High Data Throughput Exfiltration through Video Cable Emanations

Llorenç Romá Álvarez[1], Dr. Daniel Moser[1], and Dr. Vincent Lenders[1]

Armasuisse Science & Technology, Cyber-Defence Campus
{llorenc.roma, daniel.moser, vincent.lenders}@ar.admin.ch

Abstract. The present work investigates the feasibility to exfiltrate a large amount of data from a computer by leveraging the unintended electromagnetic emanations of an HDMI cable to reconstruct its content. The low signal strength and noise of the leaked signals make difficult to recover any useful information, particularly when the content information is text based, since it suffers from low readability. We consider a targeted attack in which malicious software executed inside the victim's machine encodes the desired information into QR codes, which are then modulated on the HDMI cable and in turn received and reconstructed by the attacker. The efficiency of this method is evaluated under practical conditions showing that the system is capable of achieving a data exfiltration rate up to 12.67 Kbps under optimal conditions or 2.08 Kbps at 50 m distance. To the best of our knowledge, these results outperform, in terms of distance range and exfiltration rate, previous work in the field of electromagnetic leakage from the literature.

Keywords: side channel, emanations, data exfiltration, hdmi, QR code, tempest

1 Introduction

The security issue resulting from the electromagnetic emanations of electronic devices was initially worked on by governments and military entities. The practice of eavesdropping and protecting from eavesdroppers and their study are encompassed in a framework known as TEMPEST, a term established by the National Security Agency (NSA) in the USA [8].

Our work has the objective of investigating the feasibility of exfiltrating large amounts of targeted data by leveraging video cable emanations. Our work focuses on HDMI, yet the results are generally relevant to any other types of video cable such as VGA [31]. The proposed exfiltration method allows an attacker to exfiltrate data without leaving any traces on the network ,that is, avoiding detection by classical network-exfiltration countermeasures. We investigate QR codes to encode the data to be exfiltrated reliably and at high rates and we experimentally demonstrate data exfiltration throughputs several times higher than previously reported in the literature is possible.

Our main contributions are:

- A reliable technique of data exfiltration by combining the use of QR codes as encoding system with unintended HDMI cable emanations.
- Systematic evaluation of the impact of the distance and obstructions to the throughput.
- Experimental demonstration of data exfiltration rate up to 2.08 Kbps at 50 m distance, 5.12 Kbps at two floors away and a maximum rate above 12 Kbps under optimal conditions (antenna placed next to the target system).
- Open source software pipeline to reproduce the full attack (to be uploaded in a GitHub repository).

The rest of the paper is organised as follows: in Chapter 2 we present other works related to data exfiltration using side channels, in particular those exploiting video monitor emanations. In Chapter 3 we introduce the technical background of our work and describe how QR Code technology is used to exfiltrate data reliably in such attacks. In Chapter 4 we introduce the attacker model and describe the different stages of the attack. Chapter 5 outlines the details of the software pipeline implemented for the attack and Chapter 6 summarizes the results achieved after evaluating the attack in multiple scenarios. We performed experimental analysis covering different parameters that affect the transfer bitrate of the attack. In Chapter 7, we propose countermeasures and describe the drawbacks of our approach. Finally, in Chapter 8, we draw the conclusions derived from our work and future work to consider.

2 Related Work

Compromising electromagnetic emanations have been under research from the mid 1980's. Wim Van Eck published the very first analysis of video monitor emanations security risks in 1985 [32] where he described how to reconstruct the content from video monitor emanations without the need of professional and expensive equipment but with just a TV broadcast receiver. The threat was confirmed by implementing a practical attack in which he could reconstruct and monitor the content of a screen in a real world scenario. Years later, around the year 2000, Kuhn [25, 28] analyzed the emanations coming from computer displays in depth and discussed the concept of a software based TEMPEST attack, showing that the electromagnetic emanations from LCD screens can be controlled to modulate data. The researchers also showed how recovering the content of the screen provided them a way to perform text-based exfiltration attacks at distances ranging from 3 to 10 m. Further, they proposed different techniques to improve the recovery of plain text from emanations via radio-character recognition. However, their proposal focus on recovery of text characters, which becomes more prone to error as the SNR decreases due to the similarity between different text characters (e.g, "i" and "l", or "v" and "u").

Other works exploiting video monitor emanations have been published since then. For instance, Guri et al. [14] presented a malware, namely *AirHopper*, which encodes text/binary data from a target into FM signals and uses a mobile phone

as FM receiver. In order to achieve this, they modify the content of the monitor to control the emanated signals (FM signals) as discussed in different works and technical papers [12, 25, 28, 32]. The maximum throughput they achieved was 480 bps at 1 m distance. Aside from electromagnetic emanations, other side-channels have been studied in the past. These air-gap covert channels are classified in seven main categories: electromagnetic, magnetic, electric, acoustic, thermal, optical and vibrational. In [20] the authors used cellular frequencies to exfiltrate data, achieving up to 1000 bps rate at 2.6 cm distance. Despite of this being the approach presenting the higher exfiltration rate it comes with a limitation regarding the distance. In [17–19, 22] the authors used different covert channels providing exfiltration rates ranging from 1 bps to a maximum of 100 bps, and distances up to 1.5 m being the last one the slowest, exfiltrating 8 bits per hour using the CPU generated heat as a channel.

In this work, we investigate the data exfiltration rate from a video cable. Similar to the text-based exfiltration approach showed by Kuhn [25, 28], we encode the information in the video signal. However, we combine it with QR codes in order to circumvent the limitations of text character recovery: since QR codes are highly resistant to errors, it results a much better coding scheme for environments where the SNR is reduced due to environmental noise, such as in the TEMPEST attack. Finally, we show how the system can effectively exfiltrate data from a computer from up to 50 m distance or alternatively, from different floors, achieving exfiltration rates and distances several times higher than other previous attacks. A comparison of our work and the previous related work using covert channels used for data exfiltration is shown in Table 1, where we can see the advancement of our approach in terms of exfiltration rate and distance.

3 Background

In this section, we first introduce the term TEMPEST and describe the HDMI data stream and nature of its leaked signals. Then, we detail the properties of QR codes and why we chose it as encoding scheme for data exfiltration.

