EURECOM
Sophia Antipolis

# Screaming Channels

**When Electromagnetic Side Channels Meet Radio Transceivers**

Giovanni Camurati / Sebastian Poeplau / Marius Muench / Tom Hayes / Aurélien Francillon
EURECOM, Sophia Antipolis, France

## Introduction

This white paper contains background information and supplementary material for our Black Hat USA 2018 talk. We want to hint the reader that this is an adapted version of our academic paper with the same title, which will appear at the ACM Conference on Computer and Communications Security (CCS) 2018.

We kindly ask to cite the scientific paper when referring to our work; the full version of the paper is available at http://s3.eurecom.fr/docs/ccs18_camurati.pdf.

## Motivation

The drive for ever smaller and cheaper components in microelectronics has popularized so-called *mixed-design circuits*, i.e., circuits in which analog and digital circuitry reside on the same die. A typical example is a WiFi chip featuring a (digital) microcontroller for cryptography and other computations as well as the (analog) radio. The special challenge of such designs is to separate the "noisy" digital circuits from the sensitive analog side of the system. We show that insufficient separation of digital and analog components leads to novel side-channel attacks that can break cryptography implemented in mixed-design chips over potentially large distances.

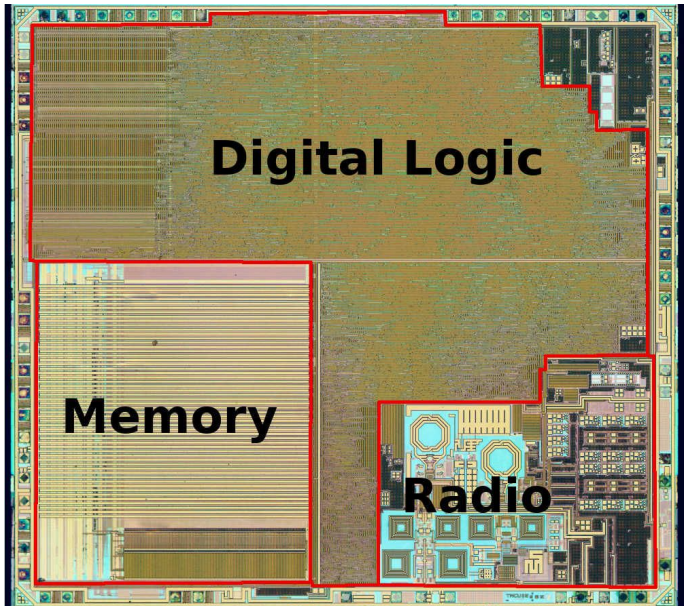
Modern cryptographic algorithms have been designed with a wide range of attacks in mind and are thus hardened against the more traditional ways of breaking the secrecy that cryptography is meant to provide. More recently, a lot of research attention has therefore been focused on side-channel attacks. In a side-channel scenario, attackers do not break the algorithm, but instead gain knowledge of the algorithm's internal state by means of observing its physical implementation; whenever such knowledge is not meant to be public, it can be used to undermine the algorithm's integrity. For example, Kocher et al. showed in 1999 that observing the power consumption of a smart card running an unprotected implementation of DES allows an attacker to guess the key, effectively breaking the cryptosystem. Their results spawned a long line of work on side-channel attacks against the implementations of all common cryptographic algorithms.

Measuring a system's power consumption usually requires direct physical access to the chip and potentially invasive application of probes inside it. A more discreet avenue of attack that has since been proved feasible is through electromagnetic emissions (*EM attacks*). Such attacks make use of inadvertent electromagnetic emissions that are common in digital circuitry - the key observation is that the emanations correlate with certain computations. EM attacks often use specialized H-field

antennas in close proximity of the target chip, typically within millimeters. Nevertheless, successful attacks against higher-power devices have been mounted from as far as several meters. However, the emissions of low-power devices are very weak and do not allow for attacks over larger distances.

## Root of the problem: Noise coupling

Our key observation is that in mixed-design radio chips the processor's activity leaks into the analog portion of the chip, where it is amplified, upconverted and broadcast as part of the regular radio output. This leakage is not due to a bad chip design, but to a fundamental difficulty in designing mixed-signal chips. Indeed chip substrate (the silicon on which the chip is manufactured) is conducting, and coupling occurs between the digital and analog sections of the chip. See, for example, the picture of an annotated mixed-design chip, which shows an nRF51822, a widespread Bluetooth LE chip.[1] In this picture the chip was only depackaged but not de-layered; therefore, the digital sections show only the top metal layer, which looks uniform, while the analog parts appear as large features. Those large features are on-chip inductors, capacitors, power transistors, etc.




