TEMPEST radio station

Paz Hameiri

https://il.linkedin.com/in/paz-hameiri-251b11143

Abstract

TEMPEST is a cyber security term that refers to the use of electromagnetic energy emissions generated by electronic devices to leak data out of a target device. The attacks may be passive (where the attacker receives the emissions and recovers the data) or active (where the attacker uses dedicated malware to target and emit specific data). In this paper I present a new side channel attack that uses GPU memory transfers to emit electromagnetic waves which are then received and processed by the attacker. Software developed for this work encodes audio on one computer and transmits it to the reception equipment positioned fifty feet away. The signals are received and processed and the audio is decoded and played. The maximum bit rate achieved was 33kbit/s and more than 99% of the packets were received. Frequency selection not only enables maximization of signal quality over distance, but also enables the attacker to receive signals from a specific computer when several computers in the area are active. The software developed demonstrates audio packets transfers, but other types of digital data may be transmitted using the same technique.

Introduction

Electronic circuits emit electromagnetic waves. Copper traces on printed circuit boards and wires act like antennas as they emit electromagnetic waves generated by the electric current flowing through the conductors.

TEMPEST (Telecommunications Electronics Materials Protected from Emanating Spurious Transmissions) is a U.S. National Security Agency (NSA) specification and a NATO certification. The acronym refers to information leakage from a system through unintentional radio signals, audio signals, electrical signals, etc. TEMPEST attacks were brought to public attention by Eck [1] in 1985. In a later work by Kuhn and Anderson [2], techniques that enable software to use the video card to control electromagnetic radiation emitted from its VGA lines are discussed. The attacker installs malware in the target computer which then manipulates the signals to emit targeted data. These are picked up by the attacker.

Thiele [3] demonstrated how audio can be transmitted from a target computer using software which generates patterns from the output of the graphics card. The audio is received and processed by an AM radio receiver. Subsequently, Kania [4] wrote a software program that transmits audio via VGA output which is received and processed by an FM radio receiver. Guri et al [5] demonstrated a TEMPEST attack by using the output of a graphics card to generate signals which were turned into electromagnetic waves by the video cables. Three types of output were examined – VGA, HDMI and DVI.

Proposed side channel attack

The side channel attack proposed by this work uses GPU memory transfers to emit electromagnetic waves which are then picked up by the attacker's antenna. These are converted to electric signals that are amplified by a low noise amplifier (LNA). The amplified signals are received by a software-defined radio (SDR) receiver and the leaked data is recovered by software on the attacker's computer. The end-to-end setup is shown in Figure 1.

Software named "Scotty" was written to transmit audio from the target computer. The software performs the following tasks:

1. Measuring the time required to perform large GPU memory transfers
2. Calculating the amount of data required to perform a memory transfer in a pre-defined amount of time. This amount of data shall hereafter be defined as the "MemoryBlockSize"
3. Setting memory clock frequency

4. Loading a WAV file and transmitting audio packets in one-second cycles in the following manner:
   4.1. Taking a predefined amount of PCM samples and encoding each sample with a G.726 audio encoder to reduce the number of bits per sample
   4.2. Creating an audio data packet comprised of a packet counter and the encoded audio data
   4.3. Calculating an error detection value for the audio packet and adding it to the audio packet
   4.4. Using a Reed-Solomon forward error correction (FEC) encoding algorithm to calculate parity data for the audio packet
   4.5. Transmitting the data in the following order:
      - Header data
      - Reed-Solomon parity data
      - Audio packet data
   4.6. When a specific amount of data intended to be tranmitted during a one-second interval has been transmitted, the software stops transmitting data and waits for the one-second interval to elapse before starting a new packet transmission cycle.

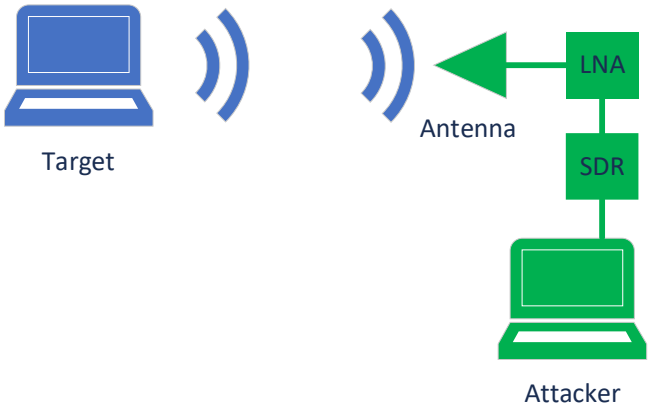

Figure 1: Radio setup

Data transmission is done in the following manner:

1. A symbol is built from a pre-defined amount of data bits
2. For a given symbol, the amount of data that needs to be copied is calculated by multiplying (symbol value + 1) by MemoryBlockSize
3. The GPU memory transfer is performed.

