2015 IEEE 28th Computer Security Foundations Symposium

BitWhisper: Covert Signaling Channel between Air-Gapped Computers using Thermal Manipulations

Mordechai Guri
Department of Information Systems Engineering
Ben-Gurion University of the Negev
Beer-Sheva, Israel
gurim@post.bgu.ac.il

Matan Monitz, Yisroel Mirski, Yuval Elovici
Department of Information Systems Engineering
Telekom Innovation Laboratories at Ben-Gurion University
Beer-Sheva, Israel
{monitzm,yisroel}@post.bgu.ac.il, elovici@bgu.ac.il

***Abstract*— It has been assumed that the physical separation ('air-gap') of computers provides a reliable level of security, such that should two adjacent computers become compromised, the covert exchange of data between them would be impossible.**

In this paper, we demonstrate *BitWhisper*, a method of bridging the air-gap between adjacent compromised computers by using their heat emissions and built-in thermal sensors to create a covert communication channel. Our method is unique in two respects: it supports bidirectional communication, and it requires no additional dedicated peripheral hardware. We provide experimental results based on the implementation of the BitWhisper prototype, and examine the channel's properties and limitations. Our experiments included different layouts, with computers positioned at varying distances from one another, and several sensor types and CPU configurations (e.g., Virtual Machines). We also discuss signal modulation and communication protocols, showing how BitWhisper can be used for the exchange of data between two computers in a close proximity (positioned 0-40 cm apart) at an effective rate of 1-8 bits per hour, a rate which makes it possible to infiltrate brief commands and exfiltrate small amount of data (e.g., passwords) over the covert channel.

Keywords—air-gap; bridging; covert channel; temperature; sensors; exfiltration; infiltration (key words)

I. INTRODUCTION

An air-gapped network is a computer network in which security measures are taken to maintain physical and logical separation from other, less secured networks. The term *air-gap* may sometimes refer to an interface between two systems or networks in which data transfers are handled manually (e.g., a human operator copies data from one system onto a flash-drive and then walks over to another system and connects the flash-drive to it [1]). Air-gapped networks are often used in situations in which the information stored or generated by the network is highly sensitive or at risk of data leakage. For instance, military networks such as the Joint Worldwide Intelligence Communications System (JWICS) are air-gapped networks [2] [3]. Despite the added security benefits of an air-gapped network, such networks have been breached in recent years. The most famous cases are Stuxnet [4] and agents.btz [5], although other cases have also been reported [6].

Despite these attacks, air-gapped networks are still widely used because they minimize the risk of data leakage and prevent malicious data from being transmitted to the network.

A. Covert Channels and Related Works

Covert channels have been widely discussed in professional literature [7] [8] [9]. However, in this paper, we focus on covert channels that can exfiltrate data from air-gapped – or physically separated – computers. This subset of covert channels employs several physical phenomena, including the use of acoustic inaudible channels, optical channels, and electromagnetic emissions. Madhavapeddy et al [10] discuss so-called 'audio networking', while Hanspach and Goetz [11] disclose a method for near-ultrasonic covert networking using speakers and microphones. Loughry and Umphress [12] discuss information leakage from optical emanations of LED status indicators on communication equipment. Software-based hidden data transmission, using electromagnetic emissions from the video display unit, is discussed by Kuhn and Anderson [13]. Guri et al [14] present 'AirHopper', a malware that utilizes FM emissions to exfiltrate data from an air-gapped computer to a nearby mobile phone. Callan et al [15] present a method for measuring the electromagnetic radiation emitted from instruction-level events using specialized hardware. Note that the aforementioned channels are unidirectional and do not permit the establishment of an inbound channel into an isolated network (in the case of an ultrasonic covert channel, a microphone has to be installed on the receiver PC). Technically speaking, thermal radiation is a form of electromagnetic emanation, and this was, in fact, previously suggested by Murdoch [16] for use as a covert channel (discussed in the 'future work' section). To the best of our knowledge, our current paper is the first to present academic research with successful and detailed results in this respect.

B. Our Contribution

In this study, we present *BitWhisper*; a novel method for exchanging information between two adjacent personal computers (PCs) that are part of separate, physically unconnected networks. BitWhisper establishes a covert channel by emitting heat from one PC to the other in a controlled manner. By regulating the heating patterns, binary data is modulated into thermal signals. In turn, the adjacent PC uses its built-in thermal sensors to measure the environmental changes. These changes are then sampled, processed, and demodulated into binary data.

DOI 10.1109/CSF.2015.26

Our experimental results demonstrate that BitWhisper is capable of a rate of eight signals per hour. Although this rate may seem slow compared with the previously mentioned methods of bridging air-gaps, BitWhisper has two unique and useful characteristics: 1) the channel supports bidirectional (half-duplex) communication as both PCs can act as a transmitter (producing heat) or receiver (by monitoring the temperature), and 2) establishing the channel is possible using off-the-shelf adjacent desktop PCs and requires no special hardware or supporting components. These properties enable the attacker to exfiltrate information from inside an air-gapped network, as well as transmit commands to it. The attacker can also directly control a malware's actions inside the network and receive feedback.

The remainder of this paper is structured as follows. Section II discusses the attack scenario, and section III provides technical background. Sections IV and V present the experimental setup and experimental results respectively. Section VI discusses the data modulation and communication protocol. Section VII discusses countermeasures, and we conclude with section VIII.

II. Attack Scenario

BitWhisper, as a generic covert channel, can be utilized for various purposes. However, we explore its use as a method for bridging the air-gap between physically separated networks. In this section we discuss a possible attack model in which BitWhisper is used by an attacker to achieve this goal. The environmental settings are comprised of a facility or office space in which PCs are either part of an isolated network or a public network, such as the Internet. The attack model consists of several phases. During the first phase, the attacker infects networks connected to the Internet, a part of the attack which can be accomplished using targeted malicious emails combined with social engineering and other similar methods. During the second, considerably more challenging phase, the attacker goes on to contaminate a node of the internal network. This can be done by attacking the supply chain [17], planting an infected USB drive, using a malicious insider, or employing an equivalent tactic. Several recent incidents have shown that this type of breach is feasible [4] [5] [6] [18].

Having established a foothold in both networks the attacker then wants to bridge the air-gap between the networks in order to covertly exfiltrate a highly sensitive piece of information (e.g., passwords, or secret keys). For example, the attacker may trigger a worm attack inside the isolated network or send a malicious command to an isolated industrial control system. At this point, the malware spreads across both networks and searches the surroundings for PCs in close proximity, spatially. Proximity is determined by periodically sending 'thermal pings' (demonstrated in Figure 1) over the air and waiting for a thermal ping response. Once a bridging attempt is successful, a logical link between the public network and the internal network in established. At this stage, the attacker can communicate with the formerly isolated network, issuing commands and receiving responses.