### Digital noise "sinks"

This design is typical for a chip with mixed signals. The most sensitive parts of the analog chip are visible on the top left: the RX chain (RX, RF PLL, RX Low Pass Filter and finally the Analog to Digital converter) while the TX chain is closer to the center (TX Digital-to-Analog converter (DAC) and TX). In an RX chain, very low power radio signals are received from the antenna. The datasheet of this specific chip mentions a sensitivity of -91 dBm (roughly 800 femto watts!) for reception, while the TX chain is capable of transmitting up to 20dBm (100 mW) with the on-chip power amplifier.

[1] "nRF51822 - Bluetooth LE SoC : weekend die-shot" - CC-BY – Modified with annotations. Original by zeptobars: https://zeptobars.com/en/read/nRF51822-Bluetooth-LE-SoC-Cortex-M0

As a consequence the RX chain is very sensitive to noise, which is why it is placed in the very corner of the chip ("RX" on the picture). In fact, the farther an analog circuit is from a digital one, the less noise it receives and the better the radio receiver sensitivity. On the other hand, TX chain performance is less sensitive to the digital noise, as the radio symbols are generated locally by the digital baseband. and can be generated at a sufficiently high power level. At the same time, the TX chain is closer to noisy digital signals and therefore less isolated from the digital noise.

## Digital noise "sources" and noise propagation

In general, noise in a chip originates from various sources, and can be distinguished as noise from the power supply, thermal noise and digital noise generated by the switching activity of transistors in the logic ports. Digital noise is interesting as it is somehow related to the logic behavior of the chip and of software running on it. This noise can be propagated by different means, but literature shows that the most important propagation mechanism is through the substrate. The so called "substrate noise coupling"[2] is known to be problematic for mixed-signal chips: if it is too high, the chip may not function at all. Substrate is the "bulk" silicon on which the chip is manufactured and is made of P-doped silicon. It has a typical resistance of 0.5 Ω/m to 20 Ω/m and forms the P of the NPN transistors (while PNP rely on an "N-Well"). As the substrate is P-doped it is somewhat conductive.

## The side channel: from digital noise to noise in radio emissions

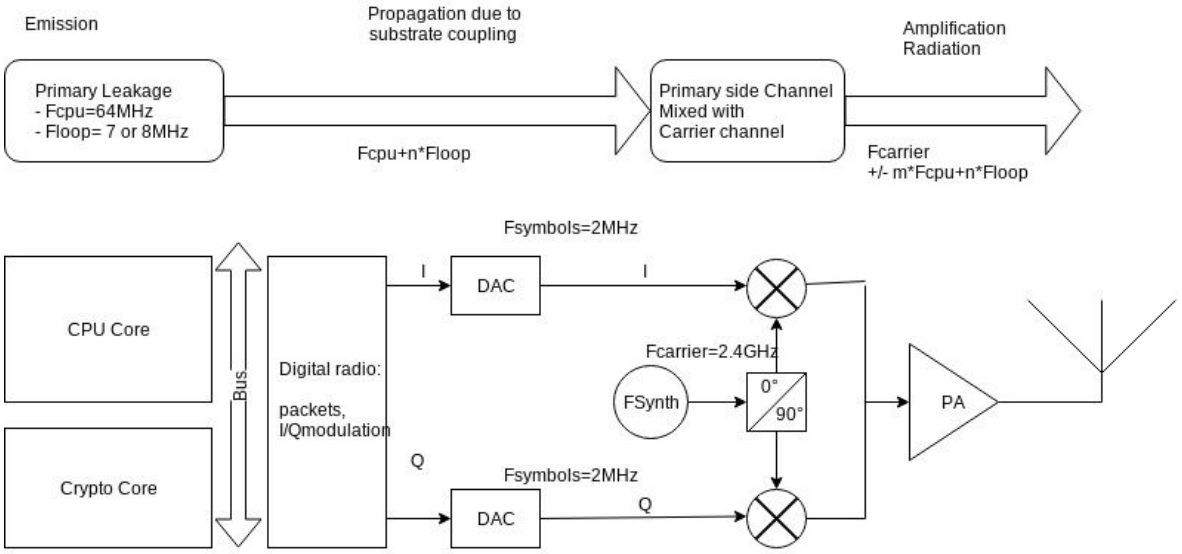
Due to the noise coupling effect, some "signal" ("noise" is only signal that we cannot control, explain or exploit...) from the cryptographic computations reaches the radio transmitter. The figure below shows a periodic computation at that occurs 8MHz (e.g., a loop) which modulates (consumes power) the CPU clock (at 64MHz). Because this is not a linear phenomenon, harmonics are produced. The signal generated is at (64+n*8)MHz (n=1 for the main components and then 2, 4, etc for the harmonics). We call this the primary leakage.