Each memory write cycle generates clock signals, control signals and data signals. An example of GDDR6 timing diagram is shown in Figure 2. Every signal passing in a PCB trace emits energy. The data transfers are performed by a direct memory access (DMA) mechanism in a repetitive process per unit time. For each symbol transferred, electromagnetic energy is transmitted during the time frame of the memory transfer. A high bandwidth memory transfer involves the use of several memory components, generating a given amount of energy per component and a larger amount of energy per graphics card. If a memory transfer is not performed, data is not written and energy is not emitted. This bi-state energy transmission is called on/off keying modulation, or OOK. An example of symbol emission versus time is shown in Figure 3.

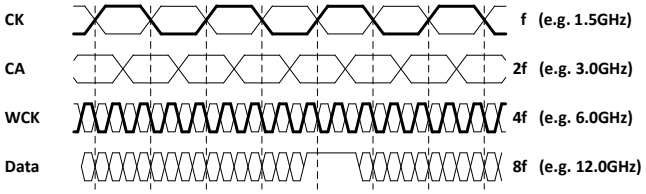

Figure 2: GDDR6 memory timing diagram

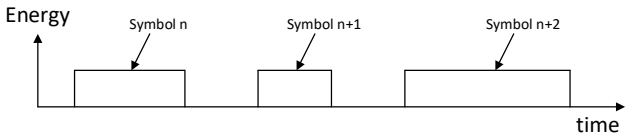

Figure 3: Electromagnetic wave emission vs time

Before the attack, the attacker selects the designated clock frequency. To maximize signal quality over distance, the attacker tries to select a bandwidth with as little interference as possible. Frequency selection not only maximizes signal quality and transmission distance, but also enables the attacker to receive signals from a specific computer in the case where there are other computers in the area that are active.

I wrote a software program named “Spock” to receive the emitted signals and recover the audio. The software performs the following tasks:

1. Setting up the SDR receiver
2. Receiving cyclic batches of I-Q samples from the SDR receiver
3. Calculating the absolute amplitude of the I-Q received vector
4. Filtering the data with a low pass filter
5. Calculating amplitude thresholds to recover the bits from the filtered data
6. Recovering the symbols using the calculated amplitude thresholds and a minimum time threshold (to filter short-term noise).
7. Saving the length of each symbol in a buffer.

When the symbol length buffer holds enough symbols to process a data packet, the buffer content is analyzed in the following manner:

8. Searching a series of symbols lengths which are proportional to the header symbol’s length
9. Recovering the basic time unit which the transmitter takes to transmit a single MemoryBlockSize

10. Recovering the series' symbol values and verifying that they are equal to the header data
11. Recovering the data packet from the rest of the symbols (Reed-Solomon parity data and audio packet data)
12. Using a Reed-Solomon forward error correction decoding algorithm to recover the received audio packet data
13. Calculating the error detection value for the audio packet and verifying that it is equal to the value calculated by the transmitting software. If the numbers are equal – a good packet was received.
14. Examining the packet counter. If no packets from the last packet reception were missed, the software decodes the audio PCM samples from the audio packet using a G.726 decoder. If packets were lost, the missing samples are replaced by a constant value
15. Storing the audio PCM samples in a buffer
16. Playing the audio PCM samples while filling the buffer with new incoming data.

Test setup

Tests were performed using two computers as transmitters:

1. An ASUS G731GU-BI7N9 laptop containing an nVIDIA GeForce GTX 1660 Ti GPU with dedicated 6GB GDDR6 RAM, an Intel i7-9750H processor and 16GB RAM installed
2. A desktop computer containing an ASUS TUF-GTX1650S-4G-GAMING card (nVIDIA GeForce GTX 1650 SUPER GPU with a dedicated 4GB GDDR6 RAM installed) and an ASUS B150M-C motherboard with an Intel i7-6700K processor and 16GB RAM installed.

As free space path loss is proportional to signal frequency, the memory clock frequency was selected for reception. Transmission parameters were set:

- Memory clock frequency = 1566.75 MHz Frequency was set and measured using nVIDIA's NVAPI
- MemoryBlockSize transmission time = 14 µsec
- Symbol size = 4 bits per symbol.

The memory clock frequency was selected after a site survey showed minimum interference at a bandwidth of 10 MHz. The site survey was conducted using the reception equipment and Steve Andrew's "RSP Spectrum Analyser" software.

