Stealth Software Trojan: Amplifying Hidden RF Side-Channels with Ultra High SNR and Data-Rate

Itamar Levy * Gal Cohen ¶
*Engineering Faculty, Computer Engineering, Bar-Ilan University, Ramat-Gan Israel.
Email: itamar.levi@biu.ac.il
¶Engineering Faculty, Computer Engineering, Bar-Ilan University, Ramat-Gan Israel.
Email: coheng15@biu.ac.il

***Abstract*—Interconnected devices enhance daily life but introduce security vulnerabilities, new technologies enable malicious activities such as information theft. This article combines radio frequency (RF) side-channel attacks with software Trojans to create a hard-to-detect, stealthy method for extracting kilobytes of secret information per millisecond over record distances with a single measurement in the RF spectrum. The technique exploits Trojan-induced electrical disturbances in RF components originating from peripherals, buses, memories and CPUs to achieve high SNR data leakage schemes. Experimental results show negligible acquisition time and stealth. The research introduces optimized modulation, demodulation schemes, and specialized synchronization symbols to minimize error rates and maximize data rates. It highlights the need for advanced detection and defense mechanisms to ensure the security and privacy of interconnected devices.**

***Index Terms*—Modulation, Remote Attacks, Radio Frequency, RF, Side Channel Analysis, SCA, Single Trace, Software, Spectrum, Trojans.**

I. Introduction

A persistent and evolving threat to the integrity and security of modern computing systems are malicious code injection attacks and software Trojans. These sophisticated malware infections are typically designed to infiltrate targeted systems, often remaining undetected while compromising sensitive data, disrupting operations, or creating backdoors for later exploitation. Combined with side-channel analysis as an extraction mechanism, such malware or memory-storage Trojans exploiting this threat vector are quite dangerous.

Seminal work in the field, such as [1], [2], provided an early framework for understanding code injection techniques and malware behavior. Subsequent research has delved deeper into specific attack vectors, including buffer overflows and their exploitation mechanisms [3], return-oriented programming, flush-reload and its exploitation through traditional side channels [4], [5]. In addition, the increasing use of rootkit malware for stealth and persistence attacks in combination with side channels [6], [7].

Alongside academic research, numerous industry reports and threat analyses have underscored the real-world impact of code-injection attacks and software Trojans. This paper highlights the need for continued investigation into these threats within the specific context of Radio-Frequency Side-Channel Analysis (RF-SCA).

Important advances in the recent years have demonstrated that secret internal data can be modulated into the RF channel [8]–[10]. Our work make use of such observations, and combined with sophisticated code injections, we demonstrate a *single-trace* (measurement), ultra-high SNR extraction mechanism including specialized encoding and decoding for such tailored malicious codes that are *optimized for such channels* and can leak information over very long distances with superior error resilience and high SCA data rates (DR).

The primary objective of this article is to investigate, optimize, and analyze methodologies for actively modulating and controlling information leakage through RF side channels using code injections, with the objective of exploiting (but not limited to) Internet of Things (IoT) devices featuring radio transmitter components. We demonstrate that, using embedded components, entire memory chunks containing kilobytes of data (e.g., images) can be leaked within milliseconds from about 25 meters away (i.e., stealthily and difficult to detect). These models, which access internal buffers in fingerprint sensors, facial recognition cameras, or algorithmically computed values stored in various memory-mapped addresses, can easily leak information in a single trace, demonstrating remarkable exploitation characteristics in the SCA context (e.g., DR, SNR), and significantly advancing the state of the art in methodology, experimentation, and knowledge.

Our approach involves identifying specific electrical perturbations, code structures and peripheral activities that leave distinct signatures in the RF spectrum. By exploiting these perturbations, we successfully generate a tailored modulation by perturbing these signals, introducing harmonics into the RF spectral map around the main RF component peak.The proposed encoding and decoding mechanisms use different constellations (FSK, Q-FSK 16-FSK etc.), which are enhanced with special synchronization methods and equipped with sophisticated internal data alignment mechanisms. We illustrate powerful and sophisticated demodulation and decoding schemes including various filters tailored to the RF channel and the Trojan modulation. Finally, we also touch on theoretical aspects such as minimizing error rates and maximizing data rates with a packed model. All of this enhances the understanding of RF-SCA, which enables the development of either: protection against such intelligent extraction schemes (covert data transmission), or on the contrary, the deliberate

use of such a scheme by the user and not covertly [11].

II. Background and Contribution

Side-channel attacks (SCAs) exploit non-deliberate electrical interference's or imperfections that can collectively generalize to significant (secret) information leakage from a cryptographic system. Generally speaking, these attacks do not target cryptographic algorithms mathematical hardness, but rather exploit weaknesses in their implementation or physical environment. By analyzing the side-channel information, adversaries can deduce sensitive data. However, typically this added information is outside the cryptographically assumed model (e.g., Black-box), so there is no shock in breaking the cryptographic algorithm's security guarantees... There is a flourishing literature and provided schemes that protect against SCAs at various levels. Nevertheless, in this paper we discuss parts of the system that are not conventionally protected, nor can all be processed in protected environments (and it would not always prevent the leakage), emphasizing the importance of diving deeper into this field.

A computer-system Trojan is a type of malicious software or hardware that disguises itself as a legitimate program or file to deceive users into installing it. Once activated, a Trojan can perform a variety of harmful actions without the user's knowledge, such as stealing sensitive information, creating backdoors for unauthorized access, or damaging system files, more related literature in this field can be found in [12]–[14]. RF (Radio Frequency) Side-Channel Attacks (SCA) exploit unintentional emissions of electromagnetic signals from electronic devices to extract sensitive information, progress in this field from [8], [13], [15], trying to achieve reduce number of traces but none utilized Trojans and modulation aspects.

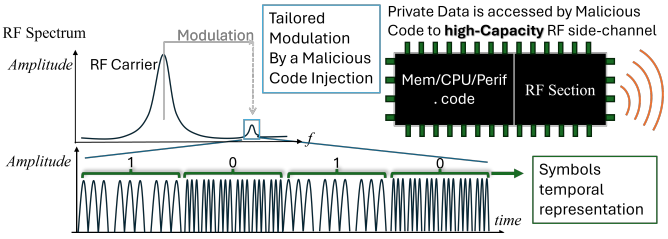


Fig. 1. Illustration of a the simplest FSK modulation technique utilized in this paper (without advanced features).

A. Related-Work

In our work, we focused on far-field electromagnetic side-channel emissions *single trace* SCA, the traditional SCA paradigm (utilizing statistics) to extract information from multiple SCA observations [8], [9], [15], [16]. This property may limit the extent of such results (on which cryptographic primitives operate), but it also introduces many other challenges, such as complicated synchronization of this plurality of independent leakages and far increased noise. I.e., the *single trace* paradigm targeted in this paper is of valuable from several perspectives (Tab. I shows the different amount of measurements used). From a different angle, significant research has investigated hardware Trojan attacks on RF equipment in electronics to extract bits of information from devices. Notable examples include in [13], [14]. Our work differs from these in terms of Trojan structure (software vs. hardware) and the amount of data being compromised, its rate and physical characteristics such as the possible capture distance. Fig. 2 illustrates the different structural properties of these two distinct Trojan designs. It highlights how the processor and the peripherals generate electrical disturbances, which can subsequently leak into the RF module, moreover it present the software Trojan advantages over the hardware Trojan.Additionally, the related work surveyed in [11], [12] explores similar electromagnetic phenomena to transmit information from electronic devices. However, the context, depth, and optimization of these studies differ from our works, both studies did not create controlled interferences and did not develop specialized modulation and demodulation schemes tailored to the presented electrical phenomena reaching poor results in terms of SNR and data rate as illustrated in Tab. I (10-15 vs 100k b/s). Furthermore the context of these articles is very different: [11] uses the electrical phenomena in order to spare RF electronics and not in an attack scenario, while Feng et-al [12] presents this vulnerability on a desktop and not via a small chip, i.e., a different scale in terms of consumption and electromagnetic footprint.