3.1 TEMPEST

In 1972, the NSA conducted a classified study which was partially declassified later [8]. The document described how in one experiment conducted at Bell Labs, they were able to detect plain text from an emanated signal coming from a message encryption device. This device was used during the Second World War by the US army and that experiment highlighted the importance of having control over electromagnetic emanations. The term TEMPEST was then used to describe a set of standards, specifications and certifications [7, 9] referring to the act of spying on information systems through leaking emanations, which includes unintentional electromagnetic signals, sounds and vibrations. TEMPEST also defines how devices can be protected against such attacks. Among these

Table 1: Comparison of current covert channels for air-gapped networks.

Attack (covert channel)	Type	Distance (m)	Rate (bit/s)
AirHopper [14] (FM signals)	EM	1-7	105-480
LCD Tempest [15] (AM signals)	EM		60-640
GSMem [20] (Cellular Frequencies)	EM	30 +	100-1000
AIR-FI [17] (WiFi signals)	EM	0-8	100
ODINI [22] (Faraday shield bypass)	Magnetic	0.05 - 0.1	1-40
PowerHammer [21] (power lines)	Electric		100-1000
Ultrasonic [23][24] (speaker)	Acoustic	19.7	20
BitWhisper [16] (CPU generated heat)	Thermal	0.4	8 bit/hour
BRIGHTNESS [19] (screen brightness level)	Optical	9	10
AiR VibeR [18] (computer fan vibrations)	Vibrational	1.4	0.5
Exfiltration with QR (QR code recovered img)	EM	1 - 50	2080-12670

measures, the standards consider distance between equipment and walls, distance between wires carrying classified and unclassified information, or masking information by introducing noise.

3.2 Video Signal

As described by Marinov [31] a video frame is built up from y_t lines formed by x_t pixels. The display refreshes a frame at f_v frames per second (fps). In addition to the visible pixels, there are also non-visible pixels (i.e., blanking pixels), which are used to synchronise the start and the end of a line or a frame. Each pixel is a combination of three colors (RGB). In the case of digital signals, the intensity of each RGB component is represented by a number n_b of bits ($n_b = 10$ bits for HDMI). Therefore, the bit duration is

$$t_b = \frac{1}{x_t \cdot y_t \cdot f_v \cdot n_b} \tag{1}$$

where x_t and y_t depends on the screen resolution and f_v is determined by the screen refresh rate. Then, pixel i is transmitted at $t_i = i \cdot t_b \cdot n_b$.

Consider a display transmitting a bit stream of values $c_k (k \in Z)$, where the k-th value represents the bit (k mod n_b) of the binary number used to represent

pixel intensity of pixel number $||\frac{k}{n_b}||$. The resulting video signal in the time domain is defined as [31]

$$\widetilde{v}(t) = \sum_{k=-\infty}^{+\infty} c_k b(t - kt_b) \tag{2}$$

where $b(t)$ is the shape of a digital bit and $b(t) = 0$ for $|t| \gg \frac{t_b}{2}$.

This signal is going to be repeated at regular intervals at frequencies multiple of $\frac{1}{t_b} = x_t \cdot y_t \cdot f_v \cdot n_b$. In order to receive this video signal, the receiver needs to be able to pick up frequencies up to $f \geq \frac{1}{2t_b}$ to be above the Nyquist sampling rate. For example, using a laptop with $n_b = 7$ with resolution values of 800x600 @ 75 fps with $x_t = 1056$ px, $y_t = 628$ lines and $f_v = 75\,Hz$, we should expect the signal centered at multiples of $\frac{1}{t_b}$: 348.2 MHz.

3.3 QR Code

The QR code technology was created in 1994 as an alternative to the commonly used barcodes due to their limitations in terms of storage capacity. Initially the QR code was adopted by the car industry for efficient work management through a range of tasks. Subsequently, the use of this code was widely spread across multiple industries and nowadays almost every smartphone has a QR code reader. A QR code consists of a square-shaped figure built up from black and white squares, called modules. The main features of this technology are described below.

Encoding modes A QR code encodes a set of characters as a string of bits. The encoding can be optimized depending on the type of characters to encode by using one out of four encoding modes: numeric, alphanumeric, byte (including UTF-8) and Kanji (a set of Japanese characters that require three or four bytes to be encoded). The encoding mode will create the shortest possible string of bits for the input, so it will result in higher capacity. For instance, encoding a set of numeric characters in *numeric* mode will allow to encode more numeric characters than when they are encoded in byte mode.

Error Correction Level (ECL) QR codes are designed to be able to recover data if the code is dirty, damaged or partially obstructed. A sensible requirement stemming from their origin in industrial and production environments where codes could be easily degraded. Four error correction levels are available and the higher the level the higher correction capability. However, the increased tolerance to errors comes with a lower data capacity.

Data Capacity How much data a QR code can store depends on three factors: the data to be encoded (encoding mode), the error correction level and the

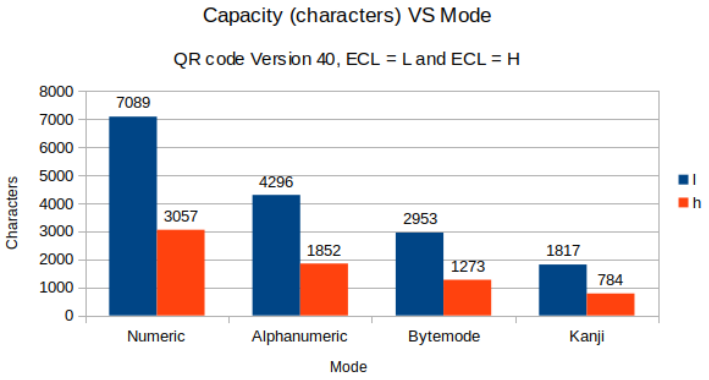

Fig. 1: Difference encoding capacity between QR codes with ECL set to L and H.

version of the QR code. The version of the QR code determines its dimensions in terms of modules - it is 21x21 for version 1 and 177x177 for version 40, the highest version number at the time of writing. Each version add 4 modules to each side of the QR code.

