MagView: A Distributed Magnetic Covert Channel via Video Encoding and Decoding

Juchuan Zhang
USSLab, Zhejiang University
juchuanzhang@zju.edu.cn

Xiaoyu Ji*
USSLab, Zhejiang University
xji@zju.edu.cn

Wenyuan Xu
USSLab, Zhejiang University
wyxu@zju.edu.cn

Yi-Chao Chen
Shanghai Jiao Tong University
yichao@sjtu.edu.cn

Yuting Tang
Huaxin Consulting CO., Ltd
tangyuting.hx@chinaccs.cn

Gang Qu
University of Maryland
gangqu@umd.edu

***Abstract*—Air-gapped networks achieve security by using the physical isolation to keep the computers and network from the Internet. However, magnetic covert channels based on CPU utilization have been proposed to help secret data to escape the Faraday-cage and the air-gap. Despite the success of such cover channels, they suffer from the high risk of being detected by the transmitter computer and the challenge of installing malware into such a computer. In this paper, we propose `MagView`, a distributed magnetic cover channel, where sensitive information is embedded in other data such as video and can be transmitted over the air-gapped internal network. When any computer uses the data such as playing the video, the sensitive information will leak through the magnetic covert channel. The "separation" of information embedding and leaking, combined with the fact that the covert channel can be created on any computer, overcomes these limitations. We demonstrate that CPU utilization for video decoding can be effectively controlled by changing the video frame type and reducing the quantization parameter without video quality degradation. We prototype `MagView` and achieve up to 8.9 bps throughput with BER as low as 0.0057. Experiments under different environment are conducted to show the robustness of `MagView`. Limitations and possible countermeasures are also discussed.**

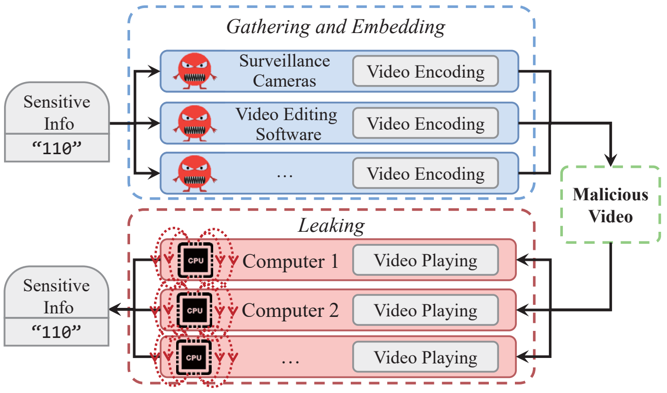


Fig. 1: The concept of separation in `MagView`. Top: sensitive information is embedded during video encoding. Bottom: sensitive information leaks through the magnetic covert channel created when video is played. "Malicious" video with sensitive information can be transmitted internally in the air-gapped network.

I. Introduction

Air-gapped networks are those private networks where the computers and other equipment are physically isolated without connection to outside public network such as the Internet. In addition to the communication on the private internal network [1], some air-gapped networks forbid the use of Wi-Fi, Bluetooth, and infrared [2] as well as the use of memory card [3], [4] to prevent data leakage. Thus, we have seen many security-aware organizations such as NSA and US Defense Intelligence Agency use air-gapped network as the infrastructure for their daily operations. However, they are not immune to breaches of covert channels [5], i.e., channels that are not intended for information transfer but may leak sensitive data, even with low signal-to-noise-ration (SNR) [6]. Common medium of a covert channel can be acoustic, ultrasonic, electromagnetic, thermal or optical [7]. However, with security enhancement, more and more existing covert channels like optical channels, acoustic channels, etc are being cut off [8].

Low frequency magnetic field, which is generated by the electric current in CPU modules, is a state-of-the-art covert channel as it can pass the Faraday-Cage and is difficult to detect. By regulating the CPU utilization, sensitive data is encoded into the changes of magnetic strength. Receivers such as magnetometers [9], smartphones [10] can receive and decode the magnetic signal to extract the leaked data. As CPU is an essential part of any computer, the covert channel can be implemented on desktop PCs, servers, laptops and even embedded systems.

Currently proposed magnetic-field-based covert channels (hereafter we name it magnetic covert channel) [9], [10], however, have two major limitations. First, they require direct regulation of the computer's CPU utilization to embed sensitive information, which can easily attract attention and be caught. Second, a malware has to be implanted on the

∗ Corresponding faculty author.

same computer for CPU utilization control and sensitive data exfiltration, which further limits its usage.

In this paper, we seek to enhance the practicability and stealthiness of the magnetic covert channel by (1) getting rid of implanting a malware on the very computer that is leaking sensitive data and (2) hiding direct CPU utilization regulation. To this end, we need physically decouple the embedding and leaking of sensitive information in order to implant the malware only where sensitive information is embedded; find a "carrier" which contains the embedded sensitive information to control CPU utilization in a stealth way during leaking.

We observe that video interfaces are ubiquitous in security-aware organizations, including videos from surveillance cameras and promotional videos, etc. Because video playing needs a decoding step, it can be a good candidate for CPU intensive operations[1]. This leads us to the idea of using video encoding to embed information and decoding to manipulate CPU utilization for information leakage. By doing this, the two requirements mentioned above can be satisfied. First, information embedding and leaking can be separated. Specifically, sensitive information embedding can be done by surveillance cameras or computers having video editing softwares if malwares are implanted on those devices. The leaking progress, on the other hand, can be on any devices playing the videos with embedded information, i.e., the malicious videos shown in Fig. 1. Second, since the CPU utilization regulation is covered by the video playing task, the magnetic covert channel becomes stealthy and difficult for people to notice. Fig. 1 depicts the overview of `MagView`. Sensitive information gathered by the the malicious surveillance camera, for example, can be embedded into malicious videos which can be played by any devices in the internal networks. A smartphone or a dedicated device with magnetic sensor placed next to the computer playing the video can pick up the magnetic signals and recover the sensitive data.

The design of `MagView` encounters several challenges. First, the re-encoded videos cannot be suspicious visually, i.e., the video content and quality such as resolution cannot be changed. Second, how to keep a high SNR for the magnetic cover channel with background application running on the devices is also a challenge. To cope with the above challenges, we carefully investigate H.264/AVC [11], a common video encoding standard, and find out that the frame type and the quantization parameter (QP) can control the size of a video frame, and thereby can affect the CPU utilization when decoding video frames. Such a strategy to increase CPU utilization is also validated on H.265 [12]. We also design the ASK modulation, DSSS-like encoding scheme, an adaptive CPU utilization control algorithm and use Forward Error Correction (FEC) to increase the robustness of the covert channel.