TABLE I
Related papers results. #Meas., Dist. and Inst. denotes # of measurements, distance and instructions, respectively

Article	#Meas.	Dist.[m]	DR [b/s]	Trojan?	Size
SCA [8]	50000	10	NA	No	NA
SCA [15]	10000	15	NA	No	NA
SCA [16]	341	15	NA	No	NA
[13]	1	NA	10	Yes	$9.56\mu m^2$
[12]	1	200	up to 15	Yes	20+ lines.
[9]	1	few cm	NA	Yes	15 lines.
Our model	1	**25**	**100k**	Yes	10 lines.

Our contribution represents the first successful integration of these diverse ideas, resulting in a robust side-channel Trojan attack that can compromise significantly (orders of magnitude) more information than other schemes discussed in this section, with stealth and ultra-high SNR from a *single trace*. We present a comprehensive end-to-end system that can encode, modulate, filter, demodulate, and extract any kind of information from (e.g.) the device's memory. This includes an optimized and flexible injection model, a transmission model, a custom modulation scheme tailored to our scenario, a demodulation model, and a signal processing scheme designed for minimal error rates and high signal-to-noise ratio (SNR). Crucially, our injection method remains both covert and effective throughout the operation, leaving no physical fingerprint on the electronic device and requiring no intervention in the manufacturing process, unlike previous hardware Trojan models. Furthermore, our model is easier to demodulate and analyze compared to other SCAs, reducing

capture time from hours or days to mere milliseconds.

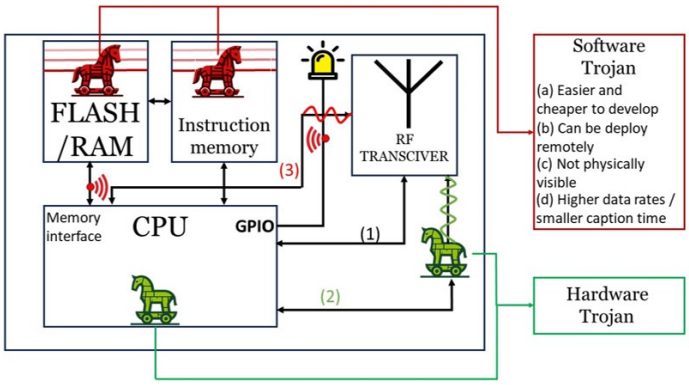


Fig. 2. Visual illustration of various Trojans employed to extract information from an embedded device.

B. Threat model

This threat model investigates the security risks associated with the combination of radio frequency (RF) side-channel attacks and software Trojans. It focuses on IoT devices with radio transmitters, such as fingerprint sensors, facial recognition systems, and medical devices. These embedded systems are particularly vulnerable to advanced code injection techniques [17]. While there are active detection tools, such as antivirus software and secure boot systems [18], [19], these methods are often energy inefficient, require dedicated hardware [20], and can sometimes be bypassed. For example, in some scenarios secure boot systems that rely on software signatures during manufacturing may be compromised if an insider injects malicious code. Key assets at risk include internal memory data (such as Corneal or fingerprints photos, secure keys, private medical information). Potential attackers may seek to covertly exploit sensitive data over long distances using RF signals with unauthorized and undetectable access to sensitive data (35 [dB] difference between leaked data, around random noise spurs, and original RF transmission amplitude, see Fig. 3). The risk is amplified by current standards and regulations, such as MIL-STD-461, EN-55022, FCC guidelines, and CISPR standards, which do not adequately prevent such information leaks. The impact of successful attacks is severe, potentially resulting in significant data breaches and loss of sensitive information.

III. Metrics, Evaluation tools and Measurement Setup

This section offers a concise review of the key metrics and evaluation tools utilized in the analysis.

A. Metrics and Evaluation Tools

Using Photos as a Transmission Media: In this study, photographs were used as a versatile medium for evaluating transmission quality because of their multiple advantages. First, photographs are a ubiquitous form of data, often stored in the memory of many electronic devices. Second, they often encapsulate sensitive or private information, such as biometric identifiers like fingerprints or corneal images. Third, photographs lend themselves well to a variety of evaluation tools, including statistical distance metrics of histograms, which facilitate the assessment of error rates and data loss. Finally, and most importantly, photographs provide a uniquely intuitive visualization of data errors and data loss, a critical attribute given the inherently noisy nature of the transmission medium under investigation and the synchronization challenges inherent in remote RF channels.

Probability Distance Measures: One of the main questions we are trying to address in the analysis below is the differences between post-demodulation and decoded leakage sequences that reflect memory contents. The leakage distributions of the original versus the attacked data should be evaluated across distances, system configurations, and modulation parameters to assess error rates, data loss, and related metrics. Therefore, the most natural tool to use for evaluation is a probability-distance measure. The relative entropy or the Kullback–Leibler (KL) divergence [21] is one measure for the probability distance between two distributions, though it is not a metric and is not symmetric:

$$D(\mathrm{X}\|\mathrm{Y}) = \sum_{z} \Pr(\mathrm{X}=z)\log_2\left(\frac{\Pr(X=z)}{\Pr(Y=z)}\right)$$

where, X and Y represent two PDFs in question over the shared support ($z \in Z$). The Jensen–Shannon divergence (JSD) is a smoothed and symmetric transform of the KL divergence: $D_{JS}(\mathrm{X}\|\mathrm{Y}) = D(\mathrm{X}\|\mathrm{M}) + D(\mathrm{Y}\|\mathrm{M})$, where, M is the arithmetic mean of X and Y, M=0.5(X+Y). As the measure is weighted with probabilities it fits better for our discussion than distance measures like the norm-1 distance (or total variation distance) Note that, probability distance and more generally, Statistical Distance are actually tightly related to the MI ,$\mathrm{MI}(Y;L) = D(\Pr(y,l)\|\Pr(y)\Pr(l))$. In this paper we use the divergence and the visual representations of the PDFs of the memory stored data as a visual proof of concept and a tool for demonstration.

a) Signal-to-Noise Ratio (SNR): SNR is a key metric in telecommunications and signal processing that quantifies the ratio of signal strength to background noise, especially over RF channels. Expressed in decibels (dB), it is calculated as:

$$\text{SNR (dB)} = 10\log_{10}\left(\frac{\text{Signal Power}}{\text{Noise Power}}\right)$$

A higher SNR indicates a stronger signal relative to noise, resulting in clearer and more reliable communication or data transmission. In our applications, maintaining a sufficient SNR is essential for achieving accurate data transmission and system performance, while utilizing only a *single measurement capture/trace*. As can be seen in Fig. 3 which illustrates a (clean) modulation example from our scheme, we obtain a clear 22 dB signal above the noise-floor, allowing us to extract information from the device with high capacity. Although the side channel emission is apparent, the SNR difference between the primary Bluetooth lobe and our modulation peak, which

appears next to our Bluetooth peak (up to 100 MHz), is a substantial -35 dB lower than the Bluetooth signal. This significant disparity in power implies that detecting the side-channel emission requires specialized equipment and precise knowledge of its location within the RF spectrum, as it is easily indistinguishable among other spurs in the system. These facts underscore the stealthy nature of injection.

