BitJabber: The World's Fastest Electromagnetic Covert Channel

Zihao Zhan[†§], Zhenkai Zhang[†], Xenofon Koutsoukos[§]
[†]Texas Tech University, [§]Vanderbilt University
Email: {zzhan, zhenkai.zhang}@ttu.edu, {zihao.zhan, xenofon.koutsoukos}@vanderbilt.edu

***Abstract*—An air-gapped computer is physically isolated from unsecured networks to guarantee effective protection against data exfiltration. Due to air gaps, unauthorized data transfer seems impossible over legitimate communication channels, but in reality many so-called physical covert channels can be constructed to allow data exfiltration across the air gaps. Most of such covert channels are very slow and often require certain strict conditions to work (e.g., no physical obstacles between the sender and the receiver). In this paper, we introduce a new physical covert channel named *BitJabber* that is extremely fast and strong enough to even penetrate concrete walls. We show that this covert channel can be easily created by an unprivileged sender running on a victim's computer. Specifically, the sender constructs the channel by using only memory accesses to modulate the electromagnetic (EM) signals generated by the DRAM clock. While possessing a very high bandwidth (up to 300,000 bps), this new covert channel is also very reliable (less than 1% error rate). More importantly, this covert channel can enable data exfiltration from an air-gapped computer enclosed in a room with thick concrete walls up to 15 cm.**

I. Introduction

In organizations where information security and privacy are top priorities, physical isolation is often used to prevent data exfiltration. Air-gapping is considered as one of the strongest physical isolation method that has been widely used by, e.g., militaries and governments. An air-gapped computer has no connections with the outside unsecured networks, so that it is believed that protection against unauthorized data transfer can be effectively guaranteed.

However, recent research has discovered that many physical side effects of computation on air-gapped computers can be exploited to construct so-called physical covert channels to re-enable data exfiltration. The physical side effects that can be exploited are various, including thermal [10], optical [15], [19], [20], [26], magnetic [5], [14], [22], acoustic [2], [11], [12], [16], or electromagnetic (EM) [7]–[9], [33]. The communication distance of such covert channels is usually very short, ranging from several centimeters to several meters, due to the high attenuation of the exploited physical effects in the distance. Information is encoded within the physical effects and transferred over the air gaps between a sender and a receiver. Normally, a sender is a piece of malware, like a Trojan horse, that has been stealthily inserted into a victim's computer, and a receiver is some device in the proximity of the sender that can capture the exploited physical effects.

Nevertheless, the security risks of such covert channels are often neglected, as they are considered hardly posing any real hazards for two reasons. First, the bandwidth of such physical covert channels is usually very low. For example, the transmission rate of the covert channel proposed in [10] is only 8 bits/hour (i.e., 0.002 bps). Even the fastest one reported in [15] can only reach 4,000 bps. Therefore, if a large amount of data needs to be exfiltrated, an attacker has to maintain the covertly communicating status for a long period of time. In a situation where the attacker can briefly have her foothold in the proximity to the targeted computer, any lingering action may cause suspicion. Second, most of these covert channels require no physical obstacles between the sender and receiver. Thus, an attacker may encounter great difficulties in managing the placement of the receiving device. In particular, locking an air-gapped computer in an enclosed room has been regarded as a sufficiently secure protection against such physical covert channels.

In this paper, we demonstrate that there in effect exist powerful covert channels that are extremely fast and strong enough to penetrate even thick concrete walls. Specifically, we construct such a covert channel named *BitJabber* from the EM signals generated by the DRAM clock. As discovered in [1], there are strong EM signals generated by different clocks in a computer that can propagate far, and these EM signals can be amplitude-modulated (AM) by activities driven by the corresponding clocks. Therefore, the EM signals generated by the DRAM clock can be AM-modulated by normal memory accesses to carry and transfer information over the air gaps between a pair of sender and receiver, namely forming an electromagnetic covert channel. Our experimental results show that the transmission rate of this new covert channel can reach 100,000 bps using binary frequency-shift keying modulation (B-FSK) with error rate around 0.3%, and 300,000 bps using multiple frequency-shift keying modulation (M-FSK) with error rate less than 1%. Moreover, this covert channel is resilient to a reasonable level of background noise and works well even in the presence of 15 cm thick concrete walls between the sender and the receiver.

The main contributions of this paper are three-fold:

- We present a new physical covert channel named *BitJabber* that can allow expedited data exfiltration between air-gapped sender and receiver.
- We verify that our *BitJabber* covert channel is much more resilient to background noise compared with the state-of-the-art ones.
- We demonstrate that this new covert channel can achieve

reliable communication within a few meters, even under the scenario where the sender and the receiver are in separate rooms with concrete walls in-between.

The rest of this paper is organized as following: Section II briefs existing work on physical covert channels and makes a comparison across different approaches. Section III states the threat model considered in this paper. Section IV presents our *BitJabber* covert channel in detail, including the techniques for modulation, demodulation and synchronization. Section V evaluates the performance of *BitJabber*. Section VI lists some possible countermeasures against this new covert channel and Section VII concludes this paper.

II. Related Work

The confinement problem was brought forth by Lampson in 1973 [18], which made the first mention of possible data exfiltration via covert channels. Since then, extensive research has been conducted on this topic. Basically, a covert channel is an unintended communication channel that can be used to transfer information between a sender and a receiver. Depending on the construction, covert channels can be classified into logical and physical ones. Logical covert channels usually manipulate the microarchitectural states in a processor to encode and transfer information [28], and the receiver normally runs on the same processor/platform/cloud as the sender [21], [23], [25], [27], [29], [30], [32]. On the other hand, physical covert channels are usually used to enable illegitimate communication between air-gapped computers, and are constructed from certain physical side effects of computation. In this section, we will mainly focus on physical covert channels.

A. Physical Convert Channels

A running computer can affect its physical environment in many ways, such as issuing heat, producing sound, emitting light, and generating EM signals. These affections are often called physical side effects of computation. To exploit such physical side effects to construct covert channels, the sender needs to be able to manipulate them in a controlled way such that information can be encoded within the physical side effects. As these physical side effects can propagate to a certain distance in the air, the carried information can be transferred over the air gaps. On the receiver side, the attacker measures the environmental changes introduced by the sender and interprets the measurement properly to recover the exfiltrated information. Many physical side effects of computation have been reported as exploitable for constructing physical covert channels.

Since many components (e.g., clocks and voltage regulators) in a computer have switching behavior and thus emit strong EM signals, several EM covert channels have been created. For example, Guri *et al.* implemented multiple EM covert channels by exploiting the EM emanations from either video display unit [8], USB connectors [9], or DRAM bus clock [7]. Similar to our *BitJabber* cover channel, their *GSMem* covert channel described in [7] also relies on the EM signals related to the DRAM clock. They discovered that memory accesses can increase the strength of the EM signals in a wide frequency range around the DRAM clock frequency. By controlling the presence/absence of intense memory access behavior, the EM signals around the DRAM clock frequency can carry information through on-off keying modulation (OOK). In our work, *BitJabber* is implemented using a different carrier with much higher signal-to-noise ratio (SNR) and new modulation techniques. Section V will present the results showing that *BitJabber* outperforms *GSMem* significantly in terms of both speed and reliability. Note that EM signals can penetrate walls and be easily measured by some cheap devices, e.g., software-defined radios or mobile phones, but they can be blocked by metal shields like Faraday cage.