Fig. 1 shows the maximum number of characters that a QR code of version 40 can store. It shows the capacity for the four encoding modes and the lowest (L) and highest (H) ECL. It can be seen that switching from the highest ECL to the lowest one is has a big impact in terms of the maximum number of characters that can be stored; it increases by more than 100%. See Appendix A for a list with the maxima for all possible combinations of encodings and ECL.

Motivation behind the use of QR code for the attack QR codes have been invented and designed to encode arbitrary data into a visual representation that can be captured and decoded using electromagnetic (electro-optical) sensors, despite (a limited amount) of damage to the QR code (e.g, scratches) itself, low SNR on the optical channel (e.g., bad light conditions) and distorsions and faults (e.g., dead pixels) introduced by the sensor. However, the elements that optimize a QR code in terms of visual representation and transmission are also useful when reconstructing the visual representation from electromagnetic radiation of video cables. In particular, these are:

- Error correction capabilities account for errors at the module level; for example, a module that was black was reconstructed as being white.
- The well-defined structure of a QR code con be leveraged to reduce the difficulty of the recovery process. Its fixed size squares perfectly aligned on a grid make it easier to locate data units(square) and cope with distortions to them. This in contrast to, for example, text-based approaches where the

width of characters might vary and even small distortions might make one character look like another (e.g., 'l' and 'i').
- Black/white have maximum hamming distance so, it is likely that despite errors, some distance remains and it is still possible to determine if a pixel should be black/white when reconstructing the QR codes.

Furthermore, with different encoding for different error correction levels, the QR code format enables an attacker to optimize throughput for different inputs and exfiltration scenarios; for example, with or without walls between the attacker and the video cable.

4 Attacker Model

Espionage by eavesdropping on the EM emissions of a video cable can in principle be carried out without manipulating the target system. The attacker must then simply hope that the user is watching something interesting and that this can then be reconstructed from the emissions. However, if specific data is to be exfiltrated and optimized with respect to the transmission method, this requires an active attacker. As a consequence, the attacker model includes at least the following capabilities and steps: (1) place malware on the target system (2) collect and exfiltrate data (3) receive and decode EM emanations from the victim's video cable. In the following, we discuss the steps in more detail. Fig. 2 shows the setup after the attacker has successfully infected the system with their malware.

In a typical scenario, an attacker with access to internal data would exfiltrate to the Internet over the network. However, this would leave traces and make the exfiltration noticeable to the implemented security solutions, namely IDS (Intrusion Detection System), logs, firewalls etc. Since our attack does not exfiltrate the data over the network, it remains unnoticeable to such security measures. However, as discussed in 7, one of the limitations is that displaying QR Codes on the screen does leave traces on a different channel (e.g., to passers by).

4.1 Target Infection

There are many different vectors for infecting a target, starting with remotely exploitable vulnerabilities, compromised software updates, malicious employees or social engineering. Depending on whether and how a system is networked with other systems, more or fewer vectors come into play. Air-gapped systems probably present the most difficult hurdle here, but even this can be overcome in practice. Examples include Stuxnet [11, 29] and Agent.btz [30]. Once the system is infected, the malware should be able to do its job; additional steps like local privilege escalation are often not required. After all, reading data and displaying images can be done by most users.

In the following subsections, we do not discuss step (1) any further. Our focus is on steps (2) and (3). For the reminder of this work, we assume that we were able to place our malware on the target system.

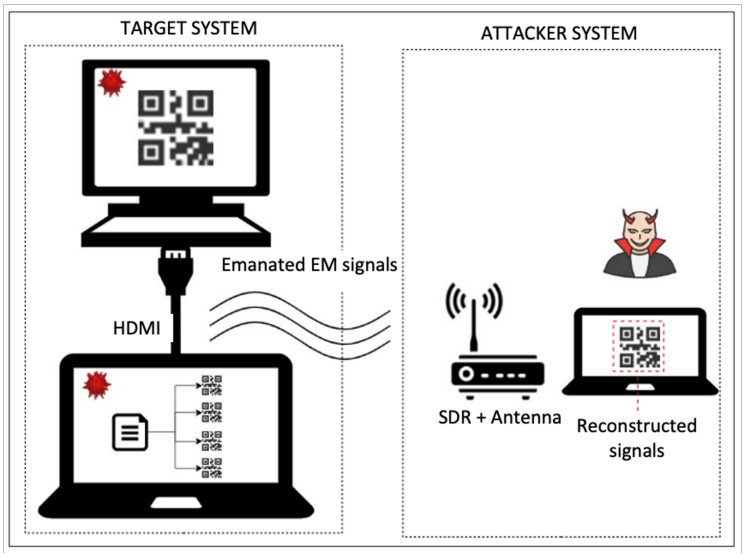

Fig. 2: Attacker model diagram.
The target computer, infected with malware, displays QR codes on the screen while the attacker receives the emanated signals with an SDR device to reconstruct the codes and extract information.

4.2 Data collection and exfiltration

Once the computer is infected, the malware collects sensitive data. It encodes the data into QR codes and finally displays the generated QR codes on the external connected screen. In such scenario, the screen's content is leaked through the HDMI emanated signals.

4.3 Signal Reception/Decoding

For reception, the attacker uses a computer with an SDR (see Section 5) and an antenna in order to detect, receive and process the signals to reconstruct the content from the emanated video signal. After signal acquisition, the attacker proceeds to the decoding of the recovered QR codes.

For this purpose, the attacker needs access to the target's surrounding where reception of the emanated signals is possible whilst not being discovered. This can be achieved in a multitude of ways. First it is important to note that the attacker does not have to be present during the attack but can place the receiver system at a suitable location and fetch it later to process the recorded signals. If the receiver can be hidden well enough, any rented-out or publicly accessible room that the target will use would do the job. If hiding it is difficult or the room will be checked for such devices before using it, the receiver could be placed in any room nearby, where the signal is still strong enough. The top candidates being the rooms above, below or next to the room in question.

5 Implementation

In this section, we describe the software pipeline we developed to experimentally investigate the exfiltration data rates in the proposed attacker model from previous sections.

5.1 File encoding – QR code generation

The malware collects sensitive data and splits them in blocks of N characters where N is the capacity of the QR code version used. To identify recovered QR codes on the receiver side, a header is added on top of the encoded data. In our experiments we used the filename and the data block number encoded (see Fig. 3). Another identification method could be used (e.g., one byte per file and one byte per file data block).