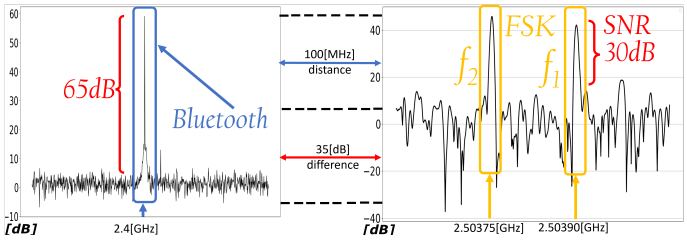


Fig. 3. Illustration of the SNR differences between the primary Bluetooth signal and the side channel transmission.

B. Measurement Environment

a) Measurement Process: To simplify the measurement process only, we have synchronized the recorder with the injection transmission and automated the data acquisition. I.e., it has no impact on the success of our attack or the significance of the results, it only allows to perform consecutive, fast and efficient experiments. While we have opted for automation, *it is not necessary*. Furthermore, it is possible to develop a real-time demodulation system based on our model that does not require synchronization or automation. It's worth noting that we initially worked without such complexity, and manually demodulated the data without using either of these approaches. Our measurement setup involves several interconnected processes. A central Python process initiates two parallel sub-processes: the first sub-process is a C application responsible for configuring the Software Defined Radio (SDR), setting parameters such as bandwidth, sample rate, center frequency, and recording time while starting the recording. Simultaneously, the second sub-process triggers an injection via UART transmission to the infected device. The transmission contains activation parameters (e.g., number of electrical disturbances (ED) per symbol, synchronization method, etc.). Once the capture, including the injected transmission, is complete, the Python code proceeds with signal demodulation, decoding and data reconstruction.

b) Measurement equipment: For all of the experiments described in this article, our test bench consisted of several essential components: **PC**: Handled the demodulation of the recorded signal. **Tektronix RSA306B Spectrum Analyzer**: Used to record the RF spectrum. **PCA10040 Development Board**: Simulates an RF device equipped with a processor, RF module, and electronic peripherals such as GPIOs, SPI, and UART.

There were (mainly) two experimental setups used: **High Data Rates Measurement Setup-** In these experiments, a small whip antenna was placed directly over the Bluetooth emitting component to maximize signal amplification. **Noise Distance Measurement Setup-** for long range measurements, we used a TL-ANT2424MD dish antenna. This setup required two computers for synchronization between the recorder and the transmitter. Instead of the original direct UART connection, we used a long Ethernet cable to connect another computer. The second computer transferred the parameters and activated the injection, through the serial port, for rapid automation. This setup provides comparable signal-to-noise ratio (SNR) results to the high data rate setup at a distance of 5 meters. Both configurations are depicted in Fig.17.

IV. Transmission System

This section discusses how electrical interference leaking into the RF module can be exploited to create modulations that compromise (e.g.,) memory and steal data from a device. As explained in the related work section, we did not originate the concept that peripherals and CPUs create an electrical footprint in the RF spectrum. In this article, we refine this notion, achieving significant results. As demonstrated in [9], different peripherals generate distinct RF signatures. To identify the most effective peripheral disturbance, we measured several of them, as shown in Fig 4, which illustrates different electrical disturbances (or ED in the time domain) that can be used in order to create modulations. The different curves in the figure show interferences of the RAM access, the GPIO-register set function and the UART transmission footprint. We have chosen to work with memory writing only as a demonstration as it provides nice results in terms of amplitude and signal bandwidth, but the research is not limited to this RF electrical disturbance mechanism.

It is important to note that a *transmission system capable of exploiting sensitive data could be created from any electrical leakage into the RF spectrum*. Therefore, when building any embedded device, we must be careful and isolate the RF module from other electrical disturbances in the system to prevent such information leakage. In practice this is quite a complex task, standards and regulations are not designed to prevent such information leakage and devices that pass (e.g.). MIL-STD-461, EN-55022 standard, Federal Communications Commission (FCC) and CISPR standards [22] may still be vulnerable for example Fig. 3 shows the SNR of our modulated leakage can still pass the FCC and CISPR reliability and electromagnetic interference (EMI) requirements.

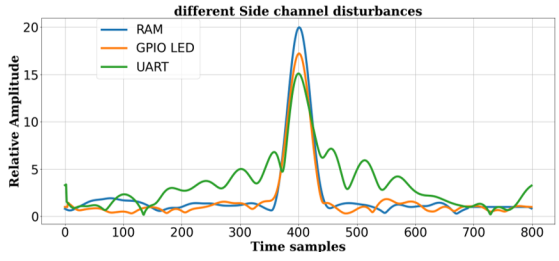


Fig. 4. Abstraction of various Trojans which may be employed to extract information from an embedded device.

A. Symbol Structure

The construction of a symbol with the proposed modulation (as demonstrated in Fig. 5) relies on several key parameters:

1) **Temporal duration of the Electrical Disturbance (ED):** represents the temporal magnitude of the electrical disturbances used to create periodic patterns. These patterns translate into a side-channel presence around our transmitter's main peak in the RF spectrum.
2) **Electrical disturbance loop structure (LS):** temporal extent of the loop code structure,which contains a some additional processor instructions that facilitate the cyclic behavior of the electrical disturbances.
3) **Number of electrical disturbances (N):** number of electrical disturbances constituting each period within the symbol. This parameter significantly influences the model's disturbances frequency as can be observed from (Eq. 2) .
4) **Number of transmission Blocks (NB)**: number of periodic disorders transitioned per symbol, exerting a substantial impact on symbol duration within the model. Notably, its significance lies in its flexibility as a modifiable variable, devoid of any influence on the underlying frequency. This parameter is the most significant factor in determining the data-rate, as the mathematical model developed in Section VI (Eq. 4) reveals.

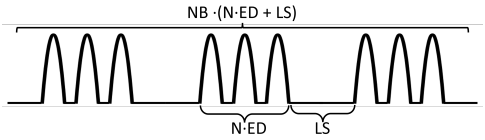


Fig. 5. Illustration of a symbol temporal parameters.

Symbol's relative frequency equation:

$$F_{Hz}(N) = \frac{1}{N * ED + LS} \tag{2}$$

B. Symbol Encoding

For symbol encoding, we select a subset from the set of all frequencies F_{Hz}. The size of the subset depends on the modulation we want to use (FSK, QFSK,...). We will denote this subset as $F_g = \{F_{\text{Hz}}(N_1), F_{\text{Hz}}(N_2) \ldots, F_{\text{Hz}}(N_n)\}$ when $N \in \mathbb{N}$, for simplicity and ease of notation we denote this subgroup by $\{f_1 \ldots f_N\}$, respectively. Assuming that $NB > 1$, and $|F_g| = 2^m$ = n, where m represents the modulation factor or the number of bits encoded into each symbol. Subsequently, each frequency within the subset F_g is assigned a numerical identifier. Conventionally, the highest frequency, often denoted as f_1, corresponds to the value 0, while the lowest frequency, represented by f_n, is assigned the value $2^m - 1$.