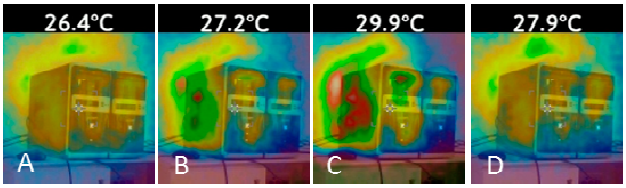


Figure 1: A "thermal ping" sent between two adjacent PCs. The snapshots were taken by using a thermal camera.

III. Technical Background

In this section, we provide a technical survey of the thermal emissions of PCs, along with methods used to cool them. This review will assist the reader in understanding how heat from one PC is emitted and then detected by an adjacent PC in a controlled manner.

A. Ambient Heat Sources in a PC

Like many electrical systems, PCs generate heat. The law of conservation of energy states that energy is conserved over time; excess power dissipates as heat, primarily in a physical process called Joule heating. Joule heating (also termed resistive heating) occurs when the passage of an electric current through a conductor releases heat. The generated heat is proportional to the current and voltage of the system [19]. Complex electronic systems such as the central processing unit (CPU) of a modern PC requires varying amounts of power (current and voltage) proportional to the workload of the system. This workload directly affects the amount of heat generated by the system. The dynamic power consumption of a CPU is generalized by the formula

$$P \cong\propto CfV^2 \qquad (1)$$

Where P is the power consumption in watts, C is the capacitance of the CPU in farads, f is the operating frequency in Hz, V is the voltage in volts, and $\propto$ is the percentage of the system that is active [20]. When the system is under load and executing different instructions, $\propto$ clearly increases as does the capacitance C, because more switching gate capacitors need to be charged. Higher operating frequencies also require quicker charging and discharging of logical gates, achieved by increasing the CPU voltage and frequency. To summarize, an increased CPU workload on a modern system causes increased power consumption across all parameters, thereby increasing the temperature as well.

In addition to the CPU, modern computers also contain other components that emit a significant amount of heat, including the graphics-processing unit (GPU) and other motherboard components such as the voltage regulator modules (VRM) and I/O controllers. Other heat sources in a computer include mechanical systems such as a hard drive or optical drive.

B. Thermal Sensors in PCs

Electronic systems like computers and smartphones incorporate multiple thermal sensors to monitor their various components and ambient temperature. Thermal sensors allow the system to protect itself from damage or performance degrada-

tion by decreasing the system's workload, activating external cooling systems such as fans, or even performing emergency shutdowns. In this section, we briefly survey common thermal sensors found in modern computer systems. Later, in section VI, we describe our use of some of these sensors to receive thermal signals transmitted through the air.

1) CPU/GPU thermal sensors

CPUs and GPUs contain multiple built-in temperature sensing diodes. In the case of CPUs, these diodes enable the system to measure the internal temperature of each core of the CPU separately. In addition to the CPU core's internal sensors, some processors include sensors that monitor the outside of the CPU package. In a typical PC, the difference between the internal temperature of a CPU and the ambient temperature inside the PC's chassis may be tens of degrees. Interestingly, CPU sensors are designed to be more accurate at higher operating temperatures, while being notoriously inaccurate at lower temperatures [21].

2) Motherboard thermal sensors

Motherboards usually incorporate thermal sensors and cooling fan controls in order to activate buzzer alarms or emergency shutdowns at extreme temperatures, thus preventing damage to the computer's sensitive components. Sensing is usually performed by the I/O controller for easy monitoring by an application. It is common to place these sensors in locations that are relatively distant from heat sources, in order to monitor the motherboard's temperature itself. In many motherboard models, the overheat threshold (the CPU temperature at which the computer shuts down to prevent damage) can be changed in the BIOS.

3) Other thermal sensors

Voltage regulators in a PC supply the required voltage to its CPUs and GPUs. Voltage converters generate a significant amount of heat and require thermal sensing and control. Some computer cases have fans and other cooling methods intended to cool a specific component while dissipating heat from the entire chassis. To that end, thermal sensors are placed in the path of the airflow created by these fans or on the chassis itself. Most hard disk drives and solid-state drives have built-in temperature sensors. Diagnostic software can access these sensor readings through an interface called the Self-Monitoring Analysis and Reporting Technology (S.M.A.R.T) interface [22].

Table 1 summarizes the thermal sensors and their primary characteristics.

Table 1: Survey of thermal sensors (common values)

Sensor	Idle temperature (℃)	Max temperature (℃)	Sensor resolution
CPU	33-35	130	1℃ degree
GPU	33-35	105	1℃ degree
Motherboard	33-35	90	1℃ degree
Voltage regulator	31-33	90	1℃ degree
Hard-disk drives	35	55	1℃ degree
Solid-state drives	25-30	70	1℃ degree

C. Cooling Mechanisms for PCs

Some components of a computer may become temporarily unusable or damaged permanently if not cooled properly. For stable continuous operation of the PC, these components must receive sufficient cooling to keep the temperature within specifications and prevent damage. Since most cooling methods aim at dissipating generated heat into the computer's surroundings, they are an important part of the thermal channel. There are two methods of cooling: passive and active.

Passive methods cool the component by letting the heat dissipate naturally into the air on its own or with the help of a heat-sink (to conduct and disperse the heat via direct contact). For most chips found in a PC, this is the cooling method of choice. However, some chips may generate enough heat under regular usage to damage themselves. In these cases, passive cooling is not sufficient.

Active methods attempt to speed up the convection process by including a fan or possibly other mechanisms that involve liquid or gaseous coolants. The most common active cooling method found in PCs is to couple a fan with a heat-sink, because it is inexpensive and effective. While rotating, a fan helps divert the heat absorbed by the heat-sink from the heat emitting component and dissipate the hot air out of the case. The effectiveness of the cooling process depends on the amount of heat generated by the component and the size and rotational speed of the fan.

In a typical PC, a fan is placed on the CPU and in the power supply. Other common configurations include the placement of fans on the left or back side of the case. In some cases fans can be found on graphic cards, the motherboard's chipset, or hard drives. Since cold air is denser than hot air, cold air moves downward, while hot air is pushed upward. Because of this fact, the fan configurations in a PC are typically designed to bring in cold air from the front of the case (at the bottom) while pushing the hot exhaust out from the back (at the top). Fans are either wired to the motherboard through dedicated connections or directly to the power supply with a 3-pin connector. For speed control, fans may be wired to the motherboard or to an external controller with a 4-pin connector. Typical fan speeds ranges from a few hundred revolutions per minute (RPM) to a few thousand RPMs. In modern systems the fan speed can be controlled from the OS by appropriate software or API, through OS device manager services, or via the computer's BIOS interface. Another dominant factor affecting fan speed is the current CPU temperature. When the temperature is not high, the fan controller reduces its speed for quieter operation, increased fan lifespan, and lower power consumption. Table 2 presents some of the

characteristics of fans commonly found in PCs.