In summary, we have made the following contributions:

- We propose an enhanced distributed magnetic covert channel `MagView`, featuring the separation of data embedding and leaking, to exfiltrate sensitive data from Faraday-caged air-gapped networks.
- We implement `MagView` where data is embedded during video encoding by the selection of frame type and QP value, and the covert channel is created whenever the video stream is played without affecting video quality.
- We prototype `MagView` and perform comprehensive experiments. We employ a real surveillance video playing on 9 different computers using 8 different smart devices as receivers. Results show that we can achieve up to 8.9 bps throughput with BER as low as 0.0057. That means that it would only take 15 seconds to pass a 128-bit key.

II. Background

In this section, we first introduce the background knowledge of video encoding and decoding for the design to change CPU utilization. Then we provide the principle of how CPU module can generate magnetic signals and the relationship between CPU utilization and magnetic signal strength.

A. Video Encoding and Decoding

A video is composed of a sequence of frames, i.e., I frame, P frame and B frame in the H.264/AVC standard [13] and each frame can be viewed as a still image. I frame is encoded without reference, while P frame and B frame are encoded as the differences from a reference frame with motion prediction to reduce video size. Consequently, the size of I frame is larger than the other two.

To reduce the video size, compression is always performed on video frames, by going through the processes of discrete cosine transform (DCT), quantization and entropy encoding. The DCT step is similar to that in image compression, which is used to reduce special redundancy of an image. Quantization step is to map the DCT coefficients to a reduced range of values and thus it should be possible to represent the DCT coefficients with fewer bits [13]. Finally, entropy encoding step is to reduce the redundancy between the compressed data symbols using variable length coding techniques [14]. Among the steps, only the quantization step introduces signal loss and its parameter, i.e., quantization parameter (QP) directly determines the compression performance. Roughly speaking, a smaller QP leads to less efficient compression, higher bit rate (larger video size) and vice versa. QP value can be configured dynamically per frame.

A video file has three parameters: frame rate, resolution and bit rate, which are related to the video quality. For most videos, frame rate and resolution are constant, while bit rate can be variable caused by different frame types and QPs in different frames. Furthermore, each video file has a video coding format, like H.264 and VP8, and a container format, such as MP4, MKV and AVI. Video decoding has exactly the opposite process of video encoding and is implemented by video players, which decode videos in the unit of frame.

B. CPU Module and Magnetic signals

The dynamic power consumption during CPU execution can be estimated as [15]

$$P = a_t C_L V_{dd}^2 f_{clock} \tag{1}$$

[1]We focus on the software decoding and details are in Sec. VIII.

where $a_t C_L$ is the effective capacitance being switched to perform a computation, which is related to CPU utilization and the performed specific computation; V_{dd} is the operating voltage of the CPU and f_{clock} is the clock frequency, both of which are variable according to the dynamic frequency scaling (DFS) for energy saving [15]. The DFS policy decreases V_{dd} and f_{clock} with low CPU load and vice versa. Therefore, the power differences between busy and idle states of CPU are determined by both effective switched capacitance and the voltage and frequency scaling caused by DFS. In other words, when the CPU is busy, i.e., the CPU utilization is high, it gains more power consumption than when the CPU is idle.

The total CPU module can be seen as a magnetic dipole. For the sake of simplicity, the magnetic field generated by CPU module can be represented as $B \propto \frac{I}{r^3}$, where B is the magnetic induction intensity, I is the total current in the CPU module, and r is the distance to the CPU module. Combining the above equations with $P = V_{dd} I$, we have $B \propto \frac{a_t C_L V_{dd} f_{clock}}{r^3}$. As $a_t C_L$, V_{dd} and f_{clock} are all positively correlated with CPU utilization, we conclude that the magnetic induction intensity of CPU magnetic field is strongly correlated with CPU utilization.

III. Threat Model and Overview

A. *Threat Model and Assumptions*

The attack scenario and main assumptions are similar to those reported in the literature [9], [10], where the attacker's goal is to exfiltrate sensitive data from air-gapped networks. More specifically, we make the following assumptions:

- In the air-gapped network, all wireless communication interfaces are disabled and the network is physically separated from public networks. Storage devices such as USB flash drives are banned.
- A malware for sensitive data gathering and video encoding has been implanted in advance by the attacker, and the implantation can be referred to existing covert channel solutions [16], [17].
- The attacker can get close to a computer playing malicious videos and have his/her smartphone close to the chassis of the computer or on the laptop's keyboard to receive the sensitive data.

Among these items, the first one is met by most of the air-gapped networks as we have discussed in Sec. I. For the second one, we assume the malware can be implanted into a video encoder of a surveillance camera or in a video editing software installed on computers during manufacturing or updating. More details on this can be done is out of scope of this paper and can be found in [16], [17]. The last assumption is necessary and common due to the short distance of the magnetic covert channel.

B. `MagView` *Design Overview*

Similar to existing magnetic covert channels [9], [10], `MagView` is based on the magnetic field during CPU execution. The novelty of `MagView` is how it manipulate CPU utilization. As we have mentioned earlier, the malware will create "malicious" video which when being played, will have specific CPU requests and impact the magnetic field so the receiver can extract the sensitive data from the cover channel.

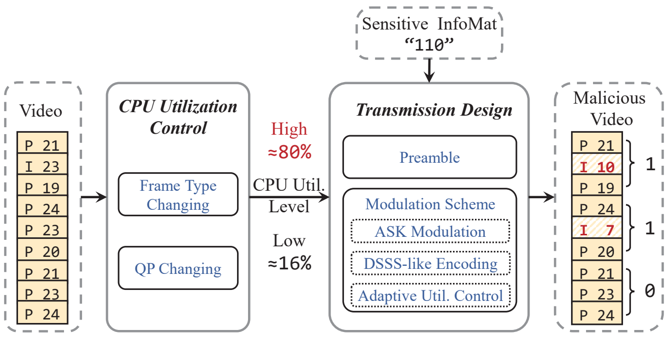


Fig. 2: Technical block diagram to generate a malicious video. For an original video, `MagView` first determines and changes the frame type and QP of each frame to achieve two different CPU utilization levels, i.e., "High (80%)" and "Low (16%)" and then use the two levels to embed and modulate the sensitive information into video frames. (B frames are omitted for simplicity but are also applicable.)

Fig. 2 illustrates the how sensitive data is embedded into the video to create the "malicious" video. In the CPU utilization control step, both frame type changing and QP changing are used to achieve two different CPU utilization levels. Then in the transmission step, ASK modulation, DSSS-like encoding and adaptive modulation with preamble are used to modulate the sensitive data on video frames with the two CPU utilization levels. The malicious video is then delivered to computers or laptops (here we call them transmitters) which will play the video. When the video plays on any computer, the sensitive data leaks from the CPU magnetic field and can be picked up by a device with a magnetometer.

In the following, we elaborate CPU utilization regulation and embedding schemes respectively in Sec. IV and Sec. V.

IV. Video Decoding to Change CPU Utilization

In Sec. II, we conclude that both frame type and QP determine the bit rate at the granularity of frame. Therefore, it is possible to change the CPU utilization of video decoding by changing frame type and QP. Although frame rate and resolution can also affect the bit rate, they are not supported to be configured and changed sometimes. As a result, we resort to both frame type and QP and incorporate them into a systematic approach to quantitatively output a target CPU utilization.