Once QR codes are generated, they are displayed on the external screen to originate the emanations of the signal that will be recovered with the receiver system. It is convenient to display the codes in full-screen, thus, the modules are better differentiated on the receiver side and it is less prone to errors due to poor quality. Moreover, adding a black border helps to identify and delimit the area containing the QR code. An example of a generated QR code displayed on the monitor is shown in Fig. 4.

5.2 Signal reception and frame reconstruction

To receive the emanated signals, we apply a band-pass filter centered at a multiple of the target's pixel frequency. The sampling rate was set at twice the target's

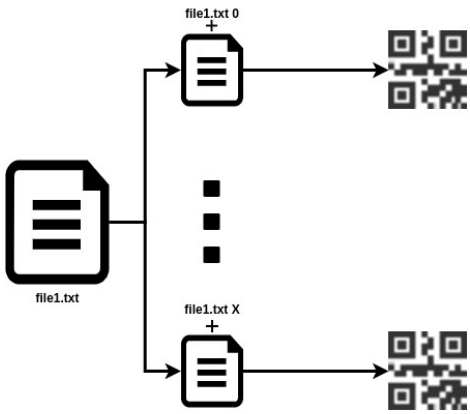

Fig. 3: QR code generation. A file is split in X blocks and encoded into several QR codes. A header with information about file name and the block of data is added to the content of each QR code.

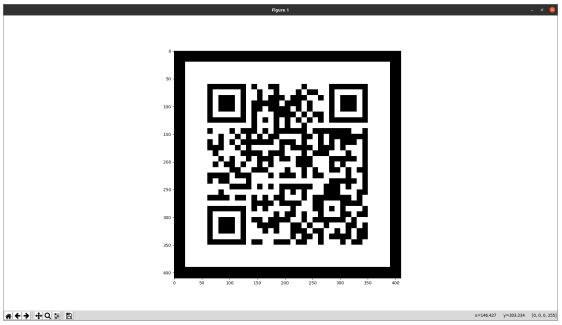

Fig. 4: QR code generated on the target computer.

pixel rate so each pixel is identified. We collected the RF signals with an SDR and they were send to a PC for demodulation and processing.

For frame reconstruction, we leverage the video signals properties as shown in Section 3. The decoding of the video signal relies on the values x_t, y_t and f_v of the screen. Different approaches to find these values are proposed by Marinov [26, 31]. First, the signal generated might be considered periodic, and therefore, the received signal can be analysed to perceive patterns and obtain f_v. For instance, in [31] autocorrelation is used to discover repeating patterns of the received signal and to estimate f_v. Second, y_t is estimated from the repeating nature of frame blanking intervals (invisible pixels) from consecutive received samples. Last, there are a set of resolutions that are broadly used, for instance 2560x1440@60Hz or 1920×1080@60Hz: an attacker could adjust the receiving parameters to match these resolutions.

Before displaying the reconstructed video frame, we applied digital frame averaging in order to reduce random noise and increase the signal SNR.

5.3 Image processing - QR code detection

In a real-world scenario, the attacker would be spying the target computer from a certain distance, thus, the received signals would be expected to contain noise, which might damage the recovered QR codes. In particular, recovered images are in gray scale and present low contrast, therefore, they must be preprocessed to reduce the impact of the noise on their readability before decoding. To recover those codes, we adjusted their contrast and binary thresholds, as well as applied noise removal techniques. Finally, we used two standard python QR code decoding libraries (i.e., *PyQR* [4] and *PyZbar* [5]) to retrieve the information from the recovered QR codes.

– **Brightness and contrast**
  Since most of the reconstructed codes' contrast level are low, decoders are not able to distinguish well between black and white modules. For instance, the QR code in Fig. 5 (a) could not be decoded by any of the decoders.

– **Thresholding/ Binarization**
Sometimes, the code is not readable after increasing contrast and brightness. However, once the contrast is higher, black and white modules are easier to discriminate and, based on a threshold, we apply binarization to map gray values to black or white. Depending on the gray value, the resulting pixel is assigned as follows:

$$\begin{cases} 0 & \text{for x} < \text{threshold} \\ 255 & \text{otherwise} \end{cases}$$

where x represents the initial pixel's value in 8-bit format.
This part is error prone due to noise: some of the gray values which should be white are considered as black and the other way around, thus, if enough modules are wrongly interpreted, the code becomes unreadable. Fig. 5 (b) is the result of thresholding Fig. 5 (a), initially not decoded. After applying thresholding, both decoders succeeded.
– **Noise removal - Erosion and Dilation**
All recovered images contain noise which we minimize by applying median filtering and linear gaussian filtering. These two techniques preserve the edges of the image, which in the case of the QR code modules is essential. Furthermore, there might still be some noise in the form of small lines all over the image as shown in Fig. 6. In order to reduce this type of noise, we apply dilation and erosion which reduce the small lines and dots.

(a) Recovered image without preprocessing. Not decodable

(b) Recovered image with increased contrast and thresholding applied. The decoded text is part of a book: "Preface to the Third Edition ..."

Fig. 5: Recovered QR codes before and after image processing.

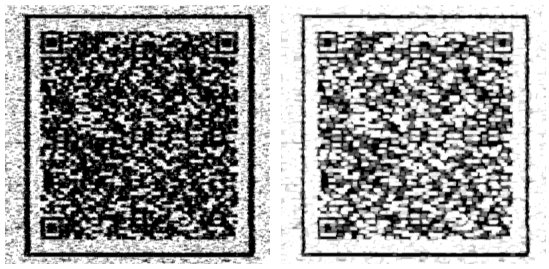

Fig. 6: On the right image we applied dilation and erosion to reduce to noise. However, it is not decodable yet due to the small size of the modules which makes it impossible to differentiate.

5.4 QR code decoding and data recovery

Once QR code data is retrieved, we assembled the different blocks of data by analysing the header of decoded data as explained in 5.1. Fig. 7 shows the steps on the receiver side, from signal reception to data recovery.

6 Evaluation

In this section, we evaluate the reliability and the data exfiltration rate of the attacker model in different scenarios.

6.1 Setup

The equipment used for the experiments and its specifications are described in the following subsections.