Table 2: Common PC fans

Location	Primary role	Optional	RPM
Motherboard (CPU socket)	cools the CPU	No	800-4000
Front bottom	pulls cold air in from outside	Yes	800-4000
Rear top	cools power supply and expels exhaust	Yes	800-4000
Side	cools GPU and drives	Yes	800-4000

IV. Physical Channel Experimental Setup

In this section, we discuss our experimental setup which is designed to evaluate heat emissions as a viable signaling channel between two PCs. Since the proposed channel is half-duplex, we will evaluate one direction of a communication at a time. Therefore, for the duration of this paper, we refer to the PC that emits the heat as the '*sender'* or '*transmitter'* and the PC that monitors the environmental temperature as the '*receiver'*. A channel configuration reflects the technical and spatial configurations that two communicating PCs may have, directly affecting the channel's performance.

A. Experimental Methodology

The experiments involved two types of setups: a single PC setup or a setup involving a pair of PCs. With the single PC, we evaluated the thermal properties (e.g., heating and cooling rates) of an active PC under different workloads. In the setup involving two PCs, we evaluated the mutual thermal effects between adjacent computers First, using the single PC setup, we performed a series of CPU workload trials. The objective was to understand the amount of heat that was effectively generated and how quickly this heat dissipates, as well as how these changes registered in the different sensors. For each trial, we generated different workloads and monitored the local heat from the same computer's sensors. Afterwards, using the two PC configurations, we examined the effect of the transmitter's heat on the receiver and its environmental sensors. Several series of trials were performed to analyze the many possible combinations that may affect the channel. For instance, some of the parameters analyzed were the transmitter/receiver's relative distance, chassis type, and relative layout. Lastly, we tested the feasibility of emitting heat from within a virtual machine guest OS. We found that although the physical CPU is controlled by the host OS, the guest OS can use it indirectly as a thermal transmitter.

B. Experiment Setup

1) Test environment

Tests were conducted in two environments: 1) a regular office room (3 by 4 meters) with standard furniture, no windows, and a closed door, and 2) a larger shared office space (4 by 9 meters) with a small window and an open door. The computers were either placed above or below a table or inside a computer desk tray. An additional experiment was conducted in an open room environment (described later in section V). Environmental temperature during the tests was controlled using the office's air-conditioning system.

2) Hardware

We used two sets of PCs for the experiments. The first set consisted of PCs with a GIGABYTE GA-H87M-D3H motherboard and a quad-core hyper-threaded Intel i7-4790 processor with a built-in Intel HD Graphics 4600 display adapter. Both computers had an identical, regular no-brand tower case, with one fan in the back of the case in addition to the CPU and PSU fans. The chassis had a perforated square area on the left side, in front of the CPU fan, and a perforated region near the back and PSU fans. The second set of PCs consisted of Lenovo m57-6072 computers with an Intel Q35 Express chipset, an Intel core 2-duo processor with an integrated display adapter. The Lenovo 6072 had a small-form-factor case with a perforated front and back panels with a single fan blowing air out of the back. We also conducted tests with a large tower case (measuring 9X48X41 cm) manufactured by Gigabyte from the GZ series with a H77-D3H motherboard, and a quad-core hyper-threaded Intel i7-3770 with a built-in Intel HD Graphics 4000 display adapter. For most of the tests, fan speeds were controlled by the default settings of the motherboard. The impact of environment temperatures on fan speeds was tested using an AK-FC-08-AT Asetek fan controller kit. The virtual machine heat emission experiment was conducted using Oracle VirtualBox. The environmental temperature and heat expansion around the PCs were captured by a Fluke VT04A Visual IR thermometer and a FLIR T335 thermal imaging camera.

3) Software

In order to generate heat, we performed CPU and GPU stress tests. Our code placed heavy stress on the processors by performing CPU intensive calculations and busy loops. We also used FurMark, a GPU stress tester [23] and prime95, a program for finding Mersenne primes that induces a high CPU load and is popular for CPU benchmarking and stress testing [24]. For data recording during the experiments we used a program called HWInfo [25] which sampled and recorded the various thermal sensors and PC fan speeds. The sample rate was 0.5Hz, and all monitored sensors were CPU and motherboard stock sensors. Using the 0.5Hz sample rate, HWInfo in itself utilizes very little CPU time and we validated that it had a negligible effect on the measurements.

4) Layouts and distances

We conducted tests with different layouts of adjacent pairs of PCs. The layouts reflected common scenarios in which two PCs are placed next to one another in typical office environments. Close proximity between PCs may occur in an office shared by two employees with adjacent desks or in a private office in which a single employee has more than one personal computer near his desk. The latter is common in work environments with air-gapped networks since security measures usually require the employee to have a separate computer for each network. In a typical workplace scenario, due to work

area sizes, two desktops are placed side by side (Figure 2) at distances ranging from 1 cm to 1 meter. Moreover, the PCs were generally positioned with the back panel facing the same direction. This layout is used in our experiments unless otherwise mentioned, and is referred to as the *parallel layout*. Other layouts tested include: 1) both PCs placed horizontally, one on top of the other, with the transmitting computer placed either on the top or the bottom (i.e., *stacked layout*); 2) two computers placed back-to-back, with the back panels facing each other; this layout reflects the common scenario of two employees sharing an office with desks facing each other (i.e., *face-away layout*), and 3) two computers positioned in some form of angle (i.e., *quadrature layout*). An illustration of the different layouts is given in Appendix A.

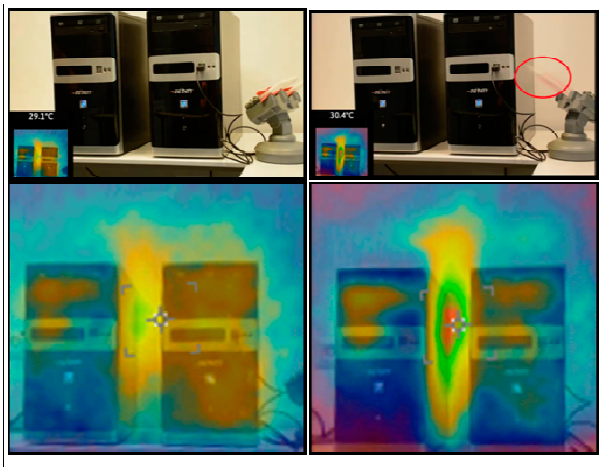


Figure 2: Two air-gapped PCs positioned in the parallel layout. The computer on the left transmits a command that instructs the computer on the right to calibrate and fire a USB game rocket.

V. Physical Channel Experimental Results

In this section, we discuss the physical characteristics of the channel obtained from the experiments described in section IV. Based on the experimental results, we developed a basic communication channel (section VI).

A. Internal Heat Emissions of a PC

The thermal properties of a PC (i.e., heating-rate versus CPU utilization) are important since they directly affect the signal quality. During our research, we identified three types of sensors that are significantly affected by the environment's temperature: A) the CPU's internal temperature sensor, B) other temperature sensors such as those in the CPU and HD, and C) the fan speed in RPM. As described in previous sections, a PC's workload has a direct effect on the temperature of its components. This subsection examines how and when heat is emitted from within a PC in order to understand how that may affect a thermal communication channel.