A. *Changing CPU Utilization*

1) Changing Frame Type: As is discussed in Section II, size of I frames is larger than P and B frames. Therefore, I frames are avoided and P, B frames are preferred by default during the encoding process, unless necessary. As a result, the number of I frames is relatively smaller than that of P, B frames. This provides us the chance to modify a P/B frame to an I frame to gain higher CPU utilization.

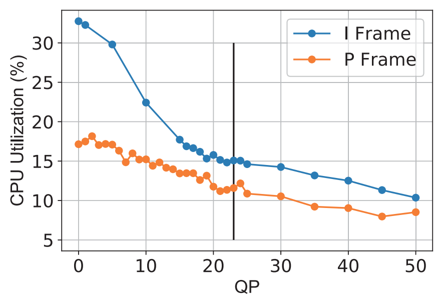


Fig. 3: CPU utilization vs. QP.

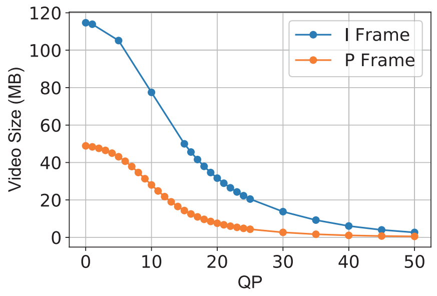


Fig. 4: Video size vs. QP.

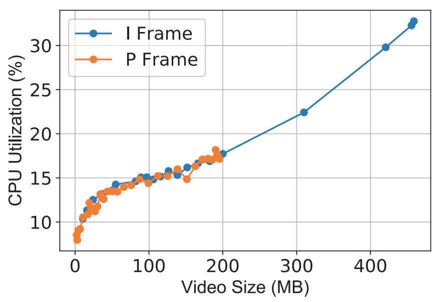


Fig. 5: CPU utilization vs. video size.

2) Changing Quantization Parameter (QP): It is mentioned in Section II that smaller QP leads to less signal loss and higher bit rate, and thus the CPU utilization during frame decoding increases. With the requirement to keep the original video quality, QP has to be less than a specific value that is the largest QP to keep the original quality. Under this condition, CPU utilization can be increased without influencing the original video quality by decreasing QP.

B. Quantitative Validation

1) Settings: To quantify the performance of changing frame type, we re-encode a 1-min video with X264 [18]. The original video is composed of 14 I frames, 572 P frames and 1214 B frames respectively. We use `--qpfile` to change frame types and `--crf` to activate CRF (Constant Rate Factor) mode to guarantee the video quality [19]. For convenience, we name the scheme by changing all frames to I/P frames *scheme-I* and *scheme-P*. We use GOM Player [20] running on a PC (i5-4200U CPU [21] with 2 cores and 4 threads, 8G RAM, Windows 10 17134.1) to play the re-encoded video and record the CPU utilization and video file size.

2) Frame type vs. CPU utilization: The result of *scheme-I*, *scheme-P* and original video is shown in Table I. The video encoded using scheme-I is all composed of I frames while the video encoded using *scheme-P* is all composed of P frames except individual I frames as necessary reference frames. The original video consists of I, P and B frames. We can find that: (1) *scheme-I* gains around 3.4% higher CPU utilization than *scheme-P*, and there is no significant difference in CPU utilization between *scheme-P* and the original video; (2) the video size increases in both cases, and *scheme-I* has a much larger increment. Therefore, we conclude that changing frames to I can both increase CPU utilization and video size, while changing frames to P only increases a little video size and has almost no effect on CPU utilization. Consequently, changing frame type from P or B to I is a feasible way to increase CPU utilization while decoding but the amount of change is limited.

3) QP vs. CPU utilization: We quantify the relationship between QP and CPU utilization under both *scheme-I* and *scheme-P*, which is shown in Fig. 3. Under CRF mode, if the QP value of frame i is not specified, it will be set as QP^{i}_{crf} according to the CRF mode parameter (default 23). Assuming that there are N frames in the video, the average QP of the video under CRF mode is

$$QP^{avg}_{crf} = \frac{1}{N}\sum_{i=1}^{N} QP^{i}_{crf} \tag{2}$$

which is denoted by a vertical line in Fig. 3. Note that the video quality will not be affected only when QP is less than QP^{i}_{crf}. We vary QP from 0 to 50, and the result shows that CPU utilization increases nearly in a linear way for *scheme-P*. While for *scheme-I*, the linear relationship between QP and CPU utilization appears separately when $QP > QP^{avg}_{crf}$ and $QP \leq QP^{avg}_{crf}$. Moreover, QP modification under *scheme-I* brings larger CPU utilization change when $QP \leq QP^{avg}_{crf}$.

TABLE I: The average CPU utilization vs. I/P frame changes.

Frame Type	*scheme-I*	*scheme-P*	Original Video
Average CPU Utilization (%)	11.76	8.36	8.34
Total Video Size (MB)	60.83	12.06	9.32

4) CPU utilization vs. video size: It is worth mentioning that decreasing QP brings about video size increase under both *scheme-I* and *scheme-P* frame scenarios. For example, the original 9.32 MB video can be increased by several times. Comparing Fig. 4 to Fig. 3, we find that though I frame gains more significant CPU utilization change by changing QP, it is at the cost of a larger video size. Fig. 5 depicts the relationship between video size and CPU utilization. The conclusion is that CPU utilization is fundamentally determined by video size, i.e., video bit rate, and no significant difference exists between I frames and P frames.

5) Algorithm to Change CPU Utilization: Without loss of generality, denote the CPU utilization under *scheme-P* and *scheme-I* $U_P(qp)$ and $U_I(qp)$ when $QP = qp$. Obviously $U_I(qp)$ should be larger than $U_P(qp)$. However, whether $U_I(QP^{i}_{crf}) < U_P(0)$ or not is uncertain and thus we have two cases:

$$U_P(QP^{i}_{crf}) < U_I(QP^{i}_{crf}) \leq U_P(0) < U_I(0) \tag{3}$$

$$U_P(QP^{i}_{crf}) < U_P(0) < U_I(QP^{i}_{crf}) < U_I(0) \tag{4}$$

Algorithm 1: CPU utilization adjustment algorithm.

```
Input: $U_{design}$
Output: $FrameType, QP$ for encoding
if $U_{design} < U_P(QP^i_{crf})$ or $U_{design} > U_I(0)$ then
    if $U_{design} < U_P(QP^i_{crf})$ then
        $FrameType \Leftarrow P$; $QP \Leftarrow QP^i_{crf}$;
    else
        $FrameType \Leftarrow I$; $QP \Leftarrow 0$;
    end
else
    if $U_{design} \geq U_I(QP^i_{crf})$ then
        $FrameType \Leftarrow I$; $QP \Leftarrow U_I^{-1}(U_{design})$;
    else
        if $U_{design} \leq U_P(0)$ then
            $FrameType \Leftarrow P$; $QP \Leftarrow U_P^{-1}(U_{design})$;
        else
            $FrameType \Leftarrow I$; $QP \Leftarrow QP^i_{crf}$;
        end
    end
end
```

The case denoted by Eq. 3 is shown in Fig. 3, and Eq. 4 is the other case where the CPU utilization of I frame and P frame are not overlapped when $QP < QP^i_{crf}$.