As the magnetic field around a computer can be affected by manipulating components like hard disk drives [22] and CPUs [5], [14], magnetic covert channels have also been constructed. The magnetic field can be measured by either some specialized equipment like digital magnetometer or any hardware equipped with magnetic sensors like mobile phones. Normally, magnetic covert channels have very low transmission speed and extremely short transmission distance. Unlike EM signals, the low frequency magnetic emanations cannot be blocked by metal shields. However, due to their limited transmission distance, magnetic emanations are very unlikely to be exploited for data exfiltration through a thick obstacle like a concrete wall.

Sound is frequently produced by a running computer, and any device with a microphone can receive these signals. The first acoustic covert channel was implemented by Carrara *et al.*, who used speakers and microphones on computers to communicate through ultrasound [2]. Further, Hanspach *et al.* leveraged ultrasound to establish covert acoustical mesh network [16]. Because the frequencies of ultrasound are higher than the upper audible limit of human hearing, the communication cannot be easily noticed. Later, Guri *et al.* designed speakerless acoustic covert channels, where cooling fans [11] and hard disk drives [12] were used to generate acoustic emissions. However, the abnormal noise generated by fans and/or hard disk drives may be easily noticed by perceptive people, which makes them less stealthy. To some extent, acoustic signals can travel through obstacles, but their strength may be significantly attenuated depending on the material of the obstacles. Besides, most acoustic covert channels have very low bandwidth.

Optical emissions can also be exploited to create covert channels. The exploitable optical emissions may be generated by light-emitting diode (LED) in components like keyboards [20], monitors [26], and even hard disk drives [15]. Most LED-based optical covert channels use OOK modulation, and Zhou *et al.* showed that the efficiency can be improved by replacing OOK modulation with binary frequency-shift keying modulation (B-FSK) [35]. Another kind of optical covert channel manipulates the monitor screen [6]. By modifying a small amount of content displayed on the screen, information may be transmitted without being noticed by humans. Theoretically, optical covert channels can reach a very high bandwidth with the help of optical instruments as long as the sender is in the sight of the attacker. However, exploiting optical emissions is harder than expected in practice, because it is rare that a highly secured target machine can be monitored by a malicious camera. In addition, it is very difficult, if not impossible, to create optical covert channels when the target machine is enclosed in a room with non-transparent walls. Similar to acoustic covert channels, some optical emissions like abnormal blinking of LED can also raise administrator's suspicion.

A thermal covert channel was constructed in [10] to transmit information between two physically adjacent but air-gapped computers. The advantage of this covert channel is that it can realize two-way communication. However, the performance of this covert channel is extremely poor. The maximum bandwidth reported is 8 bits/hour, and the sender and the receiver must be very close to each other.

A very recent study shows that power consumption is also exploitable for establishing covert channel [13]. In that study, CPU was manipulated to affect the power consumption of a computer to transmit information through power lines. The receiver can be mounted either on the in-home power lines that are directly attached to the electrical outlet or on the main electrical service panel. The bandwidth of this covert channel can reach 1,000 bps but it requires the installation of malicious hardware devices on the power lines connected to victim machines.

B. Comparisons

To highlight the advantages of *BitJabber*, we compare the existing physical covert channels in Table I. The comparisons are made in terms of their maximum achievable bandwidth and wall-penetrating ability. From the table we can see, before our work, the fastest physical covert channels was the one proposed in [15], which can achieve 4,000 bps. Compared to that covert channel, our *BitJabber* improves the performance by 75x.

Moreover, most of the existing physical covert channels have difficulties in penetrating physical obstacles like a wall. (We mark "maybe" on acoustic covert channels in terms of wall-penetrating ability, although we think it is very unlikely that they can actually penetrate a wall.) From the table, we can observe that the EM covert channels have considerable advantages over others in terms of penetrating walls. However, as illustrated in Section V, when penetrating concrete walls, approaches like *GSMem* actually have a too large error rate (from 38% to 50%) to be actually used in reality, while our *BitJabber* has an error rate even less than 0.5%. Therefore, compared to other physical covert channels, it can be found that our *BitJabber* imposes more realistic security risks on air-gapped isolation protection.

III. Attack Model

Similar to the previous work [2], [5], [7]–[16], [19], [20], [22], [26], in this paper, we explore how to construct a covert communication channel between a pair of air-gapped sender and receiver. We assume that the sender has been placed on the victim computer that stores or processes the secret data of interest, and the sender can acquire the secret through techniques like microarchitectural side-channels [4]. (How to place the sender there is out of scope, but, as presumed in the previous work, the attacker is capable of achieving this by methods like social engineering, USB interface, or physical access.) Note that we do not assume the sender has any privilege higher than the regular user level.

TABLE I: Comparison of existing physical covert channels.

Covert Channel	Type	Wall-Penetrating	Bandwidth
BitWhisper [10]	Thermal	No	0.002 bps
Fansmitter [11]	Acoustic	Maybe	0.25 bps
Matyunin [22]	Magnetic	No	2 bps
DiskFiltration [12]	Acoustic	Maybe	3 bps
MAGNETO [5]	Magnetic	No	5 bps
Monitor LED [26]	Optical	No	20 bps
ODINI [14]	Magnetic	No	40 bps
UltraSonic [2]	Acoustic	Maybe	230 bps
Keyboard LED [20]	Optical	No	450 bps
AirHopper [8]	Electromagnetic	Yes	480 bps
USBee [9]	Electromagnetic	Yes	640 bps
GSMem [7]	Electromagnetic	Yes	1,000 bps
PowerHammer [13]	Power	N/A	1,000 bps
Hard Drive LED [15]	Optical	No	4,000 bps
BitJabber	Electromagnetic	Yes	300,000 bps

We assume that the attacker can use a radio frequency (RF) receiver (like a cheap software-defined radio) to collect the EM signals emanated from the victim machine somewhere nearby. Note that we do not require the receiving device to share the same room with the sender or to be physically adjacent to the sender. The sender and the receiver may be in different rooms with concrete walls, and the straight-line distance between them can be one or two meters.

IV. The Design of *BitJabber* Covert Channel

As mentioned above, our *BitJabber* is an EM-based covert channel. The carrier EM signal is generated by the DRAM clock, and memory accesses are used to modulate the carrier signal to encode information. When modulated carrier signal is captured, demodulation is used to decode information from that signal. The overview of our *BitJabber* covert channel is illustrated in Fig. 1. In the following, we will describe the main components and techniques used in *BitJabber*.

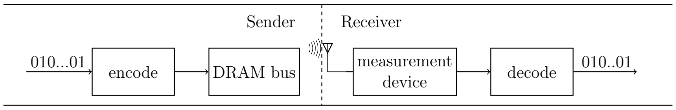


Fig. 1: Overview of *BitJabber* cover channel.

A. Spread Spectrum Clocking

Before going forward to describe the details of our *BitJabber* covert channel, we need to present a challenging problem caused by a feature named *spread spectrum clocking* (SSC). SSC has been widely used in electronic products like computers for meeting electromagnetic compatibility (EMC) regulations [3]. Due to SSC, the energy of the EM signals generated by the DRAM clock will be spread over a wide range of frequencies. Such an energy dispersion makes the exploitation of these EM signals much harder, because the power of the exploitable signals becomes weaker but the power of the background noise stays the same. As a result, the signal-to-noise ratio (SNR) is much decreased, and thus our covert channel capacity will be considerably affected. To increase the SNR, we need to use a de-spreading technique to gather the scattered signal energy back.