C. Symbol Duration balancing

One of the fundamental requirements for our modulated signal is to maintain a consistent temporal duration *per symbol*. This is crucial for ensuring the synchronization of our sampler and thus preventing any loss of synchronization. However, as discussed in the preceding section, the temporal duration of our symbols depends on various computational and architectural parameters. For example, factors such as branch prediction, speculative execution, and varying cycles for memory fetches, as well as software or hardware code payloads and constants used for initialization in our gadget, are not always deterministic in duration. These small perturbations are challenging to predict and balance. However, gross differences in symbol duration can be easily maintained by selecting a frequency set in which all element periods nicely divide the symbol duration. Achieving a state where all symbols have exact same lengths becomes impractical, and it would be limiting trying to achieve so. However, to eliminate such gross differences and achieve a balanced set of symbols, we select N_{B1} and N_1 for our initial symbol, and then compute the NB of the remaining symbols as follows ($\forall\ N > N_1$):

$$N_B(N) = \text{round}\left(N_{B1} \cdot \frac{LS + N_1 \cdot ED}{LS + N \cdot ED}\right) \tag{3}$$

This normalization adjusts all symbol lengths relative to the highest frequency symbol, as shown in Fig. 6.

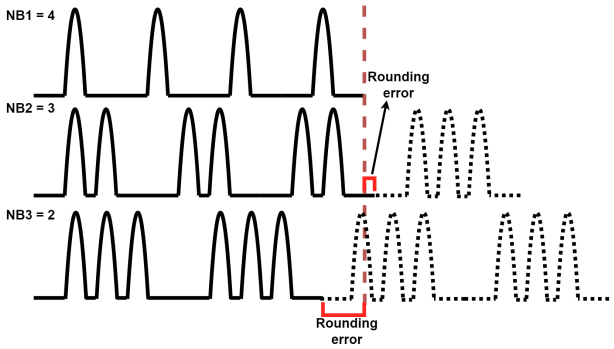


Fig. 6. illustration of symbol alignment process.

This approach is not perfect, because as the temporal magnitude of the symbols decreases (in an attempt to achieve higher data rates), the number of blocks (NB) decreases, making it more difficult to correct the rounding error (this effect is represented in Eq. 4).

$$\text{Max. Symbol Deviation } (\%) = 0.5 \cdot \frac{LS + N \cdot ED}{NB_1 \cdot (LS + N \cdot ED)} \tag{4}$$

D. Synchronisation

One of the most challenging problems we faced in accurately demodulating the signal was synchronization. As discussed in section 7, the symbols temporal duration is dependent on computational and architectural parameters, adjusting NB per symbol roughly compensates for this effect, though it is not hermetic. For signal synchronization, our initial approach was to detect the beginning of the transmission and attempt to synchronize with the first signal rise. However, synchronization loss occurred rapidly in this scenario, resulting in high error rates. To address this, we adopted three primary methods for signal synchronization.

a) Differential Synchronisation: stands out as the most efficient method of synchronization, as it does not necessitate any additional symbols. In this method, we identify changes in frequency that arise between symbols. Once we located the changes we determine the number of symbols with the same frequency transmitted within two changes of frequency, we do it by dividing the distance between them by the average symbol length thus determining how many symbols of the same frequency was transmitted between two distinct frequencies, this method maintain good performance when the signal is not constant and contain a lot of information or entropy, although it is not stable and can lose synchronization when there are long sequences of the same symbol, we used it only to acquire small packets.

b) Symbol Synchronization: In *symbol synchronization*, the $f_1, \ldots, f_n$ frequencies used for transmission are shifted by one, with f_1 dedicated to synchronization. A synchronization distance (**SD**) is defined, and for every SD data symbols transmitted, one synchronization symbol is sent. As shown in Fig. 7, f_1, which no longer carries data and is a bit lower than the rest of the frequencies, indicating a synchronization point for the sampler. In this method, f_2 becomes the new zero value. This approach increases the transmission duration, requires more bandwidth, and introduces an additional symbol for each synchronization distance, resulting in slower transmission speeds. However, it provides improved stability and re-synchronization capability in synchronization loss event.

c) UART Synchronisation: In *UART-like synchronization*, we utilize the structured communication protocol of UART. However, rather than transmitting individual bits, symbols are transmitted using f_1 as the frequency for the start symbol, and the subsequent highest frequency for the stop symbol. Furthermore, we adjust the quantity of data transmitted between the start and stop bits to correspond with the synchronization distance (SD). It is important to note that in this synchronization method, the frequencies used remain unchanged, although the synchronization requires the transmission of two symbols for each synchronization distance, which slows down the transmission speed. Fig. 7 illustrates the synchronization structure, the start and stop synchronization symbols are part of the transmission set F_g, unlike symbol synchronization.

For the remainder of this paper, we will focus on symbol and UART synchronization as they provide optimal results in terms of system stability and data rate. However, all synchronization methods (symbol, UART, differential) are available in our repository GitHub link,, accompanied by a simple user interface for adjusting basic system parameters such as NB (for different data rates), synch.-method, synch.-distance, and modulation factor (m), which may be useful for future research.

V. Injection Modulation and Field Experiments

Before delving into the demodulation process, decoding mechanisms, and mathematical optimization, we assess the practical applicability of our attack in real-world conditions.

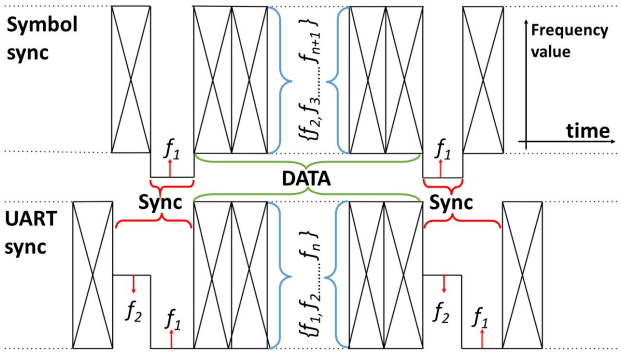


Fig. 7. The structure of the two distinct synchronisation methods used

We evaluate error rates and signal-to-noise ratios (SNR) at different distances to assess performance across various ranges. To emphasize the attack's effectiveness and simplicity, we used only a dish antenna, a software-defined radio (SDR), and a standard PC, avoiding amplifiers or specialized equipment.The experiments used QFSK modulation at 6.25 KB/s and symbol synchronization, which, although below the maximum capability of the system, ensured reliable, error-free data acquisition in a laboratory environment. Another significant aspect of this experiment was our deliberate decision not to operate in the sterile environment typical of other experiments in this article (Sec. VI-D), and in the field. Instead, we conducted our work in ordinary university rooms and noisy LABs corridors, naturally exposing our setup to ambient noise, including Bluetooth frequencies emitting from student headphones, private cellphones, and other Bluetooth emitting devices, and EM laboratory interference. (Experiments conducted with the same setup in a 25-meter open field, free of electromagnetic noise, yielded clear results).