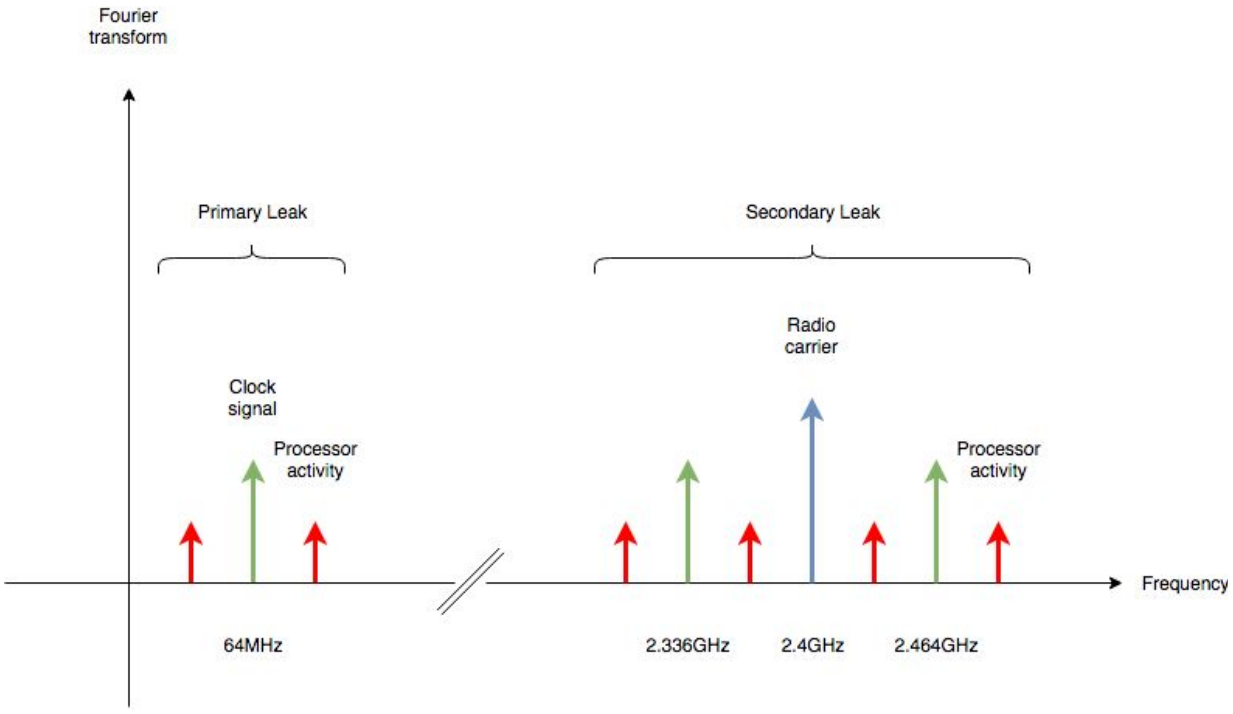
In the radio transmitter, the data to transmit is transformed into digital symbols (I and Q) by the digital radio which are then converted to analog symbols by the Digital to Analog Converter. The mixer (cross symbol) is then generating the final high frequency modulated signal which is amplified and transmitted by the antenna. While the goal of the radio transmitter is only to transmit this data, the "noise" also reaches the transmitter due to substrate coupling. Indeed we observe a "secondary leak" at m*2.4GHz±(k*64±n*8)MHz, e.g. at 2.464GHz.