For MemoryBlockSize transmission time = 14 µsec, the minimal symbol length equals 14 (1 * 14) µsec, and the maximal symbol length equals 224 µsec (16 * 14). Memory transfers were made with nVIDIA CUDA's cudaMemcpy command, followed by a cudaDeviceSynchronize command. The average time measured between adjacent symbols and byte transmission time range is shown in Table 1.

Computer	Average time between adjacent symbols [µs]	Shortest byte transmission time [µs]	Longest byte transmission time [µs]
Laptop	36.7	101	521
Desktop	61.4	151	571

Table 1: Average time between adjacent symbols and byte time range

The Reed-Solomon FEC properties were set to:

- Block size = 255
- Data bytes = 235
- Parity bytes = 20

The transmission structure was comprised of 90 bytes:

- 4 Header bytes
- 20 Reed-Solomon parity bytes
- 1 Packet counter byte
- 63 Encoded audio bytes
- 2 Checksum bytes

Audio properties were set to:

- 8000 PCM samples per seconds
- G.726 encoding = 2 bits per sample

On average 2 packets were transmitted every second, each packet delivering 252 encoded PCM samples. The last packet was set to transfer 47 bytes instead of 63 to yield a total of 8000 encoded PCM samples per second.

Radio frequency equipment used:

- A 1.35GHz-9.5GHz 8x10cm Log Periodic Antenna with 5-6db gain
- 1MHz to 3GHz low noise amplifier with 20dB gain and 2dB noise factor. Gain at 1.5GHz: 16dB
- SDRplay RSP1A receiver.

Test results

The transmitting computer and the reception equipment were placed 50 feet apart and at opposite ends of an apartment. The transmitting desktop computer was placed with its rear side facing the reception equipment as it was found that it emits the most energy from that side. The transmitting laptop computer was placed with its front side facing the reception equipment, as both the front and rear sides emitted significant energy. The desktop computer setup is shown in Figure 4. The reception equipment is shown in Figure 5. The space between the transmitting computer and the reception equipment is show in Figure 6.

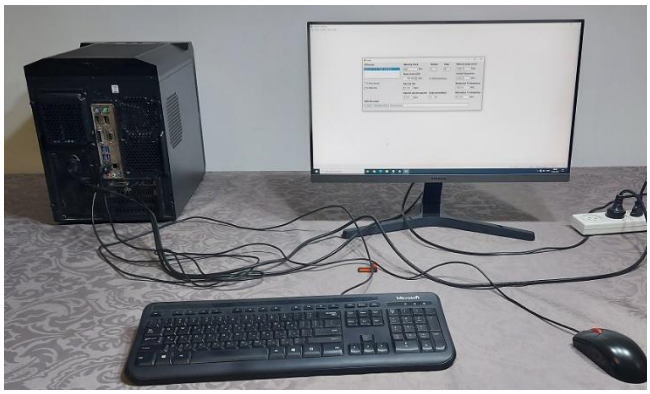

Figure 4: Desktop computer setup

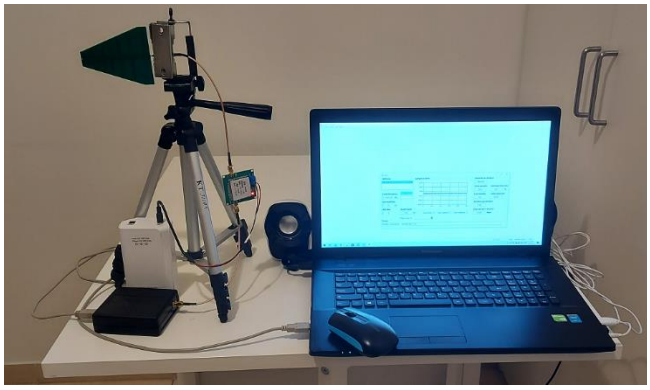

Figure 5: Reception setup

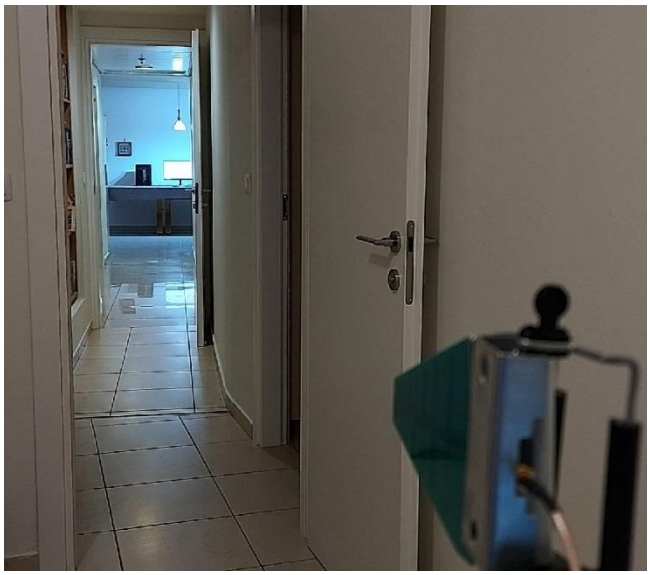

Figure 6: Line of sight between the transmitting computer and the reception equipment