Given a designed CPU utilization as U_{design}, the frame type change and QP value decision for encoding a frame can be calculated as Alg. 1:

First, U_{design} is compared to $U_P(QP^i_{crf})$ and $U_I(0)$. If $U_{design} < U_P(QP^i_{crf})$, then we use *scheme-P* with QP^i_{crf} to ensure that the video quality does not decline. If $U_{design} > U_I(0)$, which means that U_{design} is beyond the maximum CPU utilization we can reach, the frame will be encoded using *scheme-I* with $QP = 0$.

If the above conditions are not met, U_{design} is compared to $U_P(0)$ and $U_I(QP^i_{crf})$. If $U_{design} \geq U_I(QP^i_{crf})$, then we use *scheme-I* with $QP = U_I^{-1}(U_{design})^2$. Otherwise, if $U_{design} \leq U_P(0)$, then we use *scheme-P* with $QP = U_P^{-1}(U_{design})$. If not, i.e., the case of Eq. 4 appears and $U_P(0) < U_{design} < U_I(QP^i_{crf})$, we prefer to use *scheme-I* with QP^i_{crf} as I frame benefits the video playing.

V. Transmission Design

In this section, the data frame design and data modulation scheme are introduced. The data frame consists of preamble, payload followed by the FEC (Forward Error Correction) code. The preamble field is used for synchronization and parameter tuning. Hamming FEC code [22] is added on the payload to reduce BER.

A. Preamble Design

Preamble is used to synchronize the receiver with the sender. For synchronization, a template is generated and cross-correlation is performed on the received magnetic signal on all X, Y, Z axis respectively. The axis with the highest correlation coefficient peak is used for synchronization. Besides synchronization purpose, in `MagView` preamble also serves as parameter tuning for receiver to set the demodulation threshold B_{th}, which is an important parameter in the ASK modulation and introduced later.

Intuitively, the length of preamble should guarantee stable synchronization and accurate parameter estimation. In `MagView`, we empirically investigate and use a 37-bit-long preamble followed by a 300-bit payload. The designed preamble and parameter tuning process can be found in Fig. 6.

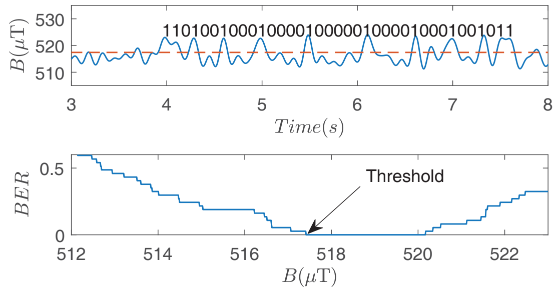


Fig. 6: Threshold tuning when receiving a preamble. B_{th} with minimum preamble decoding BER is chosen.

B. Data Frame Modulation

With a data frame consisting of the preamble, the payload and FEC, we now introduce how to modulate the data frame on CPU utilization changes.

1) ASK Modulation: Common digital modulation schemes include amplitude-shift keying (ASK), frequency-shift keying (FSK) and phase-shift keying (PSK). ASK uses different amplitudes to represent digits (or symbols), i.e., a high-amplitude signal represents "1" and a low-amplitude signal for "0". `MagView` employs ASK modulation due to this property.

Specifically, we use 2-ASK which is the simplest ASK for robust data transmission facing the weak magnetic signal with ambient inference. In 2-ASK, we define two levels of CPU utilization for each frame: the low-level U_l and the high-level U_h. U_l equals to the CPU utilization under *scheme-P* (all other frames are transformed into P frames) with QP automatically assigned by encoder, i.e., $U_P(QP^i_{crf})$. For U_h, we let $U_h = \alpha U_l$. In our implementation, we simply set $\alpha = 5$ to ensure sufficient discrimination between the magnetic signal emitted by CPU module under U_h and U_l. At the receiver side, utilization is decided by comparing with a threshold value of magnetic induction intensity of the CPU module, i.e., B_{th}.

2) DSSS-like Bit Encoding: With the two levels U_l and U_h, we can simply encode "1" and "0" on the two levels. However, in practice, this is error-prone because CPU utilization changes need a response time and it cannot change sharply. Therefore, to minimize error and enhance robustness, we employ a DSSS-like[3] bit encoding scheme. We encode a single bit with a

[2] $U_I^{-1}(\cdot)$ is the inverse function of $U_I(\cdot)$.

[3] The word DSSS (Direct Sequence Spread Spectrum) means to encode a symbol (4 bit) onto a 32-bit long sequence in 802.15.4 standard.

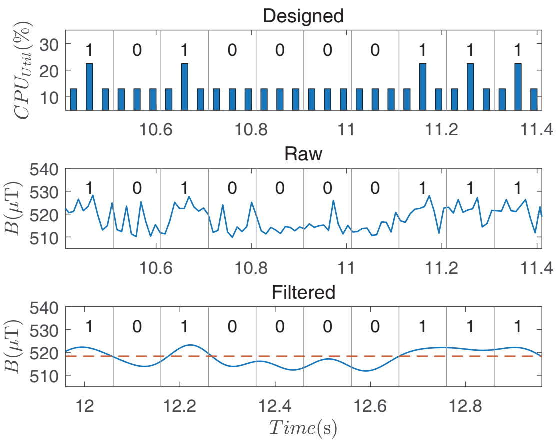


Fig. 7: DSSS-like bit encoding scheme. "*low-high-low*" means "1" and "*low-low-low*" stands for "0".

number of sequential changes in CPU utilization. Define the number of frames that represents a single bit as T_B, hereafter we name it code element length.

In `MagView`, $T_B = 3$, i.e., 3 frames represent a single bit, and we use "*low-high-low*" to encode "1" and "*low-low-low*" for "0". Low CPU utilization is preferred to keep the covert channel stealthy. Fig. 7 shows an example of encoding bit with the DSSS-like bit encoding scheme. The second row of Fig. 7 is the received raw signal shown in the first row. Note that the high frequency noises exist and a low pass filter should be used to filter out the high frequency noise. In our implementation, we choose to use a finite impulse response low pass filter with passband cutoff frequency $f_p = 2$ and stop band cutoff frequency $f_s = 3$. The signal after filtering is depicted in the bottom row of Fig. 7.