This signal is not only a high component of the harmonics of the primary leakage, but it is really a component that is mixed again once it reaches the radio. We verified this by changing carrier

[2] Those slides present a good overview and provide many good references about substrate coupling http://www2.ece.rochester.edu/~parihar/pres/Pres_SubstrateCoupling.pdf

frequency (FSynth on the figure below) and were able to verify that the secondary leakage was shifted too. Which confirms that the noise coupling is then upmixed and transmitted to the power amplifier and antennas. In general, we confirmed this behavior through in-depth measurements on a nRF52832 chip, both with a H-field probe near the device for the primary leakage and several antennas at different distances for the secondary leakage.



# Key Recovery Attack

We show that it is possible to recover the original leaked EM signal (primary leakage in the above figures) and apply variations of known side-channel analysis techniques; we call our variations *Correlation Radio Analysis (CRA)* and *Template Radio Analysis (TRA)*, inspired by the corresponding classes of power and EM attacks. Using the example of a commercial off-the-shelf Bluetooth device we demonstrate that cryptographic keys that are used by AES form mbedTLS can be recovered just observing the device's radio emissions from a distance. Note that our attack does not depend on the actual data that the device sends - all we need is the fact that the radio is transmitting while the processor carries out the cryptographic operations.

The goal of the attack is to recover the key of an AES computation carried out by the processor of a commercial Bluetooth chip, nRF52832 by Nordic Semiconductor, using only the radio signal that the chip emits. We will first describe the experimental setup, then detail trace collection and processing, and finally show how to recover the key.

## Experimental Setup

The physical setup consists of two main components: the target chip and a software-defined radio to collect the traces (initially we connect them over a coaxial cable and we then move to different kind of antennas at distances of up to 10 m in an anechoic chamber).

On the chip we run periodic computations of AES with a fixed key and random plaintexts. (The template attack requires a training phase where the key varies as well.) The AES implementation we use is *mbedTLS*, included in the Nordic Semiconductor SDK. We have also attacked the *TinyAES* implementation from the same SDK. Moreover, the chip is set to modulate and transmit random data according to the Bluetooth standard.

On the receiving side, we use a software-defined radio, USRP N210 by Ettus Research. The radio is tuned to 2.464 GHz, i.e., the frequency of the Bluetooth channel as per the Bluetooth standard increased by the clock frequency of the target device's CPU. The choice of frequency is essentially a consequence of how the leaked information from the CPU is modulated onto the output signal of the radio. We sample at 5MHz, this bandwidth is sufficient for the speed of our software under attack.

The result of running the first step of the attack in this setup is a record over time of the emitted signal's in-phase and quadrature components, spanning many AES computations of the chip's CPU.

## Trace Collection

Well-known side-channel techniques such as correlation or template attacks are based on aligned traces of the leaking signal, each covering a single execution of the computation under attack. Concretely, in order to apply such attacks to our signal, we need to partition it into individual traces, each spanning a single AES computation, and align the traces.

In a first step, we use a coarse-grained trigger mechanism[3] to recognize individual computations. By manual analysis we identified a frequency component in the signal of our target device that is only present just before AES runs. Therefore, a band-pass filter around the trigger component yields a rough trigger signal from the received emissions, which we square to amplify the trigger. We cut the original capture according to the trigger, obtaining traces that each correspond more or less to a single run of AES. For a successful attack we need precisely aligned traces, though, so the next step is to fine-tuned trace alignment.

To this end, we iteratively shift each trace in time, maximizing correlation with a "prototype trace". The prototype is the point-wise mean of all traces aligned so far. Intuitively, averaging aligned traces removes noise and irregularities due to different plaintexts used in the computation, so aligning new traces with the prototype becomes easier as the template itself becomes less noisy.

The result of partition and alignment is a set of precisely aligned traces, i.e., time-domain signals emitted by the target device at 2.464 GHz, each covering the time of a single AES computation. This dataset is suitable for known key-recovery techniques, such as correlation and template attacks.