Fortunately, this problem has been solved recently [34]. For self-containedness, we will summarize the solution here. The detailed presentation can be found in [34].

Given a clock signal whose frequency is f_c, SSC uses FM-modulation to vary the clock frequency in accordance with a signal $f_m(t)$ that is generated in the SSC hardware chip but undocumented. Normally, $f_m(t)$ is a periodic function, namely we have $f_m(t) = f_m(t + T_m)$ where T_m is the fundamental period of $f_m(t)$. At time t, the instantaneous frequency $f_i(t)$ of the clock signal becomes:

$$f_i(t) = f_c + K f_m(t) \ , \tag{1}$$

where K is some proportionality constant. In an analytic form, the effect of SSC is equivalent to multiplying the clock signal by a complex exponential function $\theta(t)$, which is defined as:

$$\theta(t) = e^{j2\pi \int_0^t K f_m(t) dt} \ , \tag{2}$$

where j denotes $\sqrt{-1}$. Hence, for the purpose of de-spreading, we just need to estimate $\theta(t)$ and multiply the measured signal by $\theta^{-1}(t)$.

The de-spreading process proposed in [34] can be summarized in the following steps:

1) FM-demodulate the signals around the DRAM clock frequency f_c to recover a noisy version of $K f_m(t)$.
2) Find the fundamental period T_m of $K f_m(t)$ which is namely the the fundamental period of $f_m(t)$.
3) Average $K f_m(t)$ over multiple periods to recover a clean $K f_m(t)$ with much less noise.
4) Derive $\theta(t)$ according to Eq. 2 using the recovered $K f_m(t)$ and T_m.
5) Multiple the measured signal by the complex conjugate of $\theta(t)$ (i.e., $\theta^{-1}(t)$) to de-spread the signal.

De-spreading can significantly improve the capacity of our covert channel in several ways. First, de-spreading gathers the scattered energy of the exploitable EM signals (i.e., it helps strengthen the signal), while de-spreading also inadvertently acts like SSC on background noise (i.e., it helps weaken the noise). Thus, the SNR will be greatly increased. Second, the EM signals of interest will be located in a narrow frequency range after de-spreading, which allows us to use more advanced modulation techniques to utilize the spectra.

B. Modulation

To encode information into the EM signals generated by the DRAM clock, modulation is required to vary the EM wave with respect to the message contents. As it is known that the EM radiation of the DRAM clock is AM-modulated by memory accesses, the modulation for *BitJabber* covert channel is accomplished through manipulating the memory access behavior.

To understand how memory access behaviors affect the EM signals generated by the DRAM clock, we perform different memory activities on a computer equipped with DDR3-1600 memory modules (i.e., the DRAM clock frequency is 800MHz) and investigate the corresponding spectra, which are shown in Fig. 2. At first, no intense memory accesses are performed. As illustrated in Fig. 2, the EM radiation after de-spreading has most of its energy concentrated near the clock frequency (i.e., 800MHz). When memory accesses with execution time around 350ns are repeatedly performed, raised energy can be observed at certain frequencies in the lower and upper sidebands. The offsets of these lobes from 800MHz are multiples of the memory access frequencies (i.e., 2.86MHz), which indicates that the EM radiation is AM-modulated by a non-sinusoidal wave with the same frequency as the memory accesses. If some delay is added to make the memory accesses slower, the positions where the lobes locate indicate that the frequency of the modulating non-sinusoidal wave also decreases. (Note that we use non-temporal load/store instructions like `MOVNTI` to avoid memory accesses being served directly from the CPU caches.)

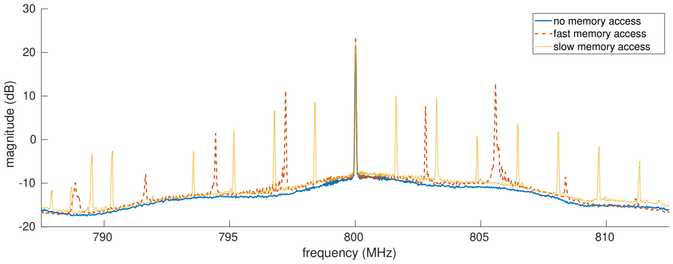
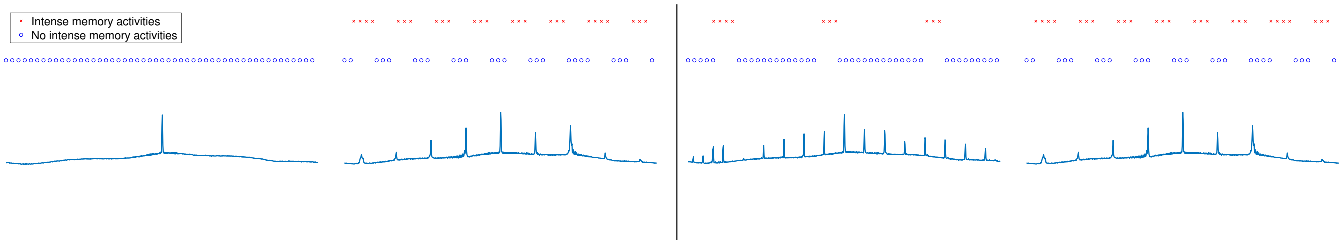


Fig. 2: Spectra of different memory access behaviors

The above observation shows that not only do intense memory accesses introduce obvious lobes in the sidebands, but also the memory access frequency has influence on where these lobes locate. Accordingly, two modulation techniques can be applied to encode information into the EM signals generated by the DRAM clock:

- The first and also the simplest modulation method is OOK. As shown in Fig. 3 (a), OOK uses the presence and absence of repeated memory accesses to encode bit ‘1’ and bit ‘0’. Consequently, the AM-modulated EM signal will have side lobes in its spectrum only when ‘1’ is transmitted; otherwise, ‘0’ is sent.
- The other modulation method is FSK, indicated by Fig. 3 (b), where different symbols are represented by different memory access frequencies. For example, to send bit ‘1’, fast memory accesses are repeated, and to send bit ‘0’, slow accesses are repeatedly made. Thus, different distances between the side lobes and the clock frequency



(a) OOK Modulation

(b) FSK Modulation

Fig. 3: Encoding of 0 and 1 using two modulation methods – OOK and FSK

in the spectra can distinguish these two cases. To realize different memory access frequencies, we can use a normal memory access as the fast one and introduce some delay to derive the slow one.

Note that the above-mentioned FSK modulation is not limited to B-FSK, in which case either bit '0' or '1' is transmitted. Because any two different memory access frequencies can result in distinguishable side lobe positions in the spectra, M-FSK modulation is also achievable by adding distinct delays to a base memory access activity `BaseMemAcc` as depicted in Algorithm 1. (The details of the `BaseMemAcc` activity will be described later.)

```
Input: $T_i$ = delay time for transmitting symbol $i$
if Transmitting symbol $i$ then
    BaseMemAcc;
    DELAY($T_i$);
end
```

Algorithm 1: Memory activities for M-FSK modulation

C. Base Memory Access Design

We have observed that randomly accessing some memory addresses may not AM-modulate the EM signals generated by the DRAM clock well. Thus, we need to have a systematic way to construct a memory access activity such that the probability of AM-modulating the EM signals of interest well is very high. We term this systematically-constructed memory access activity as base memory access `BaseMemAcc`.