3) Adaptive Utilization Control: If the malware can get the specific CPU model of the computer which plays the malicious video, an adaptive utilization control method can be used to further improve the performance of `MagView`. Essentially, `MagView` makes use of the CPU utilization margin to embed information. As a result, the available margin is limited by the capacity of CPU, the video itself (e.g., format and size) as well as background applications that use CPU. Let U_{back} denote the sum of CPU utilization of the background applications and U_{video} stand for CPU utilization of the video without re-encoding, then the margin of available CPU utilization we can use is:

$$U_{margin} = 100\% - U_{back} - U_{video} \tag{5}$$

To cope with the dynamic conditions brought by videos and background workload, we design an adaptive utilization method during modulation. To be specific, if U_{margin} is expected to be lower than a certain value, the transmission, i.e., embedding data into video frames, will be terminated as the CPU utilization margin cannot support U_{design}. Otherwise, the encoder will choose an appropriate α to calculate U_h. Then the encoder uses Alg. 1 to calculate the frame type and QP to derive the designed CPU utilization U_{design}.

Actually, it is infeasible to estimate U_{back} and we can only estimate U_{video} according to Fig. 3 with the knowledge of QP and assumptions about CPU types. The background CPU utilization U_{back} is assumed to be a constant value. This is reasonable in some scenarios such as surveillance camera system where the computer mainly runs a displaying task.

4) Throughput Analysis: The principle of `MagView` is to manipulate the computation of video frames to increase the resulting CPU utilization when decoding. As `MagView` encodes bits in the granularity of frame, thus its maximum transmission speed is limited by the frame rate of a video, denoted as FPS.

Transmission speed of `MagView` is also decided by the number of available levels, denoted as N_{level} as well as the DSSS-like encoding scheme, and we formulate it as:

$$Speed = \frac{FPS \times log_2 N_{level}}{T_B} \tag{6}$$

For example, in the current implementation, N_{level}=2, T_B=3 and FPS=30, the expected transmission speed is 10 bps. With a 37-bit long preamble and a 300-bit long payload, the expected throughput is 8.9 bps. In section. VII, we demonstrate that `MagView` can achieve this value with only 0.0057 BER even when FEC is disabled.

VI. Receiver Design

At the receiver side, signal pre-processing, preamble detection and parameter tuning (magnetic induction intensity threshold B_{th} for decoding) are designed.

The received magnetic signal is first pre-processed by a low pass filter with a slide window to filter out the high frequency noises. Then cross-correlation is conducted between the filtered signal and the template along all three axes. The axis with the highest correlation coefficient is chosen as the axis of the covert channel signal.

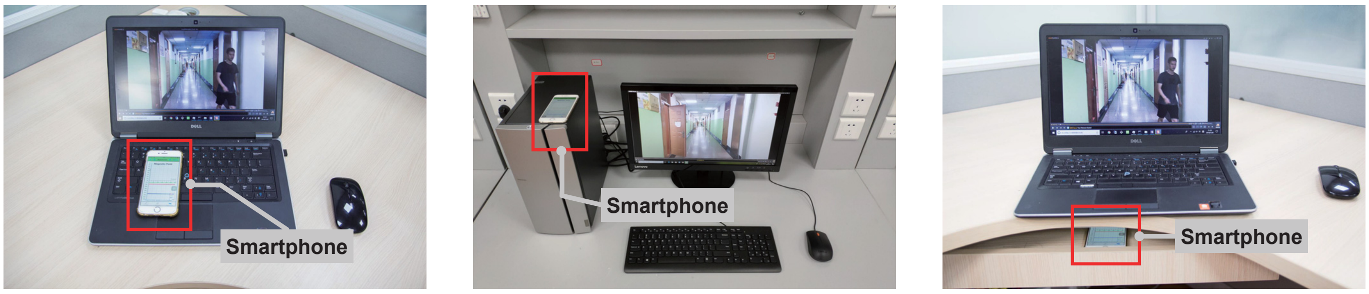
Besides, parameter tuning is performed to derive a proper B_{th}. Specifically, we increase threshold value and decode the preamble signal. With the increase of tested thresholds, the resulting decoding BER is first decreasing to a minimum value (e.g., 0) and then keeps stable and then increases, as is shown in Fig.6. Empirically, we choose the threshold value with minimum BER.

VII. Evaluation

In this section, we first prototyped `MagView` to test its overall performance, and then evaluated the impact of different factors.

A. Experiment Setup and Performance Summary

We utilize a real surveillance video [24] downloaded from Youtube. The video is with 1920x1080 resolution, 30 fps and 1642 kbps bit rate. For simplicity, we used x264 [18] to re-encoded the video offline to embed sensitive information. It is possible to online encode the video on a surveillance device



(a) Laptop-smartphone. (b) PC-smartphone. (c) Smartphone under desk.

Fig. 8: Experiment setup under different scenarios. A video with embedded data is played on laptops or desktop PCs. The magnetic signal emanated from the CPU module is collected by a smartphone with its built-in magnetometer.

TABLE II: The fundamental settings in the experiments.

Settings	Description
Video reproduction tool	FFMPEG [23]
Encoder	X264 [18]
Original video	A 1920x1080 30 fps surveillance video
Re-encoded video	Malicious: with 1.5 Kb data embedded
Video frame rate	30 fps
Operating system	Windows 10 17134.1
Sensor sampling rate	100 Hz
Video player	GOM Player [20]
Code element length	3 frames
Transmitter	DELL e7440
Receiver	iPhone 6 with its built-in magnetometer
Metric	BER without using FEC

as the hardware performance continues to improve. During the re-encoding process, U_l and U_h were 16.325% and 81.8% respectively to ensure sufficient discrimination between the magnetic signal emitted by CPU module under U_h and U_l. We embedded 1.5 Kb data into the video, and the bit rate increased to 15130 kbps consequently. The code element length T_B was set to 3 for robust transmission. We used an iPhone 6 with its built-in magnetometer to collect the magnetic signals from a Dell E7440 laptop with Intel i5-4200U Processor, as shown in Fig. 8a.

Results. At the receiver side, after demodulation and decoding, we calculated statistically the bit error of the transmitted data. Results show that `MagView` can achieve the theoretical 8.9 bps with 0.0057 BER even we disable FEC, which means that it takes only 15 seconds to transfer a 128-bit key.

B. *Impact of Different Settings*

In this subsection, we evaluated `MagView` in different settings, including **background applications, transmitters, receivers, sender-receiver distances and surroundings**. Unless otherwise stated, the experiment setup in Tab. II was used in all experiments. We also used a video taken by iPhone 7P in the corridor with the same 1920x1080 resolution, 30 fps as the surveillance video from Youtube [24] and 10594 kbps bit rate

TABLE III: Average CPU utilization and BER vs. different background applications.

APP	U_{total} (%)	U_{player} (%)	U_{back} (%)	BER
None	21.56	19.56	2.00	0.0005
Chrome	30.12	21.30	8.82	0.0435
Word	24.14	20.08	4.06	0.0090
MSTSC	24.62	20.24	4.38	0.0238

considering that the Youtube video is compressed by Youtube and therefore it is different from its original version. The metric we focused on was BER instead of transmission speed as a cover channel and therefore the following experiments were all revealed by BER without FEC.