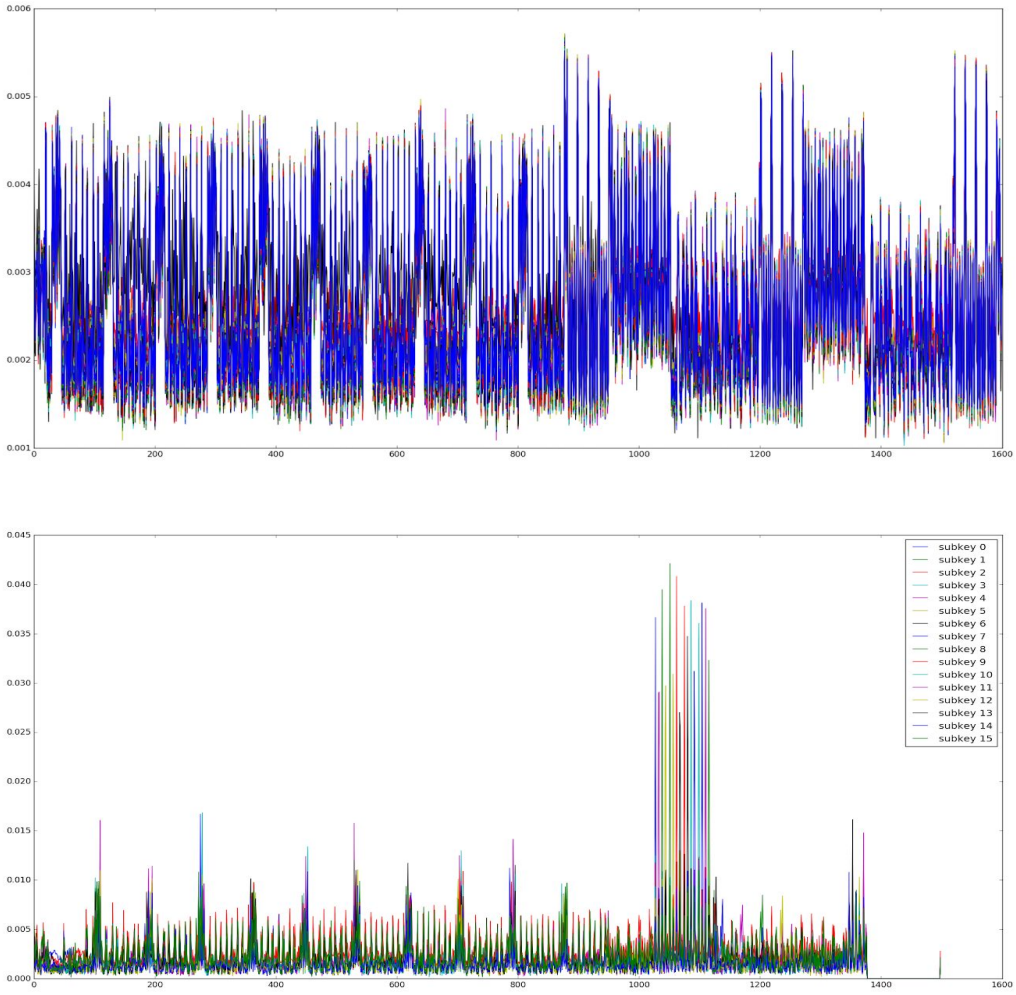
## Key Recovery

The final step of the attack is to use the collected traces and recover the AES key. At this point our data is sufficiently similar to the traces employed in power and EM attacks to use the same algorithms with only little modifications. It is the process of deriving the traces from mere radio signals that constitutes the novelty of our attack, not the key recovery algorithms that we apply afterwards. However, in order to demonstrate a full attack from start to finish we briefly discuss the application of well-known techniques for key recovery from our traces.

Our implementation is a slightly modified version of the attack code found in the ChipWhisperer [4] project, originally designed for analyzing power traces. We have successfully applied differential and template attacks to our traces, achieving full key recovery. Specifically, we attack the first round of the *SubBytes* step in AES. Currently, our best attack works in an anechoic chamber at a distance of 10 m. It requires 130k traces for the template, and 1.5k traces for the attack. Each trace is the average of 500 encryptions.

[3] We were inspired by https://github.com/bolek42/rsa-sdr, but this project was meant for conventional EM side-channels.
[4] https://newae.com/tools/chipwhisperer/

The following images show the aligned traces (first two rounds are visible) and the sum of absolute differences for the aligned traces, clearly highlighting a leak at the first round of AES for each byte of the key.




## Notes on other experiments

We obtained full key extraction also in office environments, but at closer distances. For example a correlation attack with 40k traces (each of which is the average of 500 encryptions) was successful against mbedTLS AES in a home environment.

Furthermore, we are investigating how to attack the hardware AES block in the chip (our template attack can currently recover 4 bytes out of 16 of the key).