We need `BaseMemAcc` to have the following three properties:

1) It should have a very short execution time (e.g., a few hundreds of nanoseconds).
2) It should have a relatively stable execution time.
3) It should induce obvious change in the amplitude of the EM signals generated by the DRAM clock.

To design such a base memory access activity, we need to understand how memory accesses affect the DRAM clock. Although it has been investigated in some prior work [1], [7], factors that influence the AM-modulation effect were not fully identified.

To satisfy the first two properties, we decide to use non-temporal memory access instructions, such as `MOVNTI`, `MOVNTDQ` and `VMOVNTDQ`. Since they will bypass the CPU caches, we can use them to directly access the main memory in a rapid manner. Otherwise, `CLFLUSH` instruction needs to be used to flush the cache after each memory access, which brings in more overhead and execution time variation. These non-temporal memory access instructions can support operands of different sizes, e.g., either 32-bit or 64-bit operands can be used in `MOVNTI`. The operand size can affect the execution time slightly, but can result in observable differences in side lobe positions in the spectrum.

We notice that memory locations may have a significant influence on the AM-modulation. In order to find out how the AM-modulation effect is related to the memory access instructions and memory locations, we conduct experiments and empirically conclude the following:

1) When the same memory access instruction is used to access the same memory location, the AM-modulation effect (e.g., side lobe positions and their energy) is fixed.
2) When different types of non-temporal instructions are used to access the same memory location, the AM-modulation effect is slightly different.
3) When the same instruction is used to access different memory locations, the amount of amplitude change of the EM signals of interest may be significantly different. The relationship between accessed address and the amount of amplitude change is still not clear, but in our tested platforms we notice that accessing memory addresses in the same DRAM bank tends to change the amplitude similarly.

Therefore, the more memory locations are accessed, the higher the possibility that obvious amplitude change will arise is. Based on the above observations, to satisfy the third property, `BaseMemAcc` needs to access several fixed memory locations using the same non-temporal memory access instruction. Apparently, there is a trade-off, because the more memory locations are accessed, the slower `BaseMemAcc` will become. We empirically find that accessing 4 memory locations is sufficient to have obvious AM-modulation effect while keeping the execution time short.

Note that if these fixed memory locations are randomly selected, it may incur unpredictable variations in the execution time due to row buffer conflicts in the same DRAM banks [17], [24], [31]. Such variations may make the second required property of `BaseMemAcc` violated. Therefore, it is preferable to have these memory locations in different DRAM banks.

Moreover, considering that in some platforms the amplitude change is bank-dependent, this memory location selection strategy can even help `BaseMemAcc` hold the third property. Thus, we design `BaseMemAcc` to be a memory access activity that uses a fixed non-temporal memory access instruction to access 4 fixed memory locations in different DRAM banks.

However, finding memory locations belonging to different DRAM banks can be a problem, because the address mapping information is unavailable to unprivileged attackers. To obtain such memory locations, we use a method exploiting a timing side-channel introduced by the row buffer conflicts in the same DRAM banks [17], [24], [31]. Given two virtual addresses a_1, a_2, a function $\texttt{LATENCY}(a_1, a_2)$ is used to check whether they are in the same bank. If they are in the same bank, accessing them consecutively is relatively slow due to the delay induced by the row buffer conflict, and $\texttt{LATENCY}(a_1, a_2)$ returns `True`; otherwise, accessing them is faster and $\texttt{LATENCY}(a_1, a_2)$ returns `False`. The memory location selection method is described in Algorithm 2. By repeating this method, we can derive several groups, in each of which the addresses are located in the same DRAM bank.

```
Input  : $AP$ = address pool
Output: $G$ = addresses mapped to the same bank
RefAddr ← $AP$.DEQUEUE();
$G$.ENQUEUE(RefAddr);
$n$ ←SIZEOF($AP$);
for $i \leftarrow 1...n$ do
    RemAddr ← $AP$.DEQUEUE();
    if LATENCY(RemAddr,RefAddr) then
        $G$.ENQUEUE(RemAddr)
    else
        $AP$.ENQUEUE(RemAddr)
    end
end
```

Algorithm 2: Grouping virtual addresses *w.r.t.* banks

D. Demodulation

After the EM signals are captured by the receiver, demodulation is needed to recover the encoded information from the AM-modulated signals. For our *BitJabber* covert channel, the key problem of demodulation is to classify different symbol values according to the energy distribution of frequencies.

As shown in Fig. 2, when memory accesses are performed at a fixed frequency to transmit a symbol value corresponding to that frequency, side lobes appear at the first few harmonics of that frequency. The first stage of our demodulation method is to extract features from the side lobes. To better describe the feature extraction, we will use an example in which B-FSK modulation is employed. We assume the clock frequency is f_c, and memory access frequencies are f_{zero} and f_{one} for encoding bit '0' and '1' respectively. The steps of feature extraction are as follows:

1) Find all the frequencies where side lobes locate in the spectrum of the captured EM signal (which is a sequence of sampled values). In our example, let us assume there are $2N$ lobes at $f_c \pm k_0 f_{\text{zero}}$ where $1 \leq k_0 \leq N$ and $2M$ lobes at $f_c \pm k_1 f_{\text{one}}$ where $1 \leq k_1 \leq M$.
2) For each frequency where a side lobe locates, apply a bandpass filter on the original signal to preserve only the energy of that frequency. For our example, there will be $2N + 2M$ filtered signals after this step.
3) Segment each filtered signal using the boundary finding technique described in Section IV-E. After this step, every filtered signal will have the same number of segments numbered from 0 to $W - 1$.
4) Average all the values within each segment. After this step, each filtered signal will be represented by W averaged values.
5) Form a feature vector from the averaged values of the segments with the same segment number over all the filtered signals. In our example, this step will result in W vectors, each of which has $2N + 2M$ elements.

The second stage of our demodulation method is to use a classifier to categorize a feature vector to a symbol value. Any classification technique may be used for this purpose, and we find the performance of SVM (support vector machine) is satisfactory as demonstrated in Section V.

E. Synchronization

As stated above, correctly segmenting a filtered signal is an essential step, which guarantees that parts of the signal corresponding to different symbol values will not affect each other. Moreover, we need to find the mapping between feature vectors and symbol values.

1) Finding Segment Boundaries: To segment a filtered signal correctly, we need to find which sample position in the signal corresponds to the start of a symbol, namely a segment boundary.