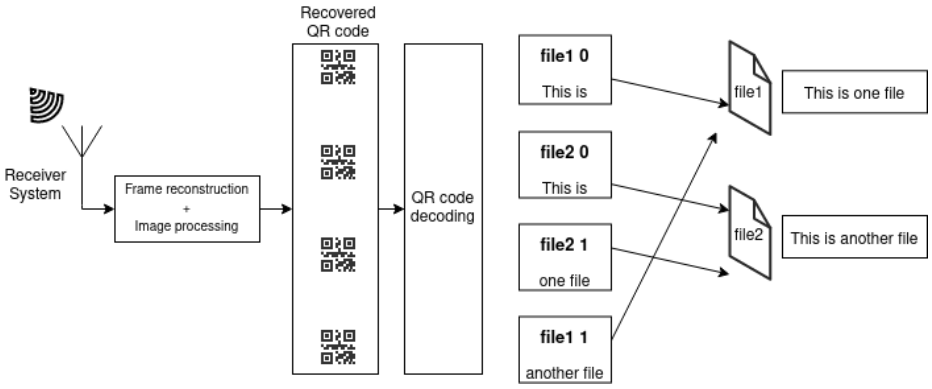

Fig. 7: Signal acquisition, frame reconstruction, QR code decoding and data collection process.

Target system As the target system, we used an *HP Elitebook* running Ubuntu 20.0.4 with 16 GB of memory, an Intel Core i7-8550UCPU@1.8GHz x 8 processor and Intel UHD Graphics 620 (KBL GT2) graphics. This system runs the malicious code described in Section 5. The computer is connected to an external video monitor: we tested an HP Z27n (27') model for the experiments and repeated them for different screens, including a modern super wide screen (*Samsung 49" QLED Gaming Monitor 5120x1440@60Hz*). The decoding rates were similar for all of them.

Regarding the video cable, we tested different commercial 3 m long High Speed HDMI cables and repeated the experiments with a 15 m long cable. The cables did not have any special shielding.

Notice that we performed the experiments with a laptop connected to an external monitor only for convenience. That is, the results also apply to desktop PCs as the emanations come from the video cable and not the computer device[12, 14, 27, 31].

Attacker system To receive and process the leaked signals, we used a commercial off-the-shelf software-defined radio, namely the Ettus USRP B210 model [2]. We used a directional Yagi antenna (with frequency range of 698 - 3800 MHz) since this type is well suited if we are targeting a specific computer. In our experiments we added a *Mini-Circuits* ZX60-3018G-S+ 13 dB amplifier [1] with an additional power supply of 12V and a filter to reduce the undesired signals [3]. Finally, we used another identical *HP Elitebook* to perform all the image processing and signal reconstruction tasks.

Receiving parameters In our setup, we recovered the images at 1210 MHz and sampling rate values between 15 MHz and 20 MHz.

6.2 Scenarios

We evaluated different scenarios. In each scenario, the antenna was placed in a different location. For each of the experiments, we evaluated the ability of the system to exfiltrate three text files encoded into 20 QR codes each, independently of the QR code version. That is, for an experiment to be successful, we should recover and decode 100% of the codes (60 in total). All QR codes were generated using *bytemode* (i.e., 8 bits each character).

Zero distance First, we evaluated a scenario in which the antenna was placed at less than 50 cm from the target system to avoid signal degradation due to distance. For this experiment, the Error Correction Level (ECL) is set to Low (L) and the time a QR code is displayed on the screen is 0.5 s. For the rest of the experiments, we consider the same ECL an display time if not specified otherwise.

The results for different QR code versions are shown in Fig. 8. The vertical axis represents the percentage of QR codes which are recovered out of the 60

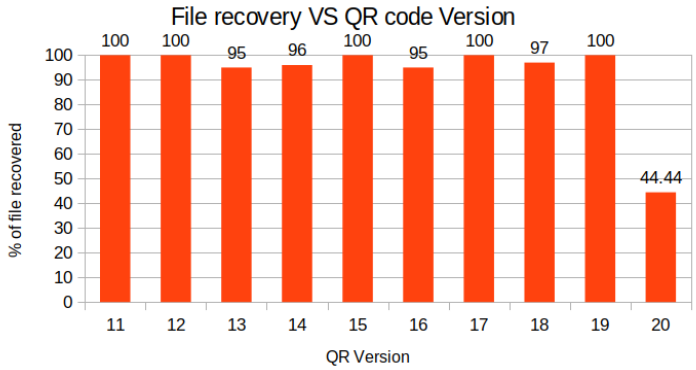

Fig. 8: File recovery percentage for different QR code versions. ECL is set to L and display time is 0.5 s.

total codes. For instance, 100% means that the whole set of 60 QR codes (20 per file) are recovered and decoded. In an environment with good conditions where the channel noise remains low, the file recovery rate is close to 100% up to QR code version 19. For higher versions, the modules of the QR codes are smaller. With small modules, subtle interference can cause adjacent modules to mix, introducing errors. Considering the capacities specified in Appendix A, using version 19 QR codes in bytemode (max of 792 characters), with the specified conditions (displaying time of 0.5 s and ECL set to L), we reach a maximum throughput of $\frac{6336\ bits}{0.5\ s} = 12\,672$ bps.

In the next evaluation, we examined the impact of reducing the time that a QR code is displayed on the exfiltration rate. The combination of a higher version code and lower display time will provide us a higher throughput. However, we will need to increase the sampling rate on the receiver and therefore, frames will start being dropped. Fig. 9 shows the results. Using QR code version 11 and a display time of 0.25 s, we were able to recover the whole set of files. This results in a maximum throughput without errors of $\frac{2568\ bits}{0.25\ s}$ =10 272 bps.

Finally, we evaluated how the value of the ECL impacts the recovery rate (shown in Fig. 10). In this experiment, we compared the file recovery percentage for different versions while using ECL set to L , Q or H. We placed the antenna at a longer distance in such a way that the noise introduced in the images is more noticeable and the effect of the ECL is observed. From version 9 on, we could not recover any block of data using level L whereas for levels Q and H it was still possible since they tolerate more degradation in the codes. We also evaluated lower versions not shown in the figure, as they all lead to a 100% recovery of the data. The main drawback is the high overhead introduced when increasing the ECL: for H level the capacity (352 characters) is about 40% lower compared to L level (848 characters).