A. Field experiments

In the initial experimental, we examined the susceptibility of a device to eavesdropping in a long Corridor, as shown in Fig. 8 our measurement setup was positioned in one side of the corridor, and on the other side we placed our emitting embedded device, listening to the injection transmitting images over the side channel. We measured both the Signal-to-Noise Ratio (SNR) and the statistical distance between the histograms of the reconstructed and original photos. For each measurement, we moved the transmitting device 2.5 meters away from the receiving antenna until the transmitted photo became unrecognizable. Fig 8 shows two intersecting lines: one for Signal-to-Noise Ratio (SNR) and the other for statistical distance (or KL divergence, see Sec. III-A), plotted against the distance between the transmitting and receiving antennas. As the distance increases, the SNR decreases and the statistical distance between the original and reconstructed photos increases. The figure visually presents reconstructed photos at each step, emphasizing this effect. the experiment highlights the model's ability to extract information from devices in noisy environments and over long distances (17.5 meters), showcasing the practicality of such attacks.

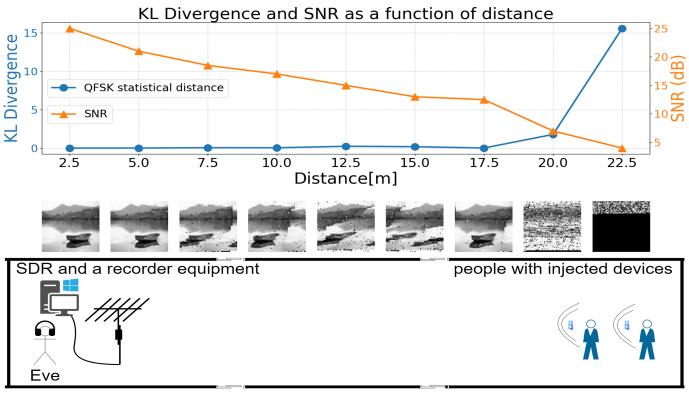


Fig. 8. A diagram illustrating the SNR and the statistical distance as a function of distance in a noisy corridor.

In another experiment, we aimed to collect information from an adjacent room by recording side-channel emissions through a wall. Our objective was to extract data from Bluetooth devices located in the neighboring space, as illustrated in Fig. 9 , Similar to our previous experiments, we measured the statistical distance and the signal-to-noise ratio (SNR) at varying distances, The wall absorbed part of the transmission, resulting in SNR values that were 5 dB lower compared to the previous experiment. Despite this attenuation, the transmission was still effective, Fig. 9 shows clear photos captured even from a distance of 10 meters. This suggests that an adversary doesn't need to be in the same room or even on the same floor to steal information, facilitating a more covert process.

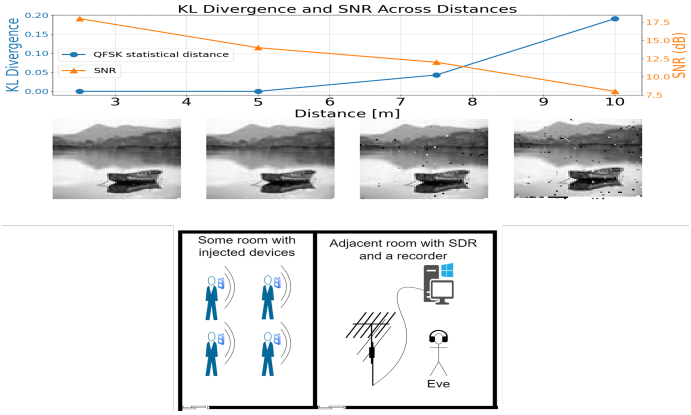


Fig. 9. illustration of the SNR and statistical distance as a function of distance through a wall in a separate room.

B. Injection structure

In this article, we designed various code injections with differing levels of complexity to illustrate the trade-off between a robust, easily demodulated injection with synchronization and activation schemes, and a smaller, more compact and simplistic injection mechanism. The next section explains the different code layers represented in Fig. 10.

Injection base: This model employs a fundamental injection scheme to demonstrate information leakage from the device

Algorithm 1 Most Basic Injection Model

```
1:  function BasicTransmission(A)
2:      Constants:
3:      TML ← MemoryLocation   ▷ Targeted memory start location
4:      TMS ← SectionLength    ▷ Targeted memory size
5:      n_f1 ← SystemConst     ▷ Modulation frequencies
6:      NB ← SystemConst       ▷ Symbol temporal magnitude
7:      for i ← 0 to TMS − 1 do
8:          for j ← 0 to ByteSizeInBits − 1 do
9:              N ← value(TML + i)&(1 << j) + N_1   ▷ Frequency for the next bit
10:             for k ← 0 to NB − 1 do
11:                 for m ← 0 to N − 1 do
12:                     ElectricalDisturbance()   ▷ Generate peripheral disturbance
13:                 end for
14:             end for
15:         end for
16:     end for
17: end function
```

(see Alg. 1). The leakage mechanism is constructed using four nested loops, each serving a distinct purpose. The first loop iterates over memory bytes, while the second loop processes the memory bits. Subsequently, the transmission frequency is derived from the bit value. Lines 10,11,12 outline the symbol construction process. The first line specifies the number of blocks (NB), and the second line details the frequency and the number of electrical Disturbances per period. The Electrical Disturbance(·) function generates disturbances, which are created by a peripheral device and subtly leak to the RF-FE further amplifying it.

Symbol alignment: For the symbol alignment process explained in Sec. IV-C, we employed a lookup table to map the current symbol to the normalized number of blocks (NB). Incorporating this lookup table into the injection scheme slightly increases the memory cost, which depends on the number of different symbols used in the modulation (e.g., FSK with 2 symbols, QFSK with 4 symbols, etc.).

Synchronization: Incorporating synchronization requires the transmission to pause for 1 or 2 symbols at every synchronization interval (d), as detailed in Section IV-D. This modification alters the structure of the injection process, necessitating the tracking of both synchronization bits and data bits. Consequently, this slightly increases the complexity of the injection, adding additional lines of code and increasing memory usage.

Initial Transmission: A stream of bits with a constant signature and transmission parameters that is transmitted before the actual data stream. By reading this signature, one can synchronize to the upcoming data stream. In our model, this initial transmission is also used to convey information about the transmission *for automation purposes only*, such as the data size (to determine where to stop processing) and the number of transmission blocks (to evaluate symbol sizes

during extensive experiments), although this is not necessary in a scenario where the transmission parameters are constant.

Packets construction: This is a crucial concept that helps reorganize the code into two distinct parts. The first part involves calculating an array of symbols, which is then added to the initial transmission. The second part consists of the modified transmission function, which, unlike before, no longer has to calculate the next symbol. This reorganization has several significant advantages: first, the transmission function itself no longer includes any calculations, synchronization and the transformation from bits to symbols are already integrated into the provided symbol array. This results in a transmission function that is much faster, has a smaller spectral footprint, and requires less acquisition time by the demodulator; second, minimizing calculations within the transmission function results in a less noisy RF transmission that is easier to demodulate. Finally, breaking the data into smaller chunks that can be transmitted independently prepares the ground for an intelligent activation scheme. *For instance, capable of detecting CPU loads and transmitting small packets during periods of low CPU usage*. In addition, avoiding processor overloading greatly reduces the complexity of detection.