Assume we know the symbol n_0 starts at sample s_0. The next symbol n_1 will start at sample $s_0 + L$, where L is the number of samples used for transmitting a symbol in the ideal case. Because the symbols are sent with a known symbol rate R_{symbol} and the EM signal of interest is sampled with a known sampling rate R_{sample}, essentially L is:

$$L = \frac{R_{\text{sample}}}{R_{\text{symbol}}} \tag{3}$$

(Note that we choose R_{sample} divisible by R_{symbol} by design, so L is an integer.) Because the sender and receiver are driven by different clocks, there exists inevitable clock drift δ. Although in reality δ is very small (e.g., around 0.001%), the accumulated error can reach a level such that a compensation in the symbol length is needed. Therefore, the symbol n_x will actually start at s_x expressed as:

$$s_x = s_0 + L + \lfloor x \times \delta \times L \rfloor \tag{4}$$

However, δ is an unknown value, and even worse, we do not know s_0 as well. To solve the problem of finding symbol boundaries, we take advantage of the fact that *only if the boundaries are correctly found, each dimension of the feature* *vectors will have a large variance*. The rationale is that two feature vectors corresponding to two symbol values should have at least one dimension in which values are distinct. We will use the following steps to find the boundaries:

1) Randomly select a sample position p_i, and use $p_i \pm kL$ where $k = 0, 1, ..., K$ as segment boundaries. Here K is an empirically chosen value that makes $\lfloor K \times \delta \times L \rfloor \ll L$. Since δ is very small, K can be a number ranging from 100 to 1000.
2) Construct $2K + 1$ feature vectors from the above segmented samples and compute the variance in each dimension of these vectors. The sum of all the variances represents a score. The *higher* the score is, the *closer* p_i is to the real segment boundary.
3) Move to the next sample position p_{i+1}, and repeat the steps above. After $L + 1$ repetitions, a boundary must be crossed once, at which time the score reaches the maximum.

For example, Fig. 4 shows the score values derived when finding segment boundaries in terms of a captured signal. The sampling rate is 25MHz, and the symbol rate is 100,000Bd, which means L is 250 according to Eq. 3. As observed in the figure, the scores change quasi-periodically and their peaks denote that the corresponding positions are the segment boundaries.

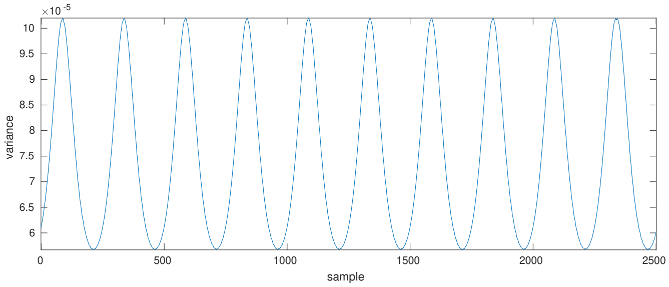


Fig. 4: Scores obtained at different sample positions

2) Deriving Symbol Mapping: To transmit a message, the sender will first transmit a head followed by its payload. The header needs to contain a sequence known to the receiver for both message synchronization and symbol mapping derivation. For better reliability, the header needs to satisfy three requirements:

1) There should be enough symbol value changes to guarantee that if the signals are segmented incorrectly the scores will be very small.
2) There should not be a repeated single symbol value pattern to guarantee that a unique symbol mapping can be derived.
3) There should be a nearly uniform distribution of the symbol values in the sequence to better train a classifier.

These requirements can usually be satisfied by a sequence generated by a pseudo random number generator as long as the seed is shared by the sender and receiver. In addition, other metadata can be embedded in the header such as sequence number, payload length, etc.

After a signal is captured, the receiver first performs feature extraction to produce a set of feature vectors. Although the mapping between feature vectors and symbol values is still unknown, the receiver can use an unsupervised machine learning technique like k-means to cluster the feature vectors. Since the symbol sequence inside the header is a shared knowledge between the sender and receiver, this sequence can be used to assign each symbol value to the corresponding cluster.

V. Experimental Results

In this section, we will evaluate the performance of our *BitJabber* covert channel in terms of its bandwidth, error rate, and capability of wall-penetrating. In the evaluations, we also compare our *BitJabber* with the existing *GSMem* covert channel [7] for the following two reasons:

1) The performance of covert channels depends on many factors like background noise and the physical architecture of the sender machine.
2) Both *BitJabber* and *GSMem* covert channels use the EM emanations generated from the DRAM clock.

A. Experimental Setup

We evaluate the performance on two victim machines. The first victim machine is a Dell Optiplex 3020 desktop computer with two 4GB 1600MHz DDR3 SDRAM memory modules installed on two different DRAM channels. The second victim machine is a Dell Optiplex 990 desktop computer with two 4GB 1333MHz DDR3 SDRAM modules installed on two different DRAM channels.

The receiver uses a log-periodic (LP) antenna whose response frequency ranges from 400MHZ to 1000MHz and a software defined radio (SDR) platform LimeSDR-USB development board to collect the EM signals around the DRAM clock frequency, as shown in the left part of Fig. 5. The EM signals are preprocessed using the GNU Radio.

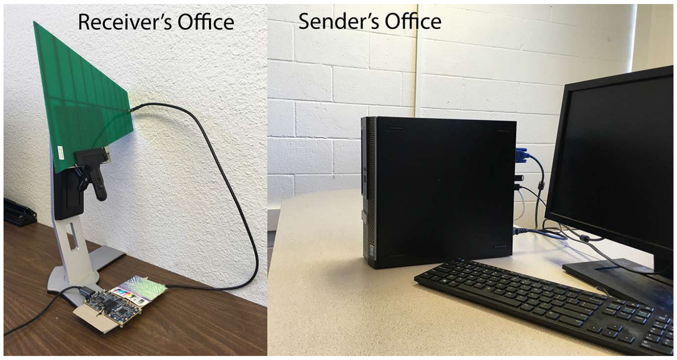


Fig. 5: Experimental setup for wall-penetrating performance evaluation

The experiments are performed in a typical office environment. In such an environment, much background noise exists, including EM waves radiated from wireless communication systems (e.g., radio stations and cell towers), nearby electronic devices, and other components in the victim computers.

The experiments are performed in two different scenarios. First, the antenna is put close to the victim machine to receive the strongest EM emanations from the DRAM clock. This experiment will show the performance upper bound of different approaches. The second scenario is to conduct the experiment in a more practical setting, as shown in Fig. 5, where the sender and receiver are located in two different offices sharing a 15cm thick concrete wall. This experiment compares the wall-penetrating data exfiltration capability of the covert channels.

B. Symbol Distinguishability

For all covert channels exploiting physical side effects, the receiver measures certain physical changes introduced by senders and transforms the measurements into different symbols. A good covert channel should have good symbol distinguishabilities. In Fig. 6, we compare the symbol distinguishabilities of two covert channels *GSMem* and *BitJabber* using the B-FSK modulation.

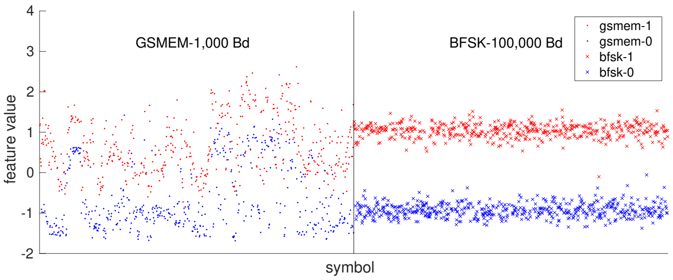


Fig. 6: Symbol distinguishability of *GSMem* and *BitJabber* using the B-FSK modulation.