The received clock signal from the desktop computer is shown in Figure 7. The signal is generated by a "spread spectrum clock generator." The clock signal is frequency modulated to "spread" radiated energy across a frequency band. The goal of this method is to comply with electromagnetic compatibility regulations. A sample of the reception process stages is shown in Figure 8.

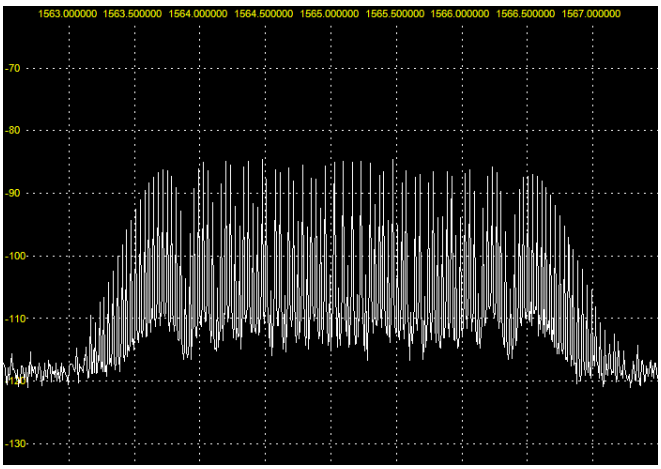

Figure 7: Received clock signal 50 feet away from the desktop computer. Frequency span between 1562.5 MHz and 1567.5 MHz.

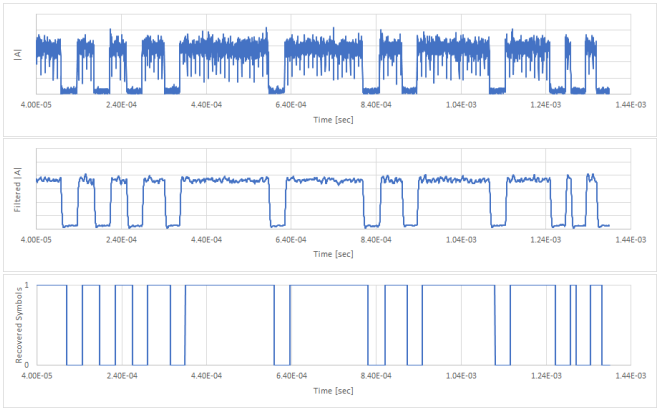

Figure 8: A sample for received and processed symbols. The upper graph shows sample of the absolute value of the received vector. The middle graph shows the data after filtering. The lower graph shows the symbols recovered from the data.

Measured average packet bit rate and packets reception quality are shown in Table 2. For valid packet percentage measurements, two results were documented – one when the monitor was on and one when the monitor was off (following Windows' power saving plan). In both cases the recovered audio quality was good, considering the sampling rate and G.726 codec.

Computer	Average bit rate [kbit/s]	Valid packets received with the monitor turned on [%]	Valid packets received with the monitor turned off [%]
Laptop	26	> 99	Irrelevant
Desktop	23	89.5	> 99

Table 2: Average bit rate and average valid packets received at 50 feet

To achieve best audio quality, the following setup changes were made:

- MemoryBlockSize transmission time = 8 µsec
- Reed-Solomon data bytes = 251
- Reed-Solomon parity bytes = 4
- G.726 encoding = 2 bits per sample

On average eight packets were transmitted every second, each packet delivering 168 encoded PCM samples. The last packet was set to transfer 39 bytes instead of 63, yielding a total of 8000 encoded PCM samples per second.

Measured average packet bit rate and packet reception quality are shown in Table 3. In both cases the recovered audio quality was good, considering the sampling rate and G.726 codec.

Computer	Best audio quality setup average bit rate [kbit/s]	Valid packets received with the monitor turned on [%]	Valid packets received with the monitor turned off [%]
Laptop	33	> 99	Irrelevant
Desktop	30	Low	> 99

Table 3: Best audio quality average bit rate and average valid packets received at 50 feet

Discussion

The tests showed good results when the distance between the transmitter and the reception equipment was fifty feet and within a line of sight. It was not tested at longer distances due to the physical limitations of the facility. Tests performed at different locations in the facility showed similar results at distances smaller than 50 feet.