Finally, we are working on adapting our attack for other Bluetooth and WiFi chips, which have likewise shown indications of leakage during preliminary testing.

# Discussion

In this section we discuss the results reported so far. In particular, we focus on the attack's applicability in real-world scenarios, on countermeasures, and on directions for future work.

## Real-World Applicability

The hardware requirements for carrying out radio attacks outside of lab environments are very modest: successful attacks from shorter distances are possible with just a commodity WiFi antenna and a hobbyist software-defined radio like the HackRF. Attacking from greater distances will require more expensive equipment, such as a high-quality directional antenna and a good SDR for trace collection.

Some knowledge of the target chip is required in order to determine the right attack parameters. In particular, the attacker needs to know or guess the clock frequency of the target's CPU to determine the radio frequency to listen on. We found in practice that due to the clock signal being modulated onto the radio the clock frequency can be guessed quite reliably from the spectrum of the target's radio emissions. Furthermore, the attack requires a trigger for cutting the signal into individual traces; in the case of our example target nRF52832 manual inspection of the signal yielded suitable trigger components.

Finally, the target device needs to use its radio in transmission mode while running the computation of interest. The destination of the transmission is irrelevant as long as the attacker can observe the signal. Since the actual data that is transmitted does not matter for the attack, any communication is fine. For targets that do not communicate enough on their own it is sufficient to periodically query for identifiers, beacons, echo replies or similar messages provided by the respective protocol stack.

## Countermeasures

Generic countermeasures against side-channel attacks have been an active field of research for some time. We refer to the relevant literature, in particular on *hiding* and *masking*. Hiding is the process of changing the design such that intermediate values of sensitive computations do not leak into observable channels, such as power, EM emissions and, as we have shown, radio signals. Masking tries to make leaked intermediate values less useful, for example by randomizing them. Both techniques will require consideration of the newly discovered radio side channel.

Another class of possible protection mechanisms are dedicated techniques to prevent information from leaking into radio signals. Since the general issue is a direct consequence of the physical proximity of analog and digital components in affected chips, countermeasures can only indirectly protect such systems:

- A simple approach is to just avoid sensitive computations in digital circuitry close to radio components. For example, a WiFi chip attached to a computer could use the PC's CPU for cryptographic operations instead of carrying them out internally. Naturally, such protections harm performance and require the availability of a separate processor in the first place. While leakage should be significantly reduced, there may still be some leakage.
- Barring the presence of an alternative processor, countermeasures have to ensure that the radio is never active in transmit mode during sensitive computations. For example, the firmware could serialize corresponding operations instead of executing them in parallel. In many cases this will require extensive redesign of the firmware and have a strong impact on performance.

New designs will be able to avoid the core issue by moving cryptography to protected hardware blocks or by incorporating strong shielding between digital and analog components. However, the required changes are likely to run counter to market demands: low cost and ever reduced chip size.

In any case it appears difficult to address the core problem without compromising on other requirements. Moreover, experience shows that protection mechanisms usually increase the difficulty of attacks but do not prevent them entirely. We therefore expect that radio side channel attacks will be possible for the foreseeable future and should thus be considered in the threat model of sensitive applications.

## Impact

We believe that the impact of this work is significant, as EM or power side-channels are usually considered out of scope for most devices which do not have any tamper resistance, such as IoT devices, wearables, Bluetooth and WiFi chipsets (in smartphones and computers).[5] The rationale for ignoring side channels in these devices is that if an attacker can come that close, then the system can already be compromised in many ways. Additionally, an eavesdropper in the vicinity should be foiled by the cryptography used by the chip. However, we show that this security model is not sufficient for some wireless devices, and that for the data to be really protected from attackers, the chip should not leak through the radio channel. As a consequence we believe such device require side channel resistance for power side channels, because radio side channels are correlated to the power consumption.

[5] As commented for example on this post by Paul Bakker, one of the main developper of mbedTLS (former CEO of Offspark the company which was developping polarSSL which became mbedTLS) https://tls.mbed.org/discussions/crypto-and-ssl/aes-implementation-resistant-to-side-channel-analysis-attacks

# Additional Resources

- Full version of the Screaming Channels paper:
  http://s3.eurecom.fr/docs/ccs18_camurati.pdf

- Project webpage:
  http://s3.eurecom.fr/tools/screaming_channels/

- Tool for replication of our results:
  https://github.com/eurecom-s3/screaming_channels