For transmitting binary symbols, we can use a single feature value to represent how likely a measurement is identified to a certain symbol (either '0' or '1'). In *GSMem*, only the magnitude of the EM signal is used for distinguishing symbols with binary values, and thus we can use this as the feature value. For *BitJabber* using B-FSK modulation, an SVM model is trained to distinguish the feature vectors, and thus we use the difference of two prediction scores as the feature value. The feature values of *GSMem* at 1,000Bd symbol rate and *BitJabber* using the B-FSK modulation at 100,000Bd symbol rate are illustrated in Fig. 6. Compared to *GSMem*, it is apparent that the measurements of *BitJabber* have much larger difference between different symbol values and smaller variances between same symbol values even if the symbol rate is 100 times higher. This comparison indicates that our *BitJabber* can greatly outperform the *GSMem*, which is demonstrated by the following experimental results.

C. Bandwidth Evaluation

The first experiment measures the maximum bandwidth of *GSMem* and our *BitJabber*. To measure the performance upper bound, all measurements are performed with the antenna set at a fixed position, at which the strongest EM emanations from the DRAM clock can be collected. The EM signals are modulated by the OOK, B-FSK, and M-FSK modulation methods. Examined symbol rates range from 1,000Bd to 100,000Bd and the evaluation results are shown in Fig. 7. Because of the huge performance difference between *GSMem* and our *BitJabber*, **logarithmic scale** is used in this plot.

Note that the original *GSMem* uses the EM signals at only 800MHz. To make a fair comparison, here we report the results related to the first victim machine only. The results related to the second victim will be reported later in Section V-E. The evaluation results indicate that:

- For all approaches, the error rates increase as symbol rates get higher.
- When the bandwidth is 1000 bps, the measured error rate of *GSMem* is 4.7% which is worse than the reported result 0.087% obtained with USRP B210 SDR kit [7]. It is probably because our evaluations are performed **using much cheaper SDR hardware in a real-world environment with more realistic background noise**. As the symbol rate increases, the error rate gets higher and it is close to 50% at the highest measured bandwidth 100,000 bps.
- When the OOK modulation is used in *BitJabber*, it has a extremely low error rate which is close to 0 at low bandwidth, and the error rate only rises to 0.4% when the bandwidth is 100,000 bps.
- Compared to the OOK modulation, the B-FSK modulation can further improve the performance which can transmit binary data at 100,000 bps bandwidth with error rate around 0.25%. B-FSK modulation has the lowest error rate among all the approaches at the highest measured symbol rate.
- Using the M-FSK modulation, *BitJabber* can transmit multiple bits with each symbol effectively, the error rate is nearly zero at a low symbol rate. At 100,000Bd symbol rate, the error rates are 0.683% for 2-bit M-FSK and 0.94% for 3-bit M-FSK.
- Considering that 3-bit M-FSK modulation can transmit 3 bits with each symbol, the fastest transmission can reach **300,000 bps**. Compared to *GSMem* at its fastest transmission rate (i.e., 1000 bit/sec), **BitJabber increases the bandwidth by 300 times but decreases the error rate by a factor of 5**.

In our evaluations, we limit the maximum symbol rate to 100,000 Baud and the maximum symbol length for M-FSK to 3 bits. Theoretically, larger symbol rate and symbol length can be used for this covert channel. Nevertheless, selection these two parameters highly depends on the hardware device used to implement this covert channel. The `BaseMemAcc` used in this victim computer takes several hundred nanoseconds to execute. If symbol rate higher than 100,000 Baud is used, the actual symbol duration tends to be more unstable, which will greatly increase the error rate. As for the symbol length, when 3 bits are represented by a single symbol, 8 different memory

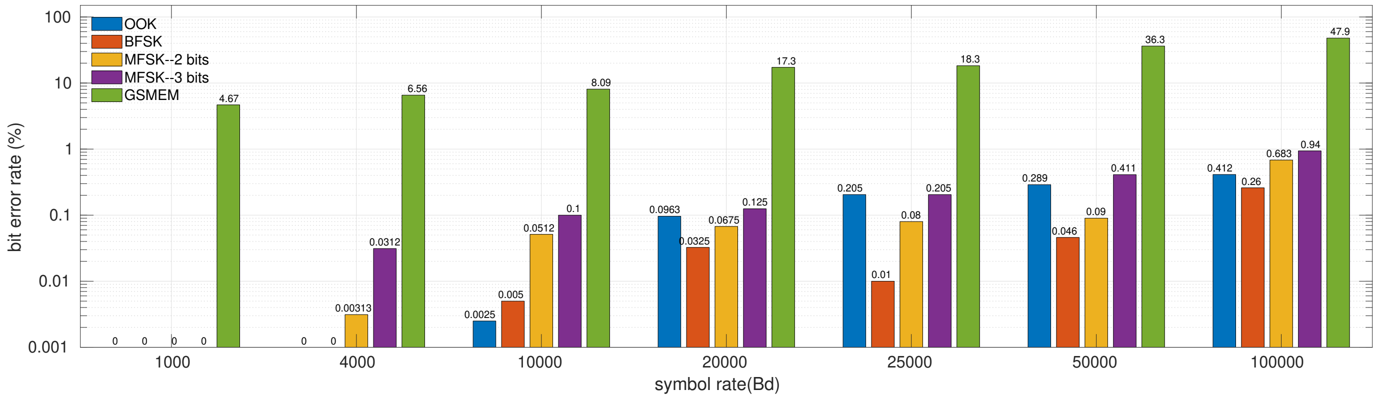


Fig. 7: Bit error rate at different symbol rate for *GSMem* and *BitJabber* using different modulation methods.

access frequencies are used and the resulting EM emanations almost affect the entire 25MHz frequency range. If more bits are transmitted, frequency ranges affected during transmission of different symbol values may overlap too much and variance between different symbol values' feature vectors tends to be smaller, which will also increase the error rate.

D. Through-Wall Evaluations

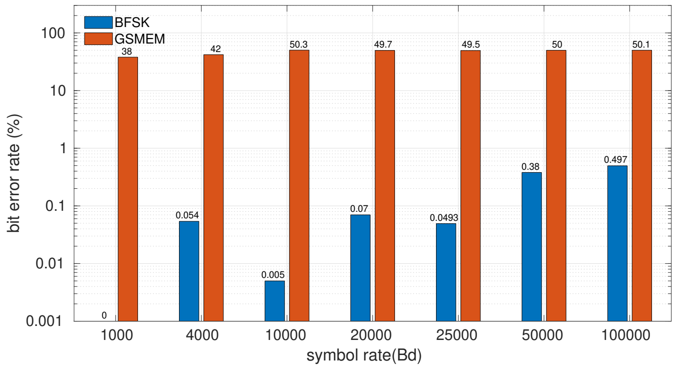


Fig. 8: Bit error rate of *GSMem* and *BitJabber* using the B-FSK modulation measured with a wall between the receiver and sender

Compared to the other covert channels, one advantage of EM covert channels is that EM signals can travel through many non-metal obstacles with little energy loss. In this experiment, *GSMem* and *BitJabber* are evaluated in a more practical scenario. The sender machine is put in an isolated room with a 15cm thick concrete wall. The distance between the sender and the wall is 50cm. The receiver is set in the next door sharing the same wall with the sender's room. Similar to the previous evaluation, background noise exists in both rooms and there are even some wire cables with unknown layout in the wall. In this scenario, the received EM emanations generated by the DRAM clock is weaker and more noise is in the transmission process. Wall-penetrating performance of *GSMem* and our *BitJabber* using the B-FSK modulation are evaluated and the results are shown in Fig. 8. From the figure, we can conclude that:

- Compared to results in Fig.7, performances of both covert channels get worse to some extent.
- *GSMem*'s performance is seriously affected and the error rate reaches 50% with symbol rate of only 10,000Bd.
- Performance of our *BitJabber* using the B-FSK modulation is only slightly affected, and at the fastest symbol rate 100,000Bd, the error rate is doubled but it is still lower than 0.5%.