Smart activation: In the context of code injection, it is essential to implement a sophisticated activation scheme in order to evade detection. In this study, our approach focused on activating the code via a UART command, without delving deeply into more advanced activation methods (which may even be remote and passive). However, future iterations could leverage our model to initiate transmissions during periods of low processor load or through commands sent via other peripherals, such as a Bluetooth receiver. This approach enhances stealth and operational efficiency, ensuring the injected code remains undetected and effectively operational.

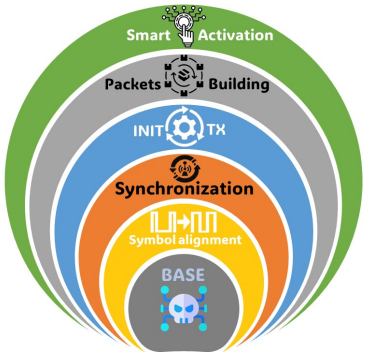


Fig. 10. Top down illustration of the different code layers.

C. Demodulation Process

Here we delve into the demodulation process using photos as our primary example, although the channel can carry any type of information. Our objective is to extract the signal from the side channel and reconstruct the original image from the sampled data using a systematic approach. In our initial demodulation scheme, we adopted a simple yet effective strategy. We partitioned the sampled data points into windows and applied the Fast Fourier Transform (FFT) to each window. From these transformed data sets, we identified the dominant symbol within each window. Next, synchronization is found, and the image is reconstructed, as shown in Fig.11. However, as we delved into higher transmission frequencies during our experiments, challenges began to emerge: First, as bit rates increase (reducing NB_1), the number of samples per symbol decreases. This reduction in samples results in decreased frequency resolution and lower SNR, complicating the differentiation of transmission symbols in the frequency domain. Moreover, another complicating factor to our ability to demodulate the signal, is that as (NB_1) decreases, it becomes increasingly challenging to standardize symbol sizes, demanding more effort to maintain synchronization of our sampler (as can be deduced for Eq. 3), all filters used are visually represented in Fig. 11.

Bandpass Filter: the bandpass filter helps us tune to on a small part of the spectrum where the change from one symbol to the next is most noticeable.

Frequencies response alignment $H^{-1}(F)$: Symbols of different frequencies have different responses in the frequency domain, which affects our ability to discriminate dominant symbol frequencies within a given window, especially if our window sees two different frequencies with different time spaces. This leads to uneven symbol durations, becuse symbols with higher frequency responses may be favoured over those with a more dominant time representation in the window. To mitigate these effects, we identify the frequency response H and apply its inverse, H^-1, to the FFT result. This alignment procedure ensures that all frequency responses are aligned, thereby promoting clearer classification.

Noise level threshold filter: This filter helps us differentiate between periods of active transmission and periods of no transmission, such as the idle time between packets when the CPU is constructing new packets but the channel is idle. We achieve this by filtering out data points lacking a strong frequency response in the measured bandwidth, as illustrated by the time domain illustration (blue curve) Fig. 11.

Gradient filter: This filter proves useful in scenarios where noise levels are high, and the conventional filter fails to adequately distinguish non-transmission areas. While it may not be effective for all data types, it particularly shines in image processing tasks. In images, consecutive pixels typically exhibit lower gradients compared to random noise pixels. By calculating the average gradient across an area of pixels, we can effectively identify regions containing information.

VI. Enhancing Data Throughput and Minimizing Error Rate

In the next section, we delve into the design's mathematical model, optimizing and predicting ideal system configurations.

A. Data Rate Equation

Several methods are used to optimise data throughput. The first method involves selecting N_1 to maximize the symbol

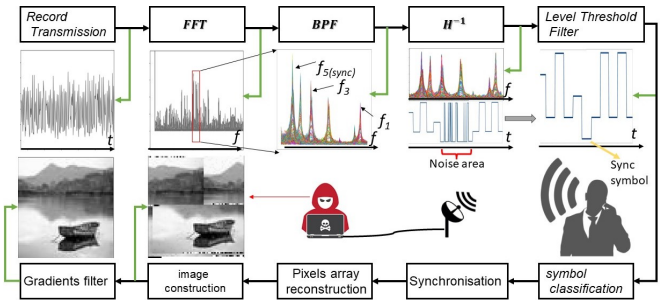


Fig. 11. Sequenced illustration of the demodulation process.

output frequency. In an ideal scenario, N_1 would be set to 1 to minimize electrical disturbances per symbol (N) and consequently reduce the transmission block duration. However, our choice is constrained as *higher symbol frequencies necessitate higher SDR sampling rate*. Another consideration is the number of transmission blocks, a lower NB_1 results in shorter symbol duration but sacrifices statistical data. Additional factor is the number of data bits encoded in each symbol (m), which determines the modulation used, and finally synchronisation can introduce additional symbols into the data stream, thereby reducing the effective data rate (as discussed in subsequent sections).

We can sum all the bit rates parameters in the next equation:

$$BR = \frac{SD}{SD + AS} \cdot \frac{1}{NB_1 \cdot (N_1 \cdot ED + LS)} \cdot m \tag{4}$$

SD , AS - synchronisation distance, synchronisation symbols

B. Available Frequencies Distribution

As described in latter chapters, in practice the model cannot choose the frequencies of transmission and the available transmission options are discrete, we can formulate the distance between two subsequent frequencies as follow:

$$\begin{aligned} \Delta f &= F_{Hz}(n-1) - F_{Hz}(n) \\ &= \frac{ED}{((n-1) \cdot ED + LS) \cdot (n \cdot ED + LS)} \end{aligned} \tag{5}$$

An important note is that the higher n is, the harder it is to differentiate between two consecutive frequencies.

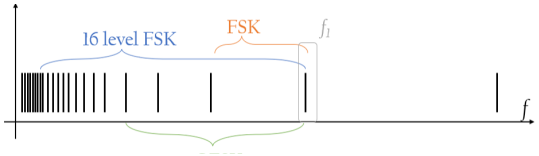


Fig. 12. illustrates the dispersion of available frequencies.

C. Unlocking Efficiency: Finding the Optimal Bit Allocation for Symbol Encoding

At first glance, the question seems simple. One might think that optimizing encoding frequencies to fit the most bits per symbol is the best solution. To simplify our model, we examined the group $F_g = \{f(N_1), f(N_1+1), \ldots, f(N_1+n-1)\}$. As can be deduced from the analysis of frequency distribution in prior sections, it becomes apparent that lower frequencies pose a bigger challenges in differentiation from adjacent counterparts. Consequently, the two least frequencies within the group F_g are particularly challenging to distinguish between. Therefore, in establishing constraints for our model, it becomes imperative to ensure that this frequency disparity exceeds the resolution of the Fast Fourier Transform (FFT), $\mathcal{F}_{res}$, i.e., constraint:

$$\begin{aligned} \mathcal{F}_{res} &= \frac{\text{Sampling Freq.}}{\text{FFT Size}} \\ &= \frac{\text{Sampling Freq.}}{\text{Sampling Freq.} \cdot (NB_1(ED \cdot N_1 + LS))} \\ &= \frac{1}{(NB_1(ED \cdot N_1 + LS))} \\ &\leq \frac{ED}{((N1+n-2) \cdot ED + LS)(ED(N_1+n-1) + LS)} \end{aligned} \tag{6}$$