1) Warming and cooling

Figure 3 shows a PC's temperature captured by its various thermal sensors over 40 minutes of 100% CPU utilization followed by 40 minutes of nearly 0% utilization. It is apparent that CPU loads have a direct impact on the CPU core's thermal sensors (sensor group A) due to an almost instantaneous increase of about 20℃ in the temperature. The motherboard's temperature sensors (sensor group C) showed a +10℃ change over 30 minutes. The voltage regulator temperature remained static for the entire duration of the test and was removed from the figures in all further analysis. From a spatiotemporal perspective, the sensors closer to the heat source were affected to a greater extent. Note that it takes about 1.5-3 minutes of continuous work for the ambient temperature sensor to increase by 1℃. This rate decreases slightly as temperatures rise but then remains constant for a maximum of 10℃ above the idle temperature.

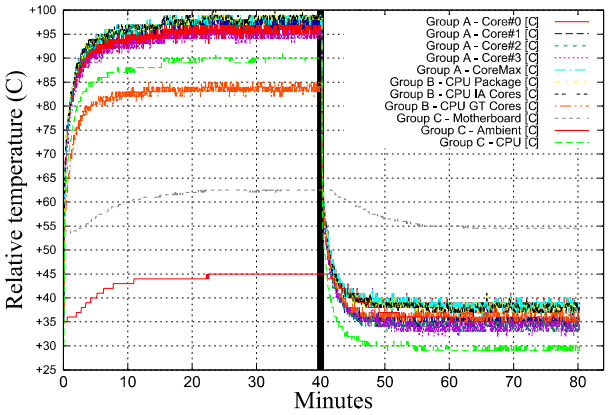


Figure 3: Temperature readings from various thermal sensors in a desktop PC as influenced by 100% CPU utilization over 40 minutes.

Once the full CPU utilization stops (minute 40), its internal temperature immediately drops. The ambient case temperature sensor shows that it took about 1-3 minutes for a drop of 1℃, with the rate decreasing slightly as it gets closer to the idle temperature.

In general, we observed that the PC heats up faster near its idle temperature and cools faster near its maximum temperature. This is because at higher temperatures, the fans rotate faster in order to conduct the heat at a faster rate.

2) Casual thermal emissions

It is important to measure the heat that a PC generates during casual work in order to anticipate its effect on the thermal communication channel. Obviously, a frequently changing PC workload can generate internal heat that interferes with the original heat transfer. Therefore it is clear that the optimal time to emit thermal signals would be when the PC is idle (e.g., during the night). However, in order to demonstrate the negligible effects casual usage has on the channel, we evaluated the system's thermal emissions during typical daily tasks.

Figure 4 shows a PC's internal ambient temperature alongside its CPU utilization and CPU internal temperature during 30 minutes of daily work. As a reference, we used a typical workstation with an Intel i7-4790 processor, running

Windows 7 64Bit. The user's activities include writing a document using Microsoft Word, browsing several websites concurrently using Google Chrome with multiple tabs, and watching video stream on YouTube. As can be observed from the test, none of the workloads accompanies by a rise in ambient temperature. The thermal level remained essentially the same while the ambient temperature was constant at 32℃. Therefore, it is feasible to use a workstation as a receiver during regular work hours, as long as the local CPU does not have an unusual workload. As can be noticed, the CPU heating sensor is much prone to interference by a user's regular work patterns.

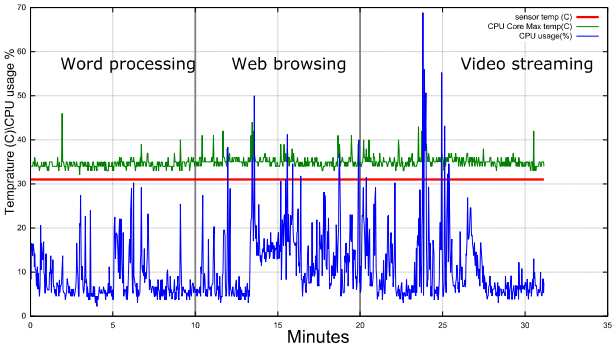


Figure 4: A PC's internal ambient temperature compared to its CPU internal temperature and utilization during routine use.

B. Two Party Thermal Channel

Thus far, we have shown that a PC's heat level can be controlled by regulating different workloads on the CPU. It is known that a PC's internal fans vent the heat exhaust. This heat is emitted to the immediate local vicinity and beyond. In the following subsections we show how the emitted heat affects nearby computer cases, thereby signaling other PCs via their thermal sensors

1) Sensor selection

A Single Input Multiple Output (SIMO) channel correlates a signal received from multiple vantage points, thereby boosting the signal with respect to natural noise and other interferences. It is possible to form a SIMO channel by correlating all of the thermal sensors together. However, for simplicity, we will only view the channel as a single input single output (SISO) channel. Improving the results (e.g., bandwidth) by using a SIMO channel will be addressed in future dedicated research.

A receiver PC on the channel output must monitor its thermal sensors continuously in order to identify changes in the ambient temperature. To maximize the signal quality, we performed experiments in order to identify the sensors which best capture the environmental temperature changes. Based on our experiments, we identified two important aspects of thermal sensors related to receiving signals on the channel's output: 1) the sensor's technical specifications, and 2) its location within the receiver.

Sensor specifications include: 1) sampling rate, 2) accuracy, 3) sample quantization resolution, and 4) value range (minimum and maximum temperature). Most sensors offer a sampling rate higher than required for accurate measurements. A more accurate sensor with higher quantization resolution can better distinguish between subtle changes in temperature level, thereby increasing the channel bit-rate. The range of the sensor must include the minimum and maximum temperatures used by the channel.

The experimental results indicate that the ambient sensor from sensor group B is the best sensor to use in the SISO channel. This is due to its accurate response to environmental temperature changes and the fact that it is relatively unaffected by the sharp CPU spikes caused by casual background processes. Some of the tests results appear in Figure 5.

We note that in cases in which a PC is unequipped with these thermal sensors (e.g., they are taken out or inaccessible), even the thermal sensor of the HD may be used to sense the channel's output.

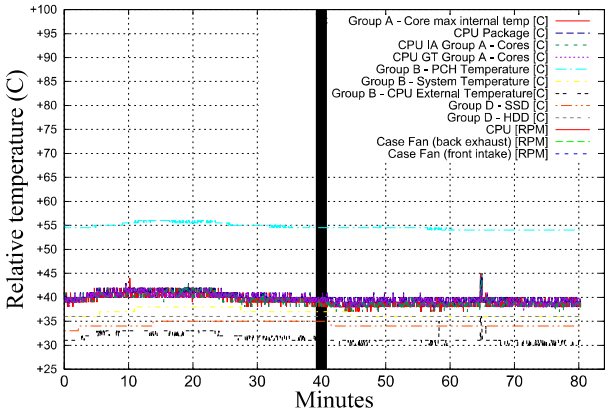


Figure 5: Temperature readings from various thermal sensors in a desktop PC as externally influenced by a transmitting computer over 40 minutes.