E. Performance on the Second Platform

BitJabber is a covert channel that works on computers equipped with DRAMs of various frequencies. To prove that, in the following part, we repeat above experiments on the second platform where DRAM bus clock frequency is around 667MHz. The evaluation results of *GSMem* and our *BitJabber* are shown in Fig. 9 and the through-wall evaluation results are presented in Fig. 7. We can get similar conclusions that the error rate tends to increase with the symbol rate and our new *BitJabber* covert channel outperforms the *GSMem* method. Compared to the evaluation results in Fig. 7 and Fig. 8, both covert channels evaluated on this platform has better performance. Our observations indicate that the carrier's SNR in these evaluations is better due to following reasons:

1) Around 800MHz, some strong EM signal from unknown sources unrelated to computer's activity was observed. While around 667MHz, the strength of background noise is much lower.
2) Compared to Dell Optiplex 3020 model, Dell Optiplex 990 model generates stronger EM emanations and the shielding effect of metal case is weaker. The factors affecting the EM emanations are unclear, but we assume that they may be related to the design of motherboard, DRAM modules installed and shape of computer cases.

VI. Countermeasures

A general method of mitigating EM covert channel is shielding the air-gapped computer to eliminate or reduce the EM radiation. Since EM signal can travel through normal walls, metal shields like Faraday cage are needed to block the EM wave propagation. As reported in [33], EM emanations from

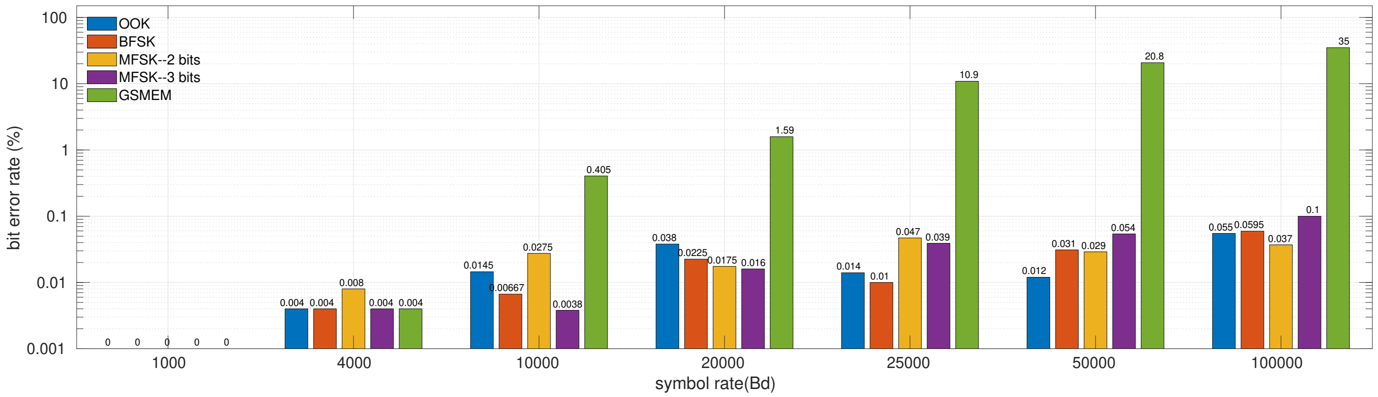


Fig. 9: Bit error rate at different symbol rate for *GSMem* and *BitJabber* using different modulation methods on the second platform

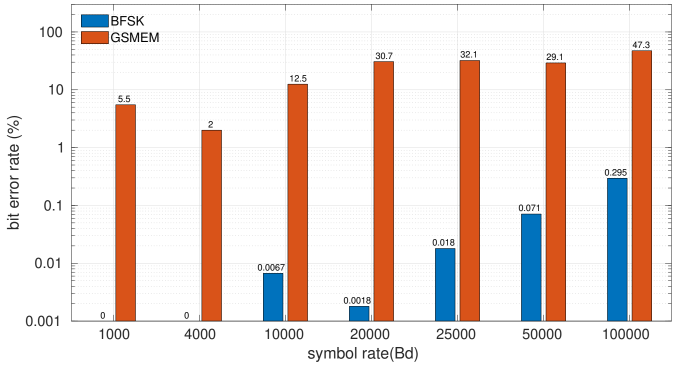


Fig. 10: Bit error rate of *GSMem* and *BitJabber* using the B-FSK modulation measured with a wall between the receiver and sender on the second platform

metal-shielded computers are weakened but not eliminated. The evaluations in this paper are performed with computers in metal cases, the longest distance where we can receive an exploitable signal is around 3 meters. However, if the metal plate on the case is replaced with tempered glass, strong EM signals from DRAM bus can be easily received even if the receiver is put more than 8 meters away from the victim computer. Furthermore, even for computers using metal cases, the shielding effect can vary a lot depending on the shape and size of the case. Therefore, the computer manufacturers can increase the difficulty of implementing EM covert channels by designing metal cases with good shielding effect.

Another commonly used mitigation is using signal interference devices to introduce more noise to reduce the carrier's SNR. However, our approach will disperse the power of random noise after de-spreading the EM signal generated by DRAM clock. To better mitigate this covert channel, the noise generator can produce noise with SSC pattern to disturb the de-spreading process.

Because performing memory activities in stable frequencies is important to implement *BitJabber*, running a protector program performing irregular memory accesses can make the sender program's memory access speed unstable and increase the error rate during transmission.

BitJabber's performance is highly dependent on the de-spreading of SSC signal. In most modern computers SSC is implemented by FM modulating the clock signal with a simple periodical signal. This despreading process can be easily reversed to recover the modulating signal. If we use more complicated SSC technique (e.g., using a secret random number sequence to FM modulate the clock signal), the attacker can not restore the high-SNR carrier and the implementation of *BitJabber* is much harder.

When the sender program is running, the carrier can be clearly identified in the spectrum. Therefore, this covert channel can be detected by using signal receivers to monitor the spectrum near the DRAM clock's frequency like the rowhammer detection technique proposed in [34].

VII. Conclusion

In this paper, the EM radiation of DRAM clock is exploited to implement a covert channel. We restore a high-SNR carrier by de-spreading DRAM clock's EM emanations and apply multiple modulation techniques to exploit the EM signals to exfiltrate data from air-gapped computers efficiently. The performance of our covert channel *BitJabber* is evaluated and compared with an existing covert channel *GSMem*, which exploited the same EM emanations from DRAM clock. *BitJabber* can reach bandwidth of 300,000 bps with error rate under 1% and it can also perform wall-penetrating data exfiltration. This covert channel greatly increases the maximum data exfiltration speed for air-gapped computers by exploiting EM side-channels, which may make people pay more attention on the protection against EM attacks.

Acknowledgment

This work is supported in part by the National Science Foundation (CNS-1739328). The authors would like to thank the anonymous reviewers for their comments and suggestions which help us improve the quality of the paper.