In an optimal scenario, we aim to minimize the temporal magnitude of the symbol and equate these two expressions, in which case we can separate NB_1:

$$NB_1 = \frac{(LS + ED(N_1+n-2))(LS + ED(N_1+n-1))}{ED(N_1 \cdot ED + LS)} \tag{7}$$

We will now employ the substitution of NB_1 into Eq. (1) to derive the Bit Rate for each modulation scheme, disregarding the supplementary influence of synchronization due to its irrelevance to the present calculation:

$$\begin{aligned} BR(n) &= \frac{\log_2(n)}{NB_1 \cdot (N_1 \cdot ED + LS)} \\ &= \frac{\log_2(n)}{(ED(N_1+n-1) + LS)(ED(N_1+n-2) + LS)} \end{aligned} \tag{8}$$

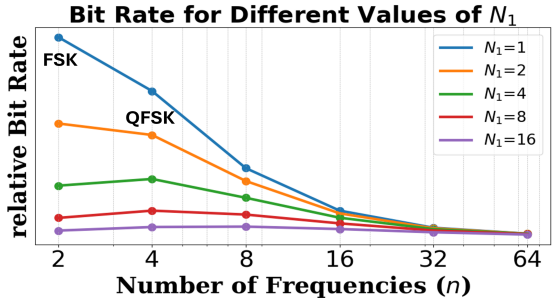


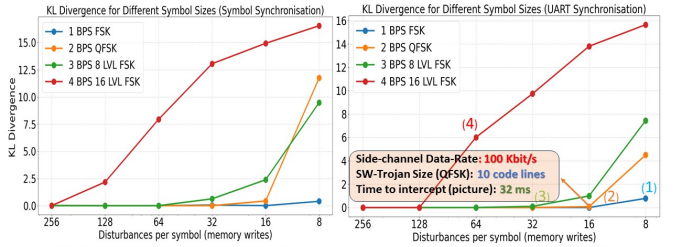
Fig. 13. This graph illustrates bit rate for different fsk modulation and different starting frequencies N_1

In an optimal scenario, where noise, sample rate and synchronization considerations are disregarded ($N_1 = 1$), the model reveals that Frequency Shift Keying (FSK) outperforms other modulation schemes, contrary to our initial assumption.

However, as discussed in the introductory chapter, not all devices can achieve the necessary sampling frequency to accurately discern all frequencies. In such circumstances, opting for a higher initial frequency (N_1) becomes imperative. In this case and as illustrated in Fig. 13, alternative modulations QFSK ($N_1 = 4$) or 8-Level-FSK may offer greater suitability and improved bit rate performance. Another advantage of employing $n > 2$ frequencies lies in the increased separation between adjacent frequencies, beyond f_{n-1} and f_{n-2}. This enhanced separation plays a crucial role in reducing error rates, particularly in environments characterized by high levels of noise. Notably f_1 and f_2 are expected to exhibit minimal error rates due to their significant frequency separation, as elaborated in subsequent sections. Synchronization within the channel relies heavily on these two frequencies, a higher quality of f_1 and f_2 ensures stable synchronization.

D. Results and discussion

To evaluate the maximum data rate achievable by the system, we created an optimal measurement environment. In this environment, the receiving antenna was placed in close proximity to the transmitting antenna, and the device was operated in an area free from electrical disturbances within the relevant frequency ranges, we chose $N_1 = 1$, meaning we used the highest subsequent frequencies set our model facilitate. We assessed the transmission capabilities of various modulation schemes, including Frequency Shift Keying (FSK), Quadrature Frequency Shift Keying (QFSK), 8-level FSK, and 16-level FSK. Each modulation type was tested under varying of electrical disturbances (ED) per symbol we controlled this parameter by changing the number of transmission blocks (NB), thus reducing the symbol temporal duration and increasing the data rate. This reduction in symbol duration also resulted in lower FFT resolution due to fewer data points per symbol, reducing the signal-to-noise ratio (SNR) and making it more difficult to distinguish between symbols.



UART Synchronization data rate table [kbits/s]

BPS \ DPS	256	128	64	32	16	8
1 (FSK)	3.125	6.25	12.5	25	50	100
2 (QFSK)	6.25	12.5	25	50	100	200
3 (8-LVL-FSK)	9.375	18.75	37.5	75	150	300
3 (16-LVL-FSK)	12.5	25	50	100	200	400

Symbol Synchronization data rate table [kbits/s]

BPS \ DPS	256	128	64	32	16	8
1 (FSK)	3.472	6.944	13.88	27.77	55.55	111.11
2 (QFSK)	6.944	13.88	27.77	55.56	111.11	222.2
3 (8-LVL-FSK)	10.41	20.83	41.66	83.33	166.6	333.3
3 (16-LVL-FSK)	13.88	27.77	55.55	111.11	222.2	444.4

Fig. 14. results illustrating different modulations settings.

In Fig. 14 we present two sub-figures that illustrate the relationship between the statistical distance from the original image and the number of electrical disturbances (ED) per symbol in two different synchronization schemes (in practice we calculated and changed NB according to the chosen number of electrical disturbances, ED). Comparing these sub-figures reveals that the symbol synchronization plot is slightly shifted to the left, indicating a higher error rate or statistical distance for the same parameters compared to the UART synchronization plot. This shift occurs because symbol synchronization is based on a larger set of frequencies (Sec. IV-D) A larger frequency set means that the lowest frequencies are closer together, making differentiation more challenging, as shown in (Fig. 12). The difference in data rates between the models results is due to the varying number of synchronization bits used in the two schemes (Fig. 7). Upon delving into the graphs, a notable trend emerges: the error rates increase as the symbol duration decreases. Moreover, in accordance with the predictions of our mathematical model, selecting $f_1 = 1$ reveals that regular FSK yields superior results in term of data rates compared to error rates or statistical distance in our case. In Fig. 14,15, the numbers (1) to (4) on the reconstructed images and corresponding data points on the UART synchronization plot denote the least reasonable reconstructed image in terms of error rate for each modulation. The red lines in the figures indicate modulation settings that result in reconstructed images of such poor quality that they are unsuitable for use in high-end systems.

VII. Conclusions

In this work, we combine the concepts of RF side-channel attacks and *software-code* Trojans for the first time, creating a *hard-to-detect* and stealthy method to extract information over considerable distances using a *single measurement*. This technique leverages Trojan-induced electrical disturbances in peripherals, buses, memory, and the CPU, which leak into RF components and enable signals modulation across various constellation types. Our goal is to achieve a high signal-to-noise ratio (SNR), high data transmission rates (DR), and *negligible capture time*.In the preceding sections, we demonstrate how to optimize modulation and demodulation schemes and construct specialized synchronization symbols to further minimize errors and maximize DR. The next table summarizes our results:

TABLE II
Sample Data for Different Setups

Setup	Noisy Area	Cap. Time[s]	Dist.[m]	DR[Kb/s]
High Data Rates	×	0.032	5	100
Open Space	×	0.512	25	6.25
Wall	✓	0.512	10	6.25
Corridor	✓	0.512	17.5	6.25

References

[1] P. Szor, *Art of Computer Virus Research and Defense, The, Portable Documents*. Pearson Education, 2005.