1) Background Application: In this experiment, Chrome, Word and Microsoft Terminal Server Connection (MSTSC) were used as background applications considering they are common working applications in an office computer. In "None" case, the video player was the only running application. In "Chrome" case, ten tabs of different news sites were opened. In "Word" case, five Word windows were opened and each contains at least one page of content. And in "MSTSC" case, the computer as the transmitter was connected to another computer by MSTSC. We used *psutil* [25] to record the total CPU utilization U_{total} and the CPU utilization of the video player U_{player}, then the CPU utilization of background application was $U_{back} = U_{total} - U_{player}$. Tab. III shows the results. The BER increases a little when there is some background application. However, compared to the situation where there is no background application, the performances are still good with BER all lower than 0.05, which is in line with the expectation of experiment. Actually, as the video player is in the front window, the background applications are in idle state with low CPU utilization, which will not significantly increase the BER.

2) Transmitter: We used 9 different computers as transmitters to test their influence to BER, which were DELL e7440 (i5-4200U), DELL xps14 (i7-3537U), DELL xps13 (i5-6300U), Lenovo g40 (i5-5200U), Lenovo zhaoyang g42-80 (i3-7100U), Lenovo r720 (i5-7300H), Dell inspiring 14 (i5-8250U), PC1 (i5-8400), PC2 (i5-3470T) respectively. The experimental setup of PC1 is shown in Fig. 8b. We respectively

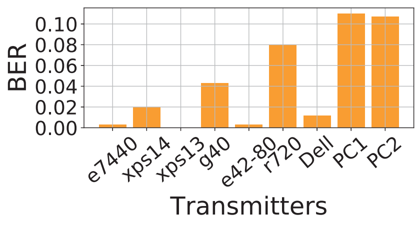


Fig. 9: BER vs. different transmitters.

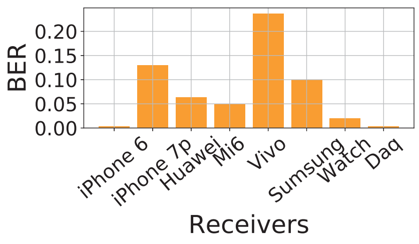


Fig. 10: BER vs. different receivers.

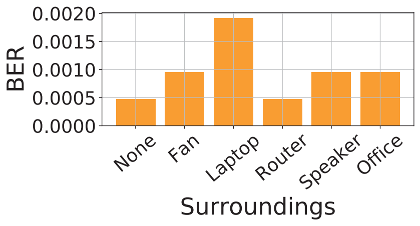


Fig. 11: BER vs. different surroundings.

found the relatively better location to set the receiver so that the receiver could record strong signals for each computer. The results are shown in Fig. 9. Except the two desktop computers PC1 and PC2, the BERs for all other computers are lower than 0.1. One of the explanations of the desktop case is that the distance from the receiver to CPU is larger than that of the laptop cases. Nevertheless, the BER of the two desktop PCs can be reduced to below 0.03 by using Hamming FEC with alphabet size $r = 2$, at the cost of a bit rate decrease to 3.0 bps.

3) Receiver: We used 6 smartphones, 1 smart watch and a data acquisition (DAQ) device [26] connected with a low-cost DRV425 [27] magnetic sensor as the receivers to record the magnetic signals. The DAQ device gains high sampling rate (200 kHz) than others. The results are shown in Fig. 10. Except iPhone 7P and Vivo, smartphones work well with the BER lower than 0.1. The reason why Vivo has a poor performance, as we infer, is that the sampling points are uneven. It is clear that DAQ receiver demonstrates the lowest BER due to its high sampling rate. In addition, Huawei Watch 2 has a good performance with the BER of 0.02, which shows the feasibility of using a smart watch to launch an attack.

4) Sender-receiver Distance: As we mentioned above, distances between the receiver and the CPU can make a difference to the results, so we put the receiver (the iPhone 6 or DAQ with DRV425 magnetic sensor) at different distances from the transmitter where the malicious video is played. The results are shown in Fig. 12, where the blue line is the result of iPhone 6 and the orange line is DAQ. Both receivers show low BER when the distance is below a value, say 6 cm with BER lower than 0.1. The BER of DAQ rises slightly slower than that of iPhone 6 as the distance increases. Moreover, we put the iPhone 6 under the transmitter computer separated by a wooden shelf as shown in Fig. 8c, in which situation the distance between the iPhone 6 and the bottom of the transmitter is about 4 cm with 0.065 BER. This illustrates that wooden shielding has little influence on magnetic signals. Even though the distance is relatively short in the current implementation, we believe it can be extended by dedicated device with more sensitive sensor.

5) Surroundings: To investigate the performance in real application scenarios, we tested `MagView` in six different surroundings as shown in Fig. 13, including (a) no adjacent device, (b) a fan nearby, (c) a laptop nearby playing a video, (d) a router nearby, (e) a speaker nearby and (f) a real office scenario with a desktop computer under desk and an air conditioner above. The BERs in all scenarios are no more than 0.003 as shown in Fig. 11, which means there is no significant effect of adjacent devices on the BER of `MagView`. The reason is that the strength of the low frequency magnetic signals is inversely proportional to the distance ($1/r^3$) from the device [28], which leads to little impact of the surrounding devices.

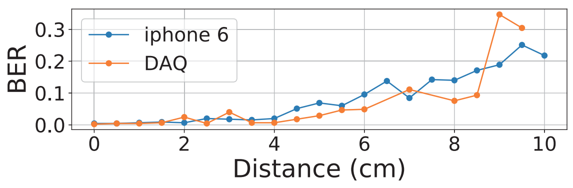
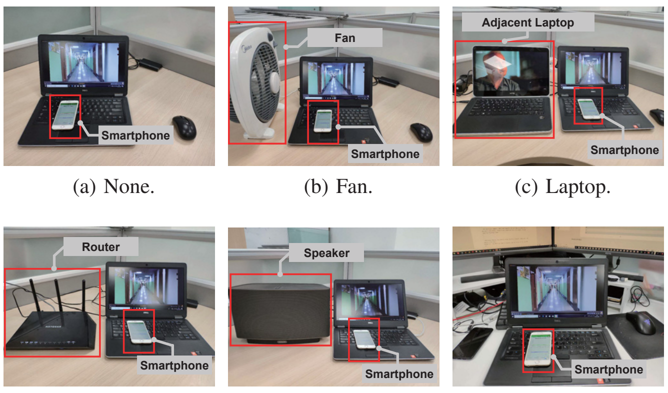


Fig. 12: BER vs. different distances.



(a) None. (b) Fan. (c) Laptop.

(d) Router. (e) Speaker. (f) Office.