2) Distances

An important part of the channel setup is the distance between the transmitter and receiver. The effect of the heat emitted from a PC's exhaust ports has a limited radius, thereby limiting the maximum channel distance. Beyond that radius, the heat intensity fades and expands across the room. Moreover, in common with all physical communication channels, there is a propagation delay between a channel's inputs (transmission) and outputs (its reception some distance away). The propagation delay is directly correlated to the distance between the two parties. During the experiments we observed that the receiver was not able to sense the change in temperature at a distance of more than 40 cm from the transmitter.

In order to measure and evaluate channel properties, we had the transmitter send a steady signal using consistent heat generation, to a receiver computer. The receiver logged the observed temperatures, each time at an increased distance. In this experiment, we used the common parallel layout and conducted a series of trials at distances of 0, 5, 10, 15, 20, 25, 30, and 35 cm, and the results of the trials are documented in

Figure 6. We observed that at short distances, the transmitter could affect the receiver by 1-4℃. For example, when placed next to each other (~0 cm), the heat propagation delay for the first 1℃ increase was three minutes. After 26 minutes the receiver recorded a +4℃ relative change.

At longer distances (e.g., 30-35 cm), we observed that the transmitter could increase the receiver's temperature by at most 1℃. We mention this observation as a constraint for the digital modulation method in section VI.

The experiment clearly demonstrates the relationship between the channel distance and the channel capacity (here represented as the output signal's heat propagation delay). When within a certain radius from the transmitter, there is a near linear relationship: every 1 cm of distance from the receiver increases the heat propagation delay by approximately 0.35 minutes.

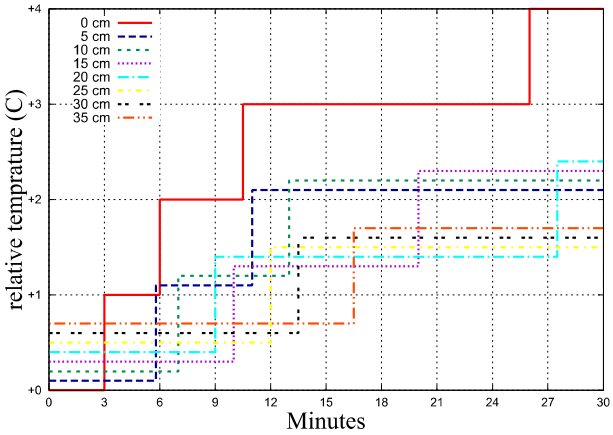


Figure 6: Relationship between distance and rate of temperature increase in the parallel layout.

3) Channel asymmetry

When the channel is used as a half-duplex channel, we noticed that the transmission times for each direction were different. Intuitively, this type of asymmetric relation could be caused simply because of the variety of PC cases or different hardware components. However, more interestingly, when using the same type of case and hardware, the channel was still asymmetric. We found, however, that this was based on case orientation: for each common case orientation, the relative spatial locations of the motherboard's heating sources had created different obstacles between them with respect to the other party's sensors.

Figure 7 illustrates the asymmetry of the half-duplex channel's capacity between a pair of adjacent PCs in the parallel layout. As can be seen, the heat propagation delay in left-to-right transmissions is smaller than a right-to-left. In the first scenario, the left PC is transmitting and the right PC is receiving. In this case, the heat propagation delay was less than 3 minutes. In the second scenario, the right PC is transmitting and the left PC is receiving. In this case, the heat propagation delay is around 7 minutes.

The asymmetrical property of the thermal channel is even more substantial when examining other case orientations like the stacked layout discussed in the following subsection and in Table 3.

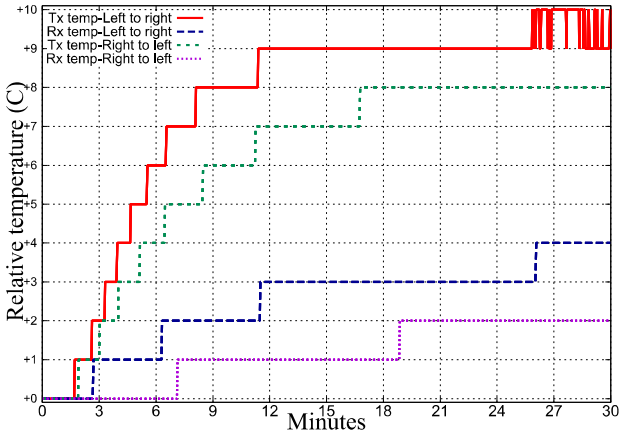


Figure 7: Channel asymmetrical properties.

4) Effects of different orientations

The attacker is usually unaware of the physical layout of the communicating PCs with respect to one another within the context of the attack scenario (section II), and obviously cannot control the layout. We explore the channel's effectiveness across different possible layouts in order to determine the best and worst case scenarios in term of distances. The different layouts are depicted in Appendix A.

Table 3 provides a summary of the results when a transmitter sends a steady signal to the receiver for computers positioned in the stacked layout, comparing those with top computer heating to those with bottom computer heading. Our experiment showed that the computers variously have a heating propagation delay of five minutes (transmitter at top) or 12 minutes (transmitter at bottom). As previously explained, the difference is due to the location of the motherboard in relation to the receiver's case: top computer motherboard is at the lower part of the case, closer to the bottom computer.

Table 3: Properties of the stacked layout.

Property	With top computer heating	With bottom computer heating
Heating computer temperature range (°C)	+9 degrees	+10 degrees
Adjacent computer temperature range (°C)	+3 degrees	+1 degree
Heat propagation delay - first degree increase	5 minutes	12 minutes
Heating time - to max temperature (heater/adjacent)	11/20 minutes	18/12 minutes
Pause propagation delay - first degree decrease	8 minutes	10 minutes
Cooling time - from max to idle temperature.	20+ minutes	10 minutes

When the PCs were positioned in the face-away layout (where the backs of the PCs are facing each other), the heat propagation delay was almost 10 minutes, after which, no further increase was recorded. However, the cooling time with this layout is less than three minutes, which is relatively fast. Notably, no more than a 1℃ increase was observed by receiver sensors during 40 minutes of heating. This suggests that the unobstructed intake of cool air from the sides of both cases has a significant cooling effect. Figure 8 details the sensor reading of a test preformed using this layout.

