank André Miede and Ivo Pletikosić for developing and making freely available such a beatiful and elegant template.