Fig. 13: Experiment setup for different surroundings. An electric appliance is placed next to the receiver, i.e., the smartphone, to test the impact of surrounding devices on BER.

VIII. Discussion

A. Countermeasures

`MagView` can be defended in several ways. The simplest way is to re-encoding the videos, but this will result in extra computational overhead. Besides, an organization can do the followings.

1) Shielding and Physical Isolation: Security-aware organizations may shield the high secure computers from emitting electromagnetic signals. For instance, a Faraday cage can prevent the leakage of electromagnetic signals emanating from various computer parts including the CPU, memory and other parts. However, the signal of the magnetic covert channel is low frequency magnetic signal and can penetrate the Faraday cage [9], [10]. Consequently, the security-aware organizations should shield the computers with thicker metal surfaces [29] as well as extend the distance. Also, they can physically isolate the computers to eliminate physical access from attackers to receive the magnetic signals.

2) Anomaly Detection: Anomaly detection system can be used to detect abnormal operation of computers. Common anomaly detection systems use both software-based [30] and side-channel-based [31]–[33] detection to monitor the CPU workload or network traffic. In principle, `MagView` can be stealthy as it hides information in a natural task of video decoding. Moreover, as the video traffic will increase when moving objects are in the video, it is difficult to detect `MagView` by network traffic. More powerful agents may exploit specially-designed, machine-learning-based classification that models video decoding process. Under this circumstance, the minor CPU changes and network traffic may be detected.

B. Limitations

As low SNR and low data rates are normally the characteristics of covert channels [6], the 8.9 bps data rate of `MagView` is acceptable. Apart from this, there are several limitations of `MagView`. Firstly, the transmitter-receiver distance is limited. Under the settings in Sec. VII, we achieve 0.1 BER at 6 cm and 8.5 cm using iPhone 6 and DAQ device respectively. The distance is actually short for practical attacks unless the attackers can get very close to the attacked computers. Nevertheless, we envision that larger distance can be achieved by devices with more powerful magnetic sensors. Besides, we can increase the CPU change to enlarge the transmitted signal strength. Secondly, `MagView` increases video size which occupies more storage and network bandwidth. We prepare to find a way to change CPU utilization without video size increasing in future work. Thirdly, `MagView` should fail with video players using hardware-decoding by default as computation is on GPU instead of CPU. However, there are still many video players using software-based decoding by default as software-based decoding can provide better video playback quality and compatibility.

IX. Related Work

Covert channel is defined as the channel that is not intended for information transfer at all but leaks sensitive data [5]. Common covert channels can be divided into four categories: acoustic covert channels, electromagnetic ones, thermal ones and others. Radio frequencies emitted from video card is utilized to bridge the air-gap between isolated networks and mobile phones [1]. And for two air-gapped computers, the thermal produced by CPU [34] and the hard drive noise [35] are used to establish covert channels. Liu et al. [32] use power side-channel to monitor code execution, which can also be exploited as a covert channel. Besides, the GSM frequencies generated by memory-related instructions of a computer can be used to transmit, and the signals can be received by a nearby cellular phone [36]. For computer-smartphone channel, low frequency magnetic signal emanated from CPUs of desktop computers or laptops can be captured by magnetometers on mobile devices for communication, which can be seen as a covert channel [9], [10], [37]. Magnetic cover channel is also reported on hard drive [38]. In addition, the authors in [39] achieve a cover channel by controlling the impedance of the devices' wireless network interface card (NIC). Existing video covert channels are based on video-camera channel [40], [41], which can be as fast as 120 kbps. Nevertheless, as cameras are generally forbidden in secure aware organizations, screen-camera covert channels are limited.

It is worth mentioning that Matyunin et al. [42] proposed an inner device covert channel that an attacker changes video frame type and resolution to control the magnetic field generated by CPU, and therefore by using built-in magnetometer to achieve an App-to-App covert channel. Such work is similar to ours but is different in the following aspects:

- The attack scenario is different. Our scenario is to exfiltrate sensitive information from an air-gapped network while their work is an inner-device covert channel.
- Besides frame type, we change QP instead of video resolution to control the CPU utilization of video playing, which is stealthier and can support more video forms like video files.
- We did an extensive evaluation including different background Apps, transmitters, receivers, distances and surroundings.

X. Conclusion

In this paper, we present a novel magnetic covert channel via CPU magnetic field. Instead of controlling the CPU workload directly, `MagView` utilizes video as a media to embed, transfer and finally leak the sensitive data via CPU magnetic field. Therefore, the sensitive data gathering and embedding step and leaking step can be decoupled. `MagView` is stealthy as it hides CPU utilization changes in video decoding task, having no influence on the original video images. We design a CPU utilization adjustment algorithm, an adaptive ASK modulation to modulate data frame. We evaluate `MagView` under various settings including device types, distances, background APPs and surroundings. `MagView` achieves up to 8.9 bps throughput with BER as low as 0.0057.

Acknowledgment

This work was supported by China NSFC Grant 61702451, China NSFC Grant 61925109, ZJNSF Grant LGG19F020020, the Fundamental Research Funds for the Central Universities 2019QNA4027 and Alibaba-Zhejiang University Joint Institute of Frontier Technologies. Gang Qu's work was done when he visited Zhejiang University as a Qiushi Chair Professor.

References

[1] M. Guri, G. Kedma, A. Kachlon, and Y. Elovici, "Airhopper: Bridging the air-gap between isolated networks and mobile phones using radio frequencies," in *Proceedings of International Conference on Malicious and Unwanted Software*. IEEE, 2014, pp. 58–67.

[2] "Enabling smart phones in intel's factory," White Paper, Intel IT, September 2011.

[3] Biddle,Sam, "Us military bans all physical media on its internal network," https://gizmodo.com/5711205/us-military-bans-all-physical-media-on-its-internal-network, 2010, [Online, 25-Jul-2018].

[4] British Broadcasting Corporation, "Ibm workers banned from using usb sticks," https://www.bbc.com/news/technology-44069488, 2018, [Online, 25-Jul-2018].

[5] B. W. Lampson, "A note on the confinement problem," *Communications of the ACM*, vol. 16, no. 10, pp. 613–615, 1973.

[6] Wikipedia, "Covert channel," https://en.wikipedia.org/wiki/Covert_channel, [Online, 31-Jul-2019].

[7] M. Guri and Y. Elovici, "Bridgeware: The air-gap malware," *Communications of the ACM*, vol. 61, no. 4, pp. 74–82, 2018.

[8] Marie-Andree, "May an Employer Prohibit Employees From Taking Pictures at the Workplace?" http://www.maw-law.com/social-media-law/may-employer-prohibit-employees-taking-pictures-workplace/, [Online, 30-Jul-2019].

[9] M. Guri, B. Zadov, and Y. Elovici, "Odini: Escaping sensitive data from faraday-caged, air-gapped computers via magnetic fields," *IEEE Transactions on Information Forensics and Security*, 2019.