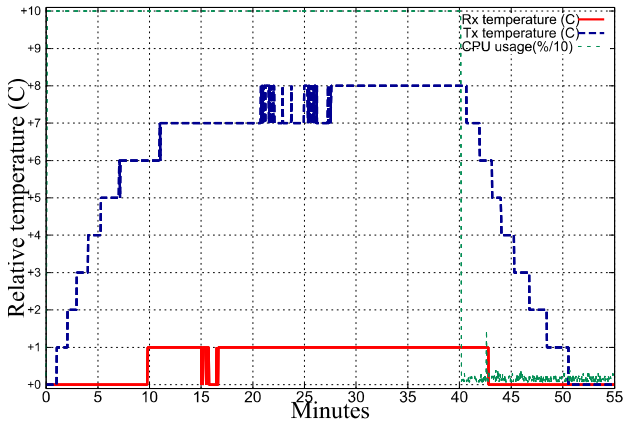


Figure 8: Heat propagation during 40 minutes of heating, face-away layout.

Although uncommon, we investigated the scenario in which the PCs are in the quadrature layout where the posterior side of transmitter points to the side of the receiver (Appendix A, layout B). From a distance of 6 cm between the transmitter and the receiver, the heat propagation delay is 115 seconds, which is relatively fast. An increase of 4℃ took 11 minutes, and the cooling time was approximately 100 seconds. While uncommon, this layout demonstrates the importance of two parameters of computer case design and layout, the location of air intake and the direction of air exhaust.

5) Effects of location relative to walls or furniture

Based on our observations, the distance of the computers from other objects such as walls and furniture has a significant impact on the thermal channel properties. To better understand their effects, we placed a pair of PCs in the parallel layout on top of a table in the middle of a room. In this scenario, the heat propagation delay for 1℃ was about 25 minutes, as shown in Figure 9. This was significantly longer compared with an arrangement in which the computers were even 35cm apart and placed near a wall or table, In which case the heat propagation delay for 1℃ was only about 16 minutes. The main reason for the difference is that in the middle of the room (with no walls or surrounding furniture); the hot air exhaust from the heating computer is unobstructed and spreads to the entire room, hence requiring a larger heat increase before a change is registered in the receiver, causing a delay. Nevertheless, locating computers in the middle of the room with no adjacent walls or surrounding furniture is far less common than being under a table or close to walls.

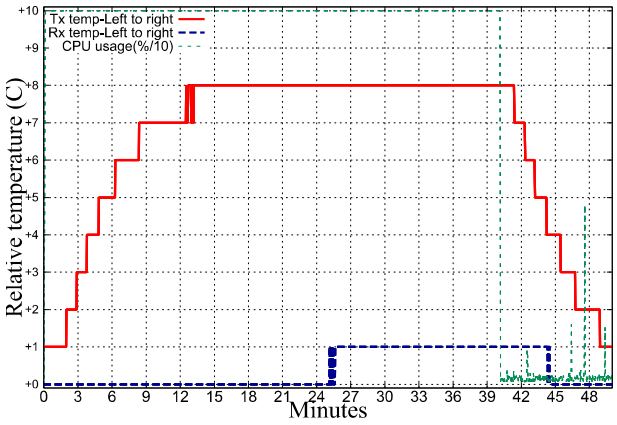


Figure 9: Heat propagation over 40 minutes, parallel layout in the middle of the room.

C. Virtual Machines (VM)

Virtualization technology has become very popular and accessible in modern IT environments. Workstations in which desktop operating systems host virtual machine guests are commonly used. In this case the host and guests are sharing the physical processor. Thus, overloading the processor from inside the guest may have a limited heat generating effect compared to overloading the physical machine. Figure 10 shows that controlling heat emission is also possible from within virtual machine guests. Tests were conducted using two adjacent computers positioned in the parallel layout. The transmitter side runs as a guest in VirtualBox, with four out of eight logical processors allocated to the virtual machine. Aside from the VM, all other parameters were identical to other tests using physical hosts. The tests indicate that operating from within the VM cause a heat propagation delay of about 3 minutes, only slightly higher than non-VM transmissions. In addition, a maximum temperature increase of 3℃ was observed which is less than the 4℃ increase observed in the experiment using physical hosts. This difference can be attributed to the limitation on the workload imposed by the hypervisor on the processors, primarily due to CPU sharing.

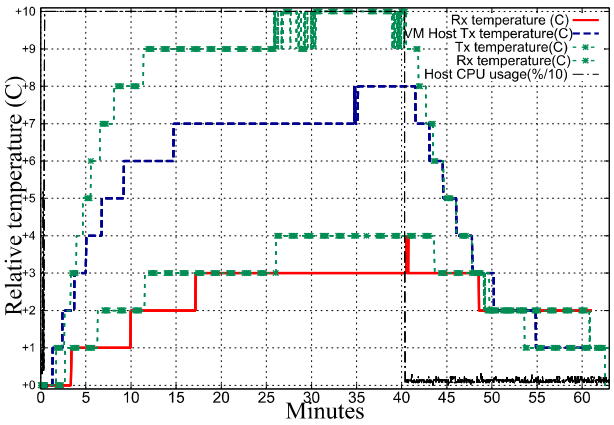


Figure 10: Propagation delay of virtual machine guest versus physical machine host over 40 minutes.

VI. COMMUNICATION PROTOCOL

In this section we show how a covert communication channel can be created between two computers, using the thermal properties observed in the previous section. Most communication systems can be modeled using the OSI communication protocol stack model [26]. In this section we focus on the physical layer (L1) as a basis for further development of the higher layers of the protocol stack (e.g., adding error detection and error corrections). Our discussion includes: 1) basic parameters of L1 2) transmission detection 3) methods for modulating binary data over the signals, and 4) methods for L1 parameter detection and synchronization. While the physical properties of the channel may be asymmetrical, the communication channel is symmetric, in that each side communicates with the other using the same protocol. For the simplicity of discussion, we may refer to the communicating computers as a sender and a receiver even though the channel is bi-directional.

A. L1 Parameters

Given two nearby computers A and B, we denote H as the heat propagation delay (from A to B), and L as B's cooling time. As we will see later in this section, the transmitter and receiver need these parameters in order to successfully transmit the data over the physical channel. As observed in section V, the values of H and L depend on the actual layout and environment of the communicating computers, which is unknown at the outset. The naïve approach is to use predefined H and L during the transmissions (e.g., maximal values). However, for an optimal throughput a better approach would be based on detection of the L1 parameters at runtime. For that end, the communicating sides conduct a handshaking protocol prior to any actual data transfer. This initial process also serves to identify the existence of a receiver in range in order to establish the channel and the detailed handshaking protocol will be discussed at the end of the section. For the presented algorithms, both sides use a rounding function f (e.g. round up calculated times to nearest minute).

B. Differentiating between Internally and Externally Induced Temperature Change

Initially, each participant in the channel continuously monitors its thermal sensors to detect transmissions and handshaking requests. However, sometimes a sudden increase in CPU workload will cause internal heat generation that interferes with its ability to receive channel signals. To help mitigate this interference, we determine the difference between externally induced heat (channel signals from other computers) and internally generated heat (interference). This is achieved by comparing the CPU temperature and CPU utilization changes with respect to other environmental temperature sensors. A high increase in environmental temperature combined with high CPU temperature is assumed to reflect potential interference therefore is not decoded. Increased environmental temperature combined with low CPU temperature can safely be regarded as a potential external signal and decoded.