Tests also showed that the desktop computer emitted signals which were not generated by Scotty. The laptop computer did not emit such signals. These signals are most likely related to the display activity, as the computer stops transmitting these signals once the monitor is turned off by the Windows power plan. The laptop uses Windows Intel's integrated, power-saving GPU for most applications. It automatically switches to the discrete, high-performance GPU when an application requires more performance. The desktop computer uses discrete GPU by default. To verify this, when the desktop computer was booted with the monitor connected to the integrated GPU HDMI output, the interferences were not received and the ratio of good packets exceeded 99% with the monitor on.

This work demonstrates transmission of audio data which can be done continuously without stop, 24/7. The same technique may be used to extract other types of data from a target computer 24/7. Thus, an attacker may choose to extract the desired data during non-working hours when the monitor is turned off, or the GPU is idle and there's no close supervision of computer activities. If the monitor is not turned off by the Windows power saving plan, the attacker may turn it off at the time of the attack.

Future directions

This work focused on reception of electromagnetic waves emitted by the base memory clock signal of the discrete GPU memory. Other signals should also be explored as well. An example can be seen in Figure 9 where the reception frequency was set to the GDDR6 clock frequency divided by half. In this case the signal was received at close range and processed by the Spock software with good results. But, as data signals have different properties than the clock signal, the software requires adaptation to achieve best results.

The potential of emitting a signal whose frequency is correlated to the data itself should also be explored. Besides the benefit of expanding the number of transmitted signals and controlling their frequencies, it may enable the use of data patterns for frequency-hopping spread spectrum (FHSS) transmission.

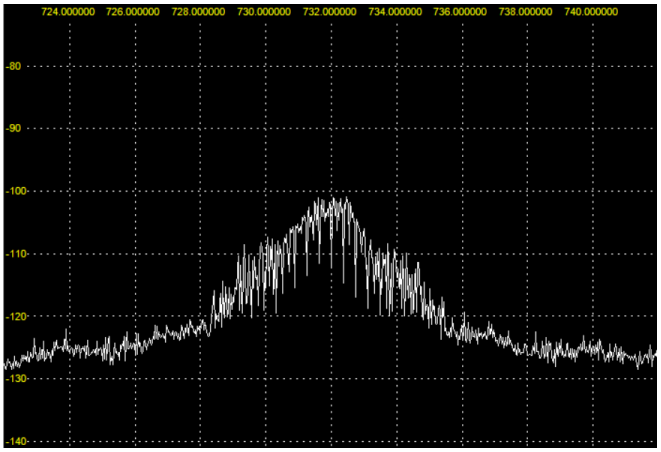

Figure 9: Received signal from the laptop computer at 732MHz (Transmission clock set to 1465.25). Frequency span between 722 MHz and 742 MHz.

Conclusions

In this paper a new side channel attack is proposed. Dedicated software encodes audio on the target computer and then uses controlled graphics card memory transfers to generate electromagnetic waves. These waves are transmitted from the target computer. The attacker's computer receives the electromagnetic waves using an antenna, an LNA and an SDR receiver. Dedicated software processes the data and recovers the audio. Both data transmission and data recovery techniques were explained and the test results were presented. Finally, the results were discussed and future directions were suggested.

Acknowledgements

I would like to thank my wife and our children for supporting my endeavor. I would also like to thank you, the reader, for your interest in my work.

References

[1] Eck W. “Electromagnetic radiation from video display units: an eavesdropping risk?” *Computers and Security*, 4, no. 4: 269-286, 1985.

[2] Kuhn, M. G., and Anderson, R. J. Soft. “Soft Tempest: Hidden Data Transmission Using Electromagnetic Emanations.” In *Information Hiding* (1998), ed. D. Aucsmith, vol. 1525 of *Lecture Notes in Computer Science*, (Springer): 124–142.

[3] Thiele, E., “Tempest for Eliza.” 2001. http://www.erikyyy.de/tempest/.

[4] Kania B., “VGASIG: FM radio transmitter using VGA graphics card.” 2009. http://bk.gnarf.org/creativity/vgasig/vgasig.pdf.

[5] Guri M., Kedma G., Kachlon A., Elovici Y. “AirHopper: Bridging the air-gap between isolated networks and mobile phones using radio frequencies.” In *Malicious and Unwanted Software: The Americas (MALWARE)*, 2014 9th International Conference on IEEE, 2014: 58-67.