[10] M. Guri, A. Daidakulov, and Y. Elovic, "Magneto: Covert channel between air-gapped systems and nearby smartphones via cpu-generated magnetic fields," *arXiv preprint arXiv:1802.02317*, 2018.

[11] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro, "H. 264/avc baseline profile decoder complexity analysis," *IEEE transactions on circuits and systems for video technology*, vol. 13, no. 7, pp. 704–716, 2003.

[12] ITU-T, *High efficiency video coding, Recommendation ITU-T H.265*, International Telecommunication Union Recommendation, February 2018.

[13] I. E. G. Richardson, "H.264 and mpeg-4 video compression: Video coding for next-generation multimedia." John Wiley & Sons, 2004, p. 306.

[14] M. Ghanbari, *Video coding: an introduction to standard codecs*, 1999.

[15] I. Hong, G. Qu, M. Potkonjak, and M. Srivastavas, "Synthesis techniques for low-power hard real-time systems on variable voltage processors," in *Proceedings of IEEE Real-Time Systems Symposium*. IEEE, 1998, pp. 178–187.

[16] K.-S. Lee, H. Wang, and H. Weatherspoon, "Phy covert channels: Can you see the idles?" in *Proceedings of the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, 2014, pp. 173–185.

[17] N. Matyunin, J. Szefer, S. Biedermann, and S. Katzenbeisser, "Covert channels using mobile device's magnetic field sensors," in *Proceedings of the 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2016, pp. 525–532.

[18] "x264," https://www.videolan.org/developers/x264.html, 2018, [Online, 16-May-2018].

[19] L. Merritt and R. Vanam, "Improved Rate Control and Motion Estimation for H.264 Encoder," in *Proceedings of the IEEE International Conference on Image Processing*, vol. 5, 2007, pp. 309–312.

[20] Rodola, Giampaolo, "gomplayer," https://www.gomlab.com/, 2018, [Online, 31-Jul-2018].

[21] Intel, "Intel® core™ i5-4200u processor," https://ark.intel.com/content/www/us/en/ark/products/75459/intel-core-i5-4200u-processor-3m-cache-up-to-2-60-ghz.html, 2019, [Online, 28-Apr-2019].

[22] Moon Todd, K, *Error correction coding: mathematical methods and algorithms*. John Wiley & Sons, 2005.

[23] "Ffmpeg: A complete, cross-platform solution to record, convert and stream audio and video," https://www.ffmpeg.org/, 2018, [Online, 16-May-2018].

[24] Wiebelhaus, "(52) Hikvision DS-2CD2085G1-I 8MP 2.8mm Darkfighter Network Bullet Camera (Color Mode at Night Sample) - YouTube," https://www.youtube.com/watch?v=xFFaccsCfxY, [Online, 30-Jul-2019].

[25] "psutil," https://github.com/giampaolo/psutil, 2018, [Online, 31-Jul-2018].

[26] Keysight Technologies, "U2541a data acquisition," https://www.keysight.com/en/pd-1250127-pn-U2541A/250ksa-s-usb-modular-simultaneous-data-acquisition?cc=US&lc=eng, 2018, [Online, 1-Aug-2018].

[27] Texas Instruments, "Drv425," http://www.ti.com/product/DRV425, 2018, [Online, 1-Aug-2018].

[28] V. P. Kodali and V. Prasad, *Engineering electromagnetic compatibility: principles, measurements, technologies, and computer models*. IEEE, 2001.

[29] Armstrong, Eur Ing Keith and Clough, Cherry, "Emc for systems and installations: Part 4," http://www.compliance-club.com/archive/keitharmstrong/systems_installations4.html, 2018, [Online, 27-Jul-2018].

[30] N. Carlini, A. Barresi, M. Payer, D. Wagner, and T. R. Gross, "Control-flow bending: On the effectiveness of control-flow integrity." in *Proceedings of USENIX Security Symposium*, 2015, pp. 161–176.

[31] Y. Han, S. Etigowni, H. Liu, S. Zonouz, and A. Petropulu, "Watch me, but don't touch me! contactless control flow monitoring via electromagnetic emanations," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 1095–1108.

[32] Y. Liu, L. Wei, Z. Zhou, K. Zhang, W. Xu, and Q. Xu, "On code execution tracking via power side-channel," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1019–1031.

[33] A. Nazari, N. Sehatbakhsh, M. Alam, A. Zajic, and M. Prvulovic, "Eddie: Em-based detection of deviations in program execution," in *Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2017, pp. 333–346.

[34] M. Guri, M. Monitz, Y. Mirski, and Y. Elovici, "Bitwhisper: Covert signaling channel between air-gapped computers using thermal manipulations," in *Proceedings of Computer Security Foundations Symposium (CSF)*. IEEE, 2015, pp. 276–289.

[35] M. Guri, Y. Solewicz, A. Daidakulov, and Y. Elovici, "Acoustic data exfiltration from speakerless air-gapped computers via covert hard-drive noise ('diskfiltration')," in *Proceedings of European Symposium on Research in Computer Security*. Springer, 2017, pp. 98–115.

[36] M. Guri, A. Kachlon, O. Hasson, G. Kedma, Y. Mirsky, and Y. Elovici, "GSMem: Data exfiltration from air-gapped computers over GSM frequencies," in *Proceedings of USENIX Security Symposium*, 2015, pp. 849–864.

[37] H. Pan, Y.-C. Chen, G. Xue, and X. Ji, "Magnecomm: Magnetometer-based near-field communication," in *Proceedings of International Conference on Mobile Computing and Networking*. ACM, 2017, pp. 167–179.

[38] S. Biedermann, S. Katzenbeisser, and J. Szefer, "Hard drive side-channel attacks using smartphone magnetic field sensors," in *Proceedings of International Conference on Financial Cryptography and Data Security*. Springer, 2015, pp. 489–496.

[39] Z. Yang, Q. Huang, and Q. Zhang, "Nicscatter: Backscatter as a covert channel in mobile devices," in *Proceedings of the International Conference on Mobile Computing and Networking*. ACM, 2017, pp. 356–367.

[40] S. Ka, T. H. Kim, J. Y. Ha, S. H. Lim, S. C. Shin, J. W. Choi, C. Kwak, and S. Choi, "Near-ultrasound communication for tv's 2nd screen services," in *Proceedings of ACM International Conference on Mobile Computing and Networking*. ACM, 2016, pp. 42–54.

[41] K. Zhang, C. Wu, C. Yang, Y. Zhao, K. Huang, C. Peng, Y. Liu, and Z. Yang, "Chromacode: A fully imperceptible screen-camera communication system," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. ACM, 2018, pp. 575–590.

[42] N. Matyunin, N. A. Anagnostopoulos, S. Boukoros, M. Heinrich, A. Schaller, M. Kolinichenko, and S. Katzenbeisser, "Tracking private browsing sessions using cpu-based covert channels," in *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, ser. WiSec '18. ACM, 2018, pp. 63–74.