Figure 11 demonstrates this noise mitigation technique at work. For the first 10 minutes, the receiver observes a temperature increase detected by the sensor, while the receiver's CPU heat is low and its utilization stays below 10% on average. Therefore, he may safely assume that any channel signals received are from an external source. During the following 20 minutes, a few sudden spikes in CPU utilization, caused by casual web browsing and other activities, are recorded. These spikes do not raise the receiver's internal temperature, and therefor do not interfere with possible incoming channel signals. During the final 30 minutes, significant CPU utilization occurs, causing an increase in local temperature. During this interval, it can be assumed that all incoming channel signals are affected by this interference. Note that by employing the channel at idle times (e.g., overnight), an attacker can reduce the potential of interference caused by internal heat generation. The attacker may also program the receiving code to limit the CPU usage of other processes during signal reception. This decreases the chance of interference but degraded user experience may raise suspicion and increase the risk of exposure. Either way, when such internal interference is detected during transmissions, higher layer mechanisms (such as retransmissions) are required to handle the interruptions.

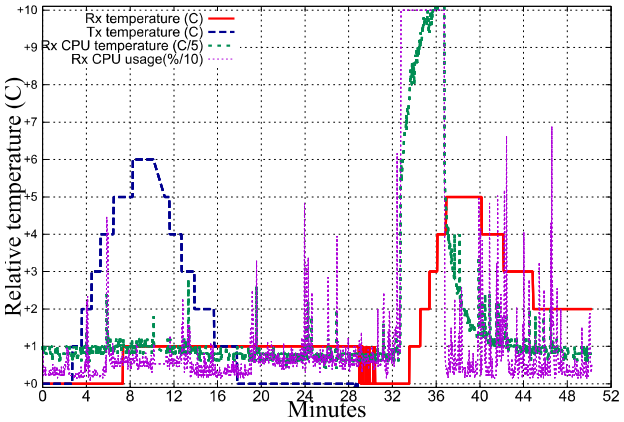


Figure 11: Externally induced temperature increase versus internal heat generation.

C. Digital Baseband Modulations

Based on our observations described in section V, it is clear that broadband modulations are not possible due to the thermal signal's slow rate of change, which is in the scale of minutes. Therefore, digital baseband modulation schemes (line encodings) are appropriate for the proposed channel. Below we summarize three methods of modulating binary data using thermal signals. For modulation purposes we use a conservative temperature manipulation of only + 1℃. This limited range is feasible in all distances, layouts and environments tested. The use of wider temperature ranges in order to increase the data transfer rate is left for future work. For all of the proposed line codes, symbols are transmitted over the medium every T minute, where the baud-rate (symbol rate) is $f = \frac{1}{T}$ symbols per minute.

1) Differential non-return-to-zero.

Differential non-return-to-zero (DNRZ) line code is a variant of NRZ [27] that we have modified to fit the slow moving trends of thermal signals and the unpredictable base temperature of the receiver. Binary symbols are decoded based on the previous base temperature (hence differential) instead of absolute levels. Meaning that, if the signal level has risen (on average) over the current interval (T) compared with the last symbol interval, then a '1' is decoded. Conversely, if the level has decreased or stays the same, then a '0' bit is decoded. Figure 12 details an example transmission and decoding of data using DNRZ with $T = 40$ minutes.

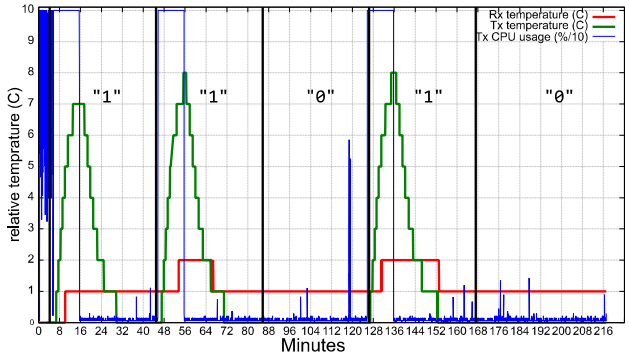


Figure 12: A DNRZ transmission of '11010' using a symbol interval of T 40 minutes, where the PCs have a parallel layout, and distance of 10 cm.

Following these rules, the transmitter decides whether to raise or lower the temperature (i.e. to heat or not heat up the CPU) at the beginning of the symbol interval for a period of T. A summary of the line coding and decoding rules can be found in Table 4 and Table 5. Δt is the time since the beginning of the symbol interval T.

Table 4: Transmitter's line encoding rules

Δt	Bit to transmit	CPU load to incur
any	0	0%
$\leq H$	1	100%
$> H$	1	0%

Table 5: Receiver's line decoding rules

Sensor reading	Symbol value
Current trend: ℃↑	1
Current trend: ℃ →	0
Current trend: ℃ ↓	0

As described in section V, the range of externally induced, relative temperature increase is sometimes very limited. This requires the transmitter to stop heating for the second part of the symbol period, allowing for the receiver to cool down before the next symbol period begins. Consequently, the symbol interval T for DNRZ is $T = f(H) + f(L)$, with a bit-rate of $\frac{1}{f(H)+f(L)}$ bits per minute (bpm).

This line coding, assumes that H and L are known to both the receiver and transmitter prior to communication. Later in this section, we propose synchronization techniques for acquiring these values during the handshake phase.

2) Custom 8b/10b.

DNRZ requires a significant cooling phase during each period to be included in each symbol period. Alternatively, this can also be avoided, to some extent, by using line coding schemes similar to 8b/10b coding, specifically designed to limit the number of consecutive '1" bits rather than evening the bits. This encoding allows the receiver to use the periods in which the transmitter is not generating heat (transmitting a '0') for cooling down. The proposed custom 8b/10b line coding focuses on a separation of '1's and '0's. it is a word mapping that converts words with a length of 8 bits to 10 bits, where the 10 bit words ensure a transmission of at most two consecutive '1"s. The proposed custom 8b/10b line code is presented in Table 6 with the input byte values appearing in decimal and binary, and the encoded value in binary with a 'WXYZ' signifying that the input bit value is maintained for that position.

Table 6: Custom 8b/10b encoding rules

Input value (dec)	Input value (bin)	Output value (bin)
7,23,39 …	WXYZ0111	WXYZ010001
14,30,46 …	WXYZ1110	WXYZ001001
15,31,47 …	WXYZ1111	WXYZ000001
112-123,126,127	0111WXYZ	10001WXYZ0
124	01111100	1000100101
125	01111101	1000100011
224-235,238,239	1110WXYZ	01001WXYZ0
236	11101100	0100100101
237	11101101	0100100011
240-251	1111WXYZ	00001WXYZ0
252	11111100	0000100101
253	11111101	0000100011

This line coding is used with DNRZ in order to improve reliability of the channel and decrease the required symbol interval T by allowing the transmitter to generate heat for the entire duration of T. Consequently, $T = MAX(H, L)$ However, In section V we saw that for most practical cases $L > H$ so $T = MAX(H, L) = L$. Due to the overhead of the added bits, the bit-rate will is slightly reduced. The bit-rate when using this line-coding scheme with DNRZ is $\frac{4}{5f(L)}$ bpm.