References

[1] Robert Callan, Alenka Zajić, and Milos Prvulovic. Fase: finding amplitude-modulated side-channel emanations. In *2015 ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA '15)*, pages 592–603, 2015.

[2] Brent Carrara and Carlisle Adams. On acoustic covert channels between air-gapped systems. In *International Symposium on Foundations and Practice of Security*, pages 3–16. Springer, 2014.

[3] Departments and agencies of the Federal Government. Code of federal regulations, 2019.

[4] Qian Ge, Yuval Yarom, David Cock, and Gernot Heiser. A survey of microarchitectural timing attacks and countermeasures on contemporary hardware. *Journal of Cryptographic Engineering*, 8(1):1–27, 2018.

[5] Mordechai Guri, Andrey Daidakulov, and Yuval Elovici. Magneto: Covert channel between air-gapped systems and nearby smartphones via cpu-generated magnetic fields. *arXiv preprint arXiv:1802.02317*, 2018.

[6] Mordechai Guri, Ofer Hasson, Gabi Kedma, and Yuval Elovici. An optical covert-channel to leak data through an air-gap. In *2016 14th Annual Conference on Privacy, Security and Trust (PST '16)*, pages 642–649, 2016.

[7] 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)*, pages 849–864, 2015.

[8] Mordechai Guri, Gabi Kedma, Assaf Kachlon, and Yuval Elovici. Airhopper: Bridging the air-gap between isolated networks and mobile phones using radio frequencies. In *2014 9th International Conference on Malicious and Unwanted Software: The Americas (MALWARE '14)*, pages 58–67. IEEE, 2014.

[9] Mordechai Guri, Matan Monitz, and Yuval Elovici. Usbee: air-gap covert-channel via electromagnetic emission from usb. In *2016 14th Annual Conference on Privacy, Security and Trust (PST '16)*, pages 264–268, 2016.

[10] Mordechai Guri, Matan Monitz, Yisroel Mirski, and Yuval Elovici. Bitwhisper: Covert signaling channel between air-gapped computers using thermal manipulations. In *2015 IEEE 28th Computer Security Foundations Symposium (CSF '15)*, pages 276–289, 2015.

[11] Mordechai Guri, Yosef Solewicz, Andrey Daidakulov, and Yuval Elovici. Fansmitter: Acoustic data exfiltration from (speakerless) air-gapped computers. *arXiv preprint arXiv:1606.05915*, 2016.

[12] Mordechai Guri, Yosef Solewicz, Andrey Daidakulov, and Yuval Elovici. Acoustic data exfiltration from speakerless air-gapped computers via covert hard-drive noise (diskfiltration). In *European Symposium on Research in Computer Security (ESORICS '17)*, pages 98–115, 2017.

[13] Mordechai Guri, Boris Zadov, Dima Bykhovsky, and Yuval Elovici. Powerhammer: Exfiltrating data from air-gapped computers through power lines. *arXiv preprint arXiv:1804.04014*, 2018.

[14] Mordechai Guri, Boris Zadov, Andrey Daidakulov, and Yuval Elovici. Odini: Escaping sensitive data from faraday-caged, air-gapped computers via magnetic fields. *arXiv preprint arXiv:1802.02700*, 2018.

[15] 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 *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA '17)*, pages 161–184, 2017.

[16] Michael Hanspach and Michael Goetz. On covert acoustical mesh networks in air. *Journal of Communications*, 8(11), 2013.

[17] Mohamed Hassan, Anirudh M Kaushik, and Hiren Patel. Reverse-engineering embedded memory controllers through latency-based analysis. In *21st IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS '15)*, pages 297–306, 2015.

[18] Butler W. Lampson. A note on the confinement problem. *Commun. ACM*, 16(10):613–615, October 1973.

[19] Arthur Costa Lopes and Diego F Aranha. Platform-agnostic low-intrusion optical data exfiltration. In *ICISSP*, pages 474–480, 2017.

[20] Joe Loughry and David A Umphress. Information leakage from optical emanations. *ACM Transactions on Information and System Security (TISSEC)*, 5(3):262–289, 2002.

[21] Ramya Jayaram Masti, Devendra Rai, Aanjhan Ranganathan, Christian Müller, Lothar Thiele, and Srdjan Capkun. Thermal covert channels on multi-core platforms. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 865–880, 2015.

[22] Nikolay Matyunin, Jakub Szefer, Sebastian Biedermann, and Stefan Katzenbeisser. Covert channels using mobile device's magnetic field sensors. In *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC '16)*, pages 525–532, 2016.

[23] Clémentine Maurice, Christoph Neumann, Olivier Heen, and Aurélien Francillon. C5: cross-cores cache covert channel. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA '15)*, pages 46–64, 2015.

[24] Peter Pessl, Daniel Gruss, Clémentine Maurice, Michael Schwarz, and Stefan Mangard. Drama: Exploiting dram addressing for cross-cpu attacks. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 565–581, 2016.

[25] Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In *Proceedings of the 16th ACM conference on Computer and communications security (CCS '09)*, pages 199–212. ACM, 2009.

[26] Vitali Sepetnitsky, Mordechai Guri, and Yuval Elovici. Exfiltration of information from air-gapped machines using monitor's led indicator. In *2014 IEEE Joint Intelligence and Security Informatics Conference*, pages 264–267. IEEE, 2014.

[27] Dean Sullivan, Orlando Arias, Travis Meade, and Yier Jin. Microarchitectural minefields: 4k-aliasing covert channel and multi-tenant detection in iaas clouds. In *NDSS '18*, 2018.

[28] Jakub Szefer. Survey of microarchitectural side and covert channels, attacks, and defenses. *Journal of Hardware and Systems Security*, 3(3):219–234, 2019.

[29] Zhenghong Wang and Ruby B Lee. Covert and side channels due to processor architecture. In *2006 22nd Annual Computer Security Applications Conference (ACSAC '06)*, pages 473–482. IEEE, 2006.

[30] Zhenyu Wu, Zhang Xu, and Haining Wang. Whispers in the hyper-space: High-speed covert channel attacks in the cloud. In *Presented as part of the 21st USENIX Security Symposium (USENIX Security 12)*, pages 159–173, 2012.

[31] Yuan Xiao, Xiaokuan Zhang, Yinqian Zhang, and Radu Teodorescu. One bit flips, one cloud flops: Cross-vm row hammer attacks and privilege escalation. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 19–35, 2016.

[32] Yunjing Xu, Michael Bailey, Farnam Jahanian, Kaustubh Joshi, Matti Hiltunen, and Richard Schlichting. An exploration of l2 cache covert channels in virtualized environments. In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop (CCSW '11)*, pages 29–40, 2011.

[33] Alenka Zajić and Milos Prvulovic. Experimental demonstration of electromagnetic information leakage from modern processor-memory systems. *IEEE Transactions on Electromagnetic Compatibility*, 56(4):885–893, 2014.

[34] Zhenkai Zhang, Zihao Zhan, Daniel Balasubramanian, Bo Li, Peter Volgyesi, and Xenofon Kousoukos. Leveraging em side-channel information to detect rowhammer attacks. In *2020 IEEE Symposium on Security and Privacy (S&P '20)*, May 2020.

[35] Zheng Zhou, Weiming Zhang, Zichong Yang, and Nenghai Yu. Exfiltration of data from air-gapped networks via unmodulated led status indicators. *arXiv preprint arXiv:1711.03235*, 2017.