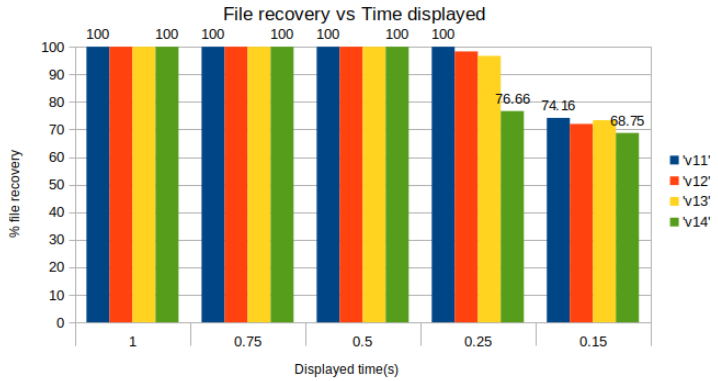

Fig. 9: File recovery percentage for versions 11 to 14 while modifying the display time of the QR codes.

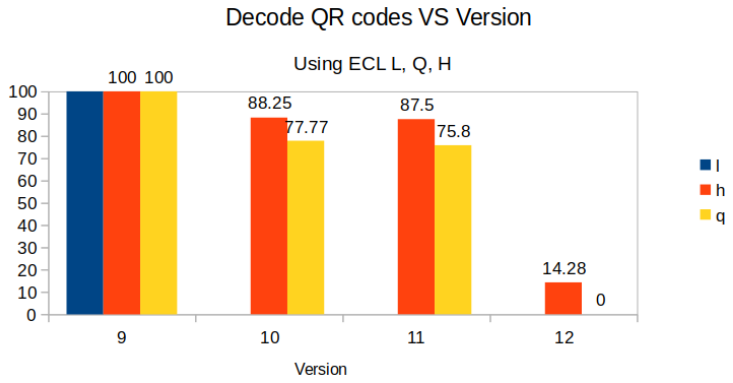

Fig. 10: File recovery percentage for versions 9 to 12 and ECL values L (blue), Q (yellow) and H (orange). For version 12, we can only recover some of the codes using high error correction level.

Throughput vs Distance We measured throughput obtained under line-of-sight conditions between the antenna and the target system over the distance. Despite being under line-of-sight conditions, the experiments were not performed in an interference-free environment but in a real office floor with other laptops and monitors used at the same time.

To get the maximum throughput, for each distance we combined the version and the displaying time values that allowed us to recover all the 60 QR codes.

Fig. 11, depicts the results for up to 50 meters distance. At 50 meters distance, we obtained the maximum throughput displaying version 4 QR codes during 0.3 seconds. Notice that for longer distances, the image contained a lot of noise and it was not possible to recover all the files.

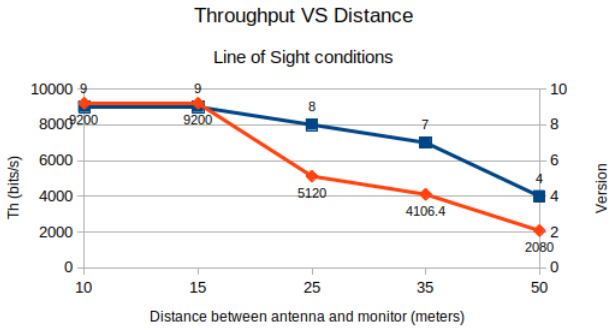

Fig. 11: Characters per second VS the distance under line-of-sight conditions. The line with diamond markers shows the throughput whilst the line with square markers shows which version was used to obtain this throughput.

Throughput vs Walls In the next scenario, we placed the antenna in adjacent rooms. Fig. 12 depicts the position of the antenna as well as the distances to the target system and Table 2 shows the throughput obtained for each position. As expected, walls and doors significantly attenuated the leaked signals leading to a limited (yet still high) recovery rate.

In the first case (Room 1), where the antenna is located in an adjacent room with only one glass door in between, we managed to reach approximately the same throughput as in the line-of-sight experiment with no obstacles.

The second case (Room 2) considers the attacker to be located in an adjacent room, with two closed glass doors in between. With two doors and a longer distance (10 m) in between, the throughput dropped by 41.7% compared to the case with just one obstacle.

In the last case (Room 3), we placed the antenna in another room, at 15 meters and again with two doors in between. The results show how the combination of the glass doors and the distance negatively impacts the recovery rate,

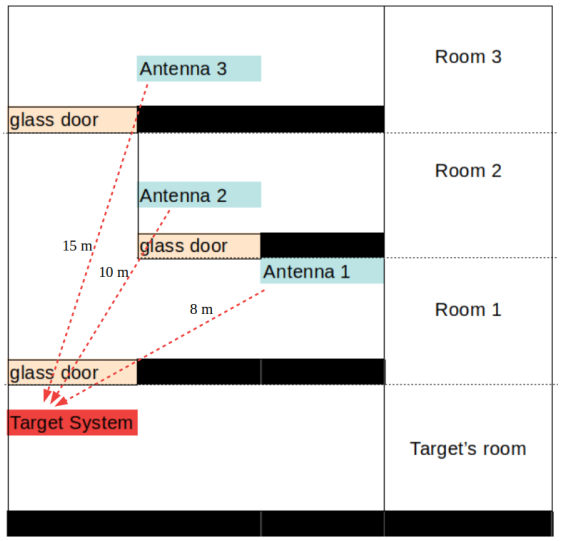

Fig. 12: Antenna and target position on the same floor.

Table 2: Maximum throughput in bits per second for different positions of the antenna on the same floor. Version and display time of the QR codes that provided the maximum throughput are shown for each position.

Room	Version	Time (s)	Distance (m)	Throughput (bits/s)
1	9	0.2	8	9200
2	6	0.2	10	5360
3	4	0.3	15	2080

which barely reached 22% compared to the results at the same distance in the line-of-sight scenario.

Throughput vs Floors If we consider a scenario in which an attacker has access to one floor in the same building of the target, we could assume that the victim would not notice the attacker's presence at all. For this experiment, we placed the attacker setup in different floors below the target system. Notice that the attacker should roughly know the target's position in order to point the antenna in such direction.