[2] N. K. Dixit, L. Mishra, M. S. Charan, and B. K. Dey, "The new age of computer virus and their detection," *International Journal of Network Security & Its Applications*, vol. 4, no. 3, p. 79, 2012.

[3] C. Cowan, F. Wagle, C. Pu, S. Beattie, and J. Walpole, "Buffer overflows: Attacks and defenses for the vulnerability of the decade," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2. IEEE, 2000, pp. 119–129.

[4] X. Zhang, Y. Xiao, and Y. Zhang, "Return-oriented flush-reload side channels on arm and their implications for android devices," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 858–870.
[5] J. Lee, J. Jang, Y. Jang, N. Kwak, Y. Choi, C. Choi, T. Kim, M. Peinado, and B. B. Kang, "Hacking in darkness: Return-oriented programming against secure enclaves," in *26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 523–539.
[6] L. Yin, C. Wang, J. Li, R. Yin, Y. Jiao, and H. Jiang, "When side channel becomes good: Kernel malware attack investigation," in *Artificial Intelligence and Security: 5th International Conference, ICAIS 2019, New York, NY, USA, July 26-28, 2019, Proceedings, Part II 5*. Springer, 2019, pp. 571–583.
[7] B. Singh, D. Evtyushkin, J. Elwell, R. Riley, and I. Cervesato, "On the detection of kernel-level rootkits using hardware performance counters," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, 2017, pp. 483–493.
[8] G. Camurati, S. Poeplau, M. Muench, T. Hayes, and A. Francillon, "Screaming channels: When electromagnetic side channels meet radio transceivers," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 163–177.
[9] E. Danieli, M. Goldzweig, M. Avital, and I. Levi, "Revealing the secrets of radio embedded systems: Extraction of raw information via rf," *IEEE Transactions on Information Forensics and Security*, 2023.
[10] R. A. Riley, J. T. Graham, R. M. Fuller, R. O. Baldwin, and A. Fisher, "A new way to detect cyberattacks: Extracting changes in register values from radio-frequency side channels," *IEEE Signal Processing Magazine*, vol. 36, no. 2, pp. 49–58, 2019.
[11] J. Feng, T. Jacques, O. Abari, and N. Sehatbakhsh, "Everything has its bad side and good side: Turning processors to low overhead radios using side-channels," in *Proceedings of the 22nd International Conference on Information Processing in Sensor Networks*, 2023, pp. 288–301.
[12] C. Shen, T. Liu, J. Huang, and R. Tan, "When lora meets emr: Electromagnetic covert channels can be super resilient," in *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2021, pp. 1304–1317.
[13] K. S. Subramani, N. Helal, A. Antonopoulos, A. Nosratinia, and Y. Makris, "Amplitude-modulating analog/rf hardware trojans in wireless networks: Risks and remedies," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3497–3510, 2020.
[14] N. Miguélez-Gómez and E. A. Rojas-Nastrucci, "Rf fingerprinting: hardware-trustworthiness enhancement in the hardware trojan era: Rf fingerprinting-based countermeasures," *IEEE Microwave Magazine*, vol. 24, no. 11, pp. 35–52, 2023.
[15] R. Wang, H. Wang, and E. 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*, 2020, pp. 35–44.
[16] R. Wang, H. Wang, E. Dubrova, and M. Brisfors, "Advanced far field em side-channel attack on aes," in *Proceedings of the 7th ACM on Cyber-Physical System Security Workshop*, 2021, pp. 29–39.
[17] H. A. Noman and O. M. Abu-Sharkh, "Code injection attacks in wireless-based internet of things (iot): A comprehensive review and practical implementations," *Sensors*, vol. 23, no. 13, p. 6067, 2023.
[18] N. Sehatbakhsh, M. Alam, A. Nazari, A. Zajic, and M. Prvulovic, "Syndrome: Spectral analysis for anomaly detection on medical iot and embedded devices," in *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2018, pp. 1–8.
[19] Z. Ling, H. Yan, X. Shao, J. Luo, Y. Xu, B. Pearson, and X. Fu, "Secure boot, trusted boot and remote attestation for arm trustzone-based iot nodes," *Journal of Systems Architecture*, vol. 119, p. 102240, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762121001661
[20] G. Dar, G. Di Natale, and O. Keren, "Nonlinear code-based low-overhead fine-grained control flow checking," *IEEE Transactions on Computers*, vol. 71, no. 3, pp. 658–669, 2021.
[21] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.
[22] CISPR22, "International standard EN 55022:2010," http://http://www.rfemcdevelopment.eu/en/en-55022-2010, 2010, [Online;].

VIII. Appendix

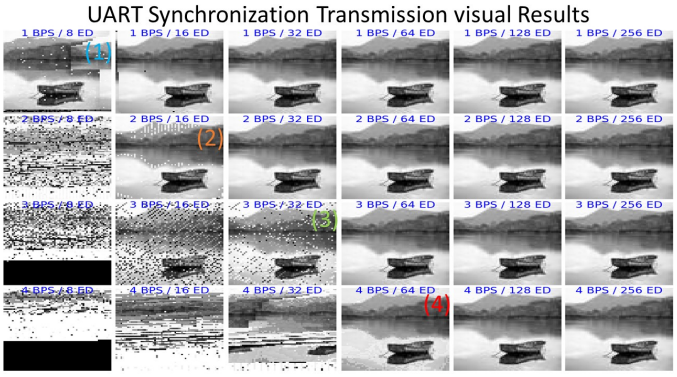


Fig. 15. pictures presenting different UART synchronisation modulations settings.

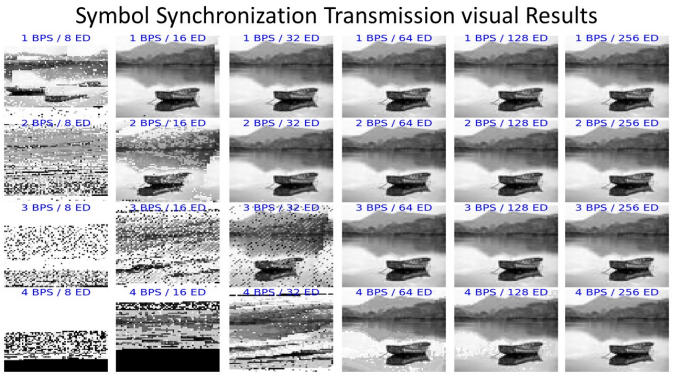


Fig. 16. pictures presenting different Symbol synchronisation modulations settings.

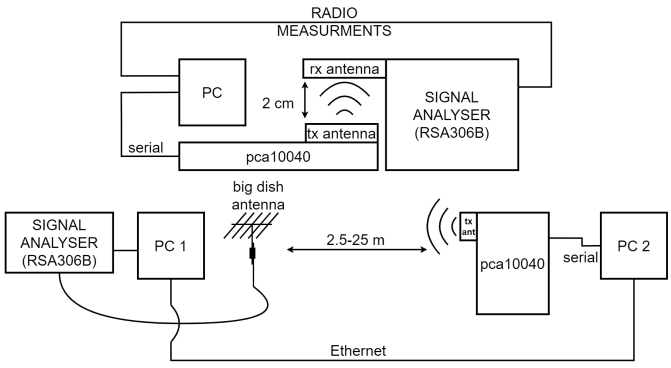


Fig. 17. pictures presenting different modulations settings.