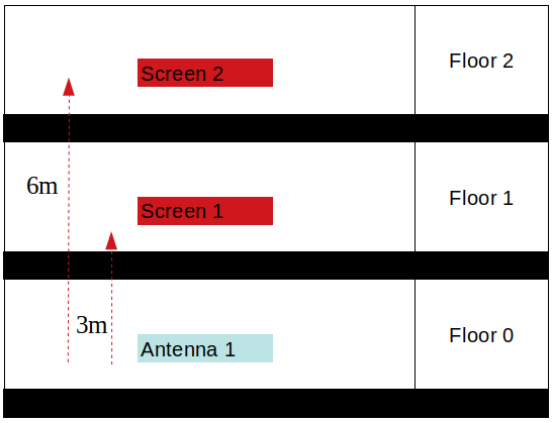

Fig. 13: Position of the antenna in different floors

The attacker' setup was placed in one and two floors below the target. The floors were built out of concrete. Table 3 shows the parameters that provided the highest throughput for each case. The throughput obtained in the latter case is around 44% less compared to the former.

Table 3: Highest QR code readable version and version and display time providing the highest throughput when positioning the antenna in different floors.

Floor	Version	Time (s)	Distance (m)	Th (bits/s)
1	9	0.2	≈ 3	9200
2	12	0.4	≈ 6	7336

7 Discussion

In this paper, we show how QR Codes can be used to increase the data exfiltration rate when exploiting electromagnetic signals from video cables compared to

other related works. We demonstrate how this approach worked in different real scenario and saw that an attacker would be able to effectively exfiltrate high amounts of data in a limited time without need of high-cost equipment. The results in Section 6 show that the use of QR Codes provides a higher resistance to noise/signal degradation, allowing to achieve exfiltration rates much higher than in similar literature. We learned that lower versions of QR Code provide more reliability when the environmental noise is high, while higher versions of QR Codes may be used to increase exfiltration rate when the amount of noise is reduced.

However, these very good results also come with several limitations. First and foremost, displaying QR codes on a computer's screen won't go unnoticed by its user or any passers-by. While the malware could eventually detect the presence of a user (e.g., using sensors like camera, microphone/speaker, or monitoring keyboard/mouse activity) and not show the codes whenever one is present, this might be much harder or impossible to do for the case of random passers-by. Things are simpler when the attackers knows time slots where no one is around or when the malware displays QR codes in an way invisible to humans. The later could eventually be achieved with real-time watermarking [13] techniques or with dithering [26].

Another limitation of our approach is that the attacker has no way of controlling the malware after it has been deployed; there is no backchannel. The attacker has no way of telling the malware that decoding a certain QR code failed and to have it displayed again. Fortunately, for conservative settings (small QR code versions, where modules' size are bigger), this is very unlikely to happen as our experiments show. However, it cannot be ruled out completely and depending on the nature of the data to be exfiltrated, especially if loosing one QR code could void the whole exfiltration, measures to mitigate this should be considered. The first choice here would probably be adding inter QR code error correction as this consumes less of the exfiltration bandwidth than transmitting all QR codes twice (or more times). Another option would be the use of side-channels as backchannels. Depending on the available sensors on the target computer and the overall situation this might work but is rather iffy and not completely passive anymore.

Despite having achieved good results during the experiments, further research should be done to make the exfiltration more robust and reliable: as discussed, improving the image processing to decode very noisy QR Codes might provide a larger exfiltration rate. In addition, one of the main obstacles in our approach (and related work) is the user's presence: investigating how to hide the QR Codes from user's presence remains as a challenge.

7.1 Countermeasures

Designing an electronic device which does not emanate electromagnetic signals can be hard, however, there are measures that can minimize those emanations.

Different standards and specifications define the requirements that electronic systems should implement in order to protect against general TEMPEST attacks [7, 9]. For instance, the *red* and *black* rule, or *zones* classification to define

the perimeter to be controlled or prevent signal reception. However, we have shown that structural building elements such us concrete walls/floors, do not mitigate the exfiltration method described, if we can get close enough. Shielding the transmitter component appears to be a better countermeasure, however, the HDMI video chip also leaks signals and that might be more difficult to shield. Another defensive strategy involves monitoring the presence of passive eavesdroppers such as in [10]. However the current detection range of such system is far below the exfiltration range of the work presented in our paper.

Another alternative we propose is the use of High Bandwidth Digital Content Protection (HDCP), which is supported by almost every modern HDMI chip and encrypts the protected content at chip level before sending the signals through the HDMI cable. Thus, the signals received by the attacker would be encrypted and therefore, illegible. Currently, HDCP encrypts only Digital Protected content. The solution would consist of applying the encryption to all the content sent through the HDMI cable. We experimented by playing Netflix on the target system and the recovered content was completely scrambled.

Finally, the target system could apply a permanent invisible overlay that results in visible patterns on the receiver side, in such a way that the QR codes are obstructed with those patterns (i.e., a watermarking to block the QR codes).

8 Conclusions

In our work, we presented a method for exfiltrating data using a commercial off-the-shelf software defined radio without leaving network traces. The covert channel used is based on the electromagnetic waves leaked from a video cable connecting a computer to an external display. The software used for exfiltration leverages QR codes to encode the targeted data. We have provided the technical background about video signals and QR code technology, and justified the use of it as a low SNR resistant encoding scheme.

We have described the whole attack pipeline, from data encoding, to signal reception, noise reduction and data recovery. We have evaluated the method using extensive variations and encoding parameters. In our results, we demonstrated that the attacker model described is feasible even with an isolation of several concrete floors. We also showed that the screen and the video cable length do not have an impact in our proposed method. With the experiments, we showed an effective distance of 50 m without obstacles and also a scenario where the target was located at two levels above the receiving system. Compared to similar works, the presented method provides an exfiltration distance range and a throughput several times higher.

Finally, we discussed the limitations of our approach and proposed countermeasures to protect against such attack.

Bibliography

[1] Amplifier Datasheet. https://www.minicircuits.com/pdfs/ZX60-3018G+.pdf, accessed 12-08-2020

[2] Ettus USRP B210. https://www.ettus.com/all-products/ub210-kit/, accessed 12-08-2020

[3] Filter Datasheet. https://www.minicircuits.com/pdfs/ZX75BS-88108+.pdf, accessed 14-08-2020

[4] Python-qrcode. https://github.com/lincolnloop/python-qrcode, accessed 12-08-2020

[5] PyZbar. https://github.com/NaturalHistoryMuseum/pyzbar, accessed 12-08-2020

[6] QR Code Tutorial. https://www.thonky.com/qr-code-tutorial/, accessed 12-08-2020

[7] Agency, N.S.: National Security Agency Specification For Shielded Enclosures Specification NSA No. 94106 (1994)

[8] Agency, U.N.S.: TEMPEST: A signal problem (1972)

[9] Assurance, N.I.: Tempest equipment selection process (1981)

[10] Chaman, A., Wang, J., Sun, J., Hassanieh, H., Roy Choudhury, R.: Ghostbuster: Detecting the presence of hidden eavesdroppers. In: Proceedings of the 24th Annual International Conference on Mobile Computing and Networking. p. 337–351. MobiCom '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3241539.3241580, https://doi.org/10.1145/3241539.3241580

[11] Clark, A., Zhu, Q., Poovendran, R., Başar, T.: An impact-aware defense against stuxnet. In: 2013 American Control Conference. pp. 4140–4147 (2013)

[12] Erik Thiele: Tempest For Eliza. http://www.erikyyy.de/tempest/, note=Accessed on 12-08-2020, (2001)

[13] Gugelmann, D., Sommer, D., Lenders, V., Happe, M., Vanbever, L.: Screen watermarking for data theft investigation and attribution. pp. 391–408 (05 2018). https://doi.org/10.23919/CYCON.2018.8405027

[14] Guri, M., Kedma, G., Kachlon, A., Elovici, Y.: 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). pp. 58–67 (2014)

[15] Guri, M., Monitz, M.: Lcd tempest air-gap attack reloaded. In: 2018 IEEE International Conference on the Science of Electrical Engineering in Israel (ICSEE). pp. 1–5 (2018). https://doi.org/10.1109/ICSEE.2018.8646277

[16] Guri, M., Monitz, M., Mirski, Y., Elovici, Y.: Bitwhisper: Covert signaling channel between air-gapped computers using thermal manipulations. In: 2015 IEEE 28th Computer Security Foundations Symposium. pp. 276–289 (2015). https://doi.org/10.1109/CSF.2015.26

[17] Guri, M.: Air-fi: Generating covert wi-fi signals from air-gapped computers (2020)

[18] Guri, M.: Air-viber: Exfiltrating data from air-gapped computers via covert surface vibrations (2020)

[19] Guri, M., Bykhovsky, D., Elovici, Y.: Brightness: Leaking sensitive data from air-gapped workstations via screen brightness. 2019 12th CMI Conference on Cybersecurity and Privacy (CMI) (Nov 2019). https://doi.org/10.1109/cmi48017.2019.8962137, http://dx.doi.org/10.1109/CMI48017.2019.8962137

[20] Guri, M., Kachlon, A., Hasson, O., Kedma, G., Mirsky, Y., Elovici, Y.: Gsmem: Data exfiltration from air-gapped computers over GSM frequencies. In: 24th USENIX Security Symposium (USENIX Security 15). pp. 849–864. USENIX Association, Washington, D.C. (Aug 2015), https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/guri

[21] Guri, M., Zadov, B., Bykhovsky, D., Elovici, Y.: Powerhammer: Exfiltrating data from air-gapped computers through power lines (2018)

[22] Guri, M., Zadov, B., Daidakulov, A., Elovici, Y.: Odini : Escaping sensitive data from faraday-caged, air-gapped computers via magnetic fields (2018)

[23] Hanspach, M., Goetz, M.: On covert acoustical mesh networks in air (2014)

[24] Hanspach, M., Goetz, M.: Recent developments in covert acoustical communications. Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI) pp. 243–254 (01 2014)

[25] Kuhn, M.: Compromising emanations: Eavesdropping risks of computer displays (04 2004)

[26] Kuhn, M.G.: Compromising Emanations: Eavesdropping Risks of Computer Displays. Ph.D. thesis, Technical Report Number 577. University of Cambridge (2003)

[27] Kuhn, M.G.: Electromagnetic eavesdropping risks of flat-panel displays. In: Martin, D., Serjantov, A. (eds.) Privacy Enhancing Technologies. pp. 88–107. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)

[28] Kuhn, M.G., Anderson, R.J.: Soft tempest: Hidden data transmission using electromagnetic emanations. In: Aucsmith, D. (ed.) Information Hiding. pp. 124–142. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)

[29] Larimer, J.: An inside look at stuxnet. IBM X-Force pp. 1–37 (2010)

[30] Lynn, F.: Defending a new domain: The pentagon's cyberstrategy. Foreign Affairs **2010**, 13 (09 2010)

[31] Marinov, M.: Remote Video Eavesdropping using a Software Defined Radio Platform. Ph.D. thesis, MA thesis. University of Cambridge (2014)

[32] van Eck, W.: Electromagnetic radiation from video display units: An eavesdropping risk? Computers Security **4**(4), 269 – 286 (1985). https://doi.org/https://doi.org/10.1016/0167-4048(85)90046-X, http://www.sciencedirect.com/science/article/pii/016740488590046X

A QR code capacities

The following table shows the maximum capacity of different QR code versions using ECL levels L and H and for the different modes. We omitted the capacities for levels M and Q, and we just included a limited number of versions. For more details about the capacities of each version we suggest [6].

Table 4: Modules and capacities of different QR code versions for ECL L and H when using different modes.

Version	Modules	ECL	Numeric	Alphanumeric	Bytemode	Kanji
1	21x21	L	41	25	17	10
		H	17	10	7	4
2	25x25	L	77	47	32	20
		H	34	20	14	8
3	29x29	L	127	77	53	32
		H	58	35	24	15
4	33x33	L	187	114	78	48
		H	82	50	34	21
5	37x37	L	255	154	106	65
		H	106	64	44	27
40	177x177	L	7089	4296	2953	1817
		H	3057	1852	1273	784