The last proposed line codes assume that both communicating computers have synchronized world clock times in a resolution of one minute. As previously described, the temperature changes measured by the sensors occur on a scale of minutes. Therefore, we assume that both computers have sufficiently synchronized clocks, whether they are using different time servers or none at all. Note that in administrated and managed networks this assumption is reasonable.

3) Time Index Signaling.

Time Index Signaling (TIS) is a line coding in which sig-

nals are encoded over different allocated time-slots. Let the message to be transmitted be $M = \{X_1, X_2, \dots, X_n\}$ where $X \in \{0,1,..15\}$ value is indicating a hexadecimal symbol (4 bits). D is a predefined interval of time in minutes. Both the receiver and transmitter wait until a predetermined world-clock time (such as 12:00am), at which point, the transmitter waits $X_1 * D$ minutes and then heats the channel for H minutes followed by a cooling phase of L minutes. This is repeated n times until the entire message M is transmitted. The receiver decodes symbol X_i by dividing the delay $(X_i * D)$ between signals by D minutes. The following pseudo codes details the action of transmitter and receiver. The *Generate-Heat* function starts the heat emission (at the transmitter) for a period of H minutes. The *MonitorTemperatureRise* function (at the receiver), monitors the temperature sensor and wait until an increase of 1℃ is detected.

Algorithm 1: TIS Transmission

Input : T - predetermined time, $data =< x_1, x_2, \dots x_n > | x_i \in \{0 \dots 15\}$, H - heating time, L - cooling time, D - interval

```
if (currentTime() = T) then
    for x in data do
        Sleep(x · D)
        generateHeat(H)
        Sleep(L)
    end
end
```

Algorithm 2: TIS reception

Input : T - predetermined time, $data\_len$ = amount of expected data , H - heat propagation delay, L - cooling time, D - interval, f - rounding function

Output: $data$

```
if (currentTime() = T) then
    for data_len do
        t ← monitorTempratureRise() - T
        data ← data + ⌈(t − H)/D⌉
        T ← t − H + f(H) + L
    end
end
```

We assume that H and L parameters are synchronized and known by both sides, similar to the previous coding methods. In Figure 13 we can see a successful transmission of the B5 hexadecimal value using the TIS method. The parameters in this case are D=2, H=6, f(H)=10, and L=30. The first digit is 0xB which is 11 decimal, so the sender waits 22 minutes (D*11=2*11=22) for the right timeslot. At the next step, the transmitter generates heat for 10 minutes (H=10) and then remain idle for 30 minutes (L). To signal the 0x5 digit, it sleeps for an additional 10 minutes (D*2=5*2=10) and generates a heat pulse for 10 minutes (H). The receiver senses a temperature increase for B and 5 after approximately 28 and 77 minutes respectively. Decoding the data is performed by subtracting the heating time H from the time of the sensed temperature increase to retrieve the time slot delay and divide it by the slot size D. in our example

$$\left\lceil \frac{(28-0)-6}{2} \right\rceil = B \text{ and } \left\lceil \frac{(77-62)-6}{2} \right\rceil = 5.$$

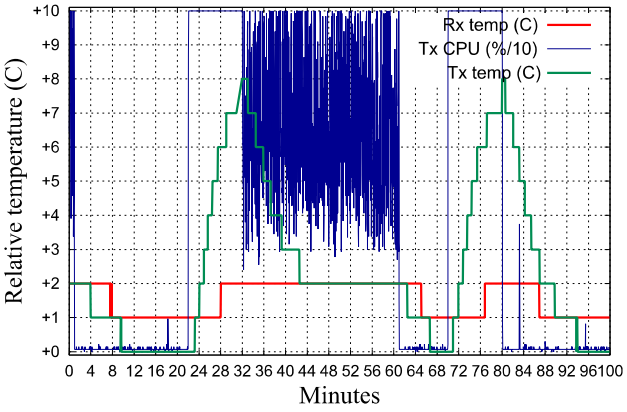


Figure 13: A TIS transmission of hex values B5 using D=2min , H=6min, f(H)=10min, L = 30min, with a parallel layout and distance of 10cm.

It is clear that the TIS scheme's bit-rate is variable based on the value of the message M. In the worst case where $M = \{F, F, F, \dots\}$, the bit-rate of TIS is $\frac{4}{(f(H)+L+15D)}$ bpm.

During the experiments, we observed that $D = 2$ minutes is a good interval for most cases. However, a smaller D may be used in environments with a steady temperature. Moreover, a parity bit can be added to the transmission, allowing for useful error detection. Because in most cases, the received time index is not likely to drift by more than D minutes, resulting in a change in parity. One of the advantage of this encoding scheme is that it minimizes the number of heat emissions required to only one emission for every 4 bits. This characteristic is especially important to the thermal based channel since each heating and cooling phase consumes resources and is prone to environmental interference.

4) Comparison of digital baseband modulations

The bit-rates of each encoding scheme depend on H and L, which vary between setups. To summarize, we present an example of bit-rates based on parameters from our experiments in Table 7. The H and L values are based on observations from a parallel layout setup with a 10 cm distance between the PCs. Although the parameters are not optimal, in order to more clearly demonstrate possible results.

Table 7: Data rate formulas and examples

Modulation	General data rate (bits per minute)	Example: Parallel 10 cm H=6,L=20,D=2,data='B5'
DNRZ	$\frac{1}{f(H)+f(L)}$	208 minutes (2.3 bits per hour)
DNRZ +8b/10b	$\frac{4}{5f(L)}$	200 minutes (2.4 bits per hour)
TIS	$\frac{4}{(f(H)+L+x_iD)}$	64 minutes (7.5 bits per hour)

D. Handshaking Protocol

The main objective of the handshake protocol is to convey

and agree upon the parameters used for the communication protocol, the heat propagation delay (H), and cooling time (L). Following the handshake phase, each side will know its H and L. Due to the static nature of computer's locations and layouts, the handshaking can during the initial communication setup, then be carried out occasionally (e.g., once a week). In the handshaking algorithm, we arbitrarily refer to the handshaking initiator as A and the other side (recipient) as B.

The following algorithm consists of exchanging messages of decreasing length between A and B in order to agree on the propagation heat delay and symbol interval. At the outset, both A and B are in a listening state, waiting for handshaking request. At an arbitrary time, side A generates heat for a specified initial length of time T (A->B) and then stops and waits for a response (line 5). B receives the signal (line 17), waits for its temperature to decrease back to normal, and responds with a heating period of a fixed length (B->A). When A receives this signal (line 10), it decreases the initial length T by a fixed amount and sends another signal. At some point, the heating pulse will not reach B due to the short length of the transmission, at this point the last duration used is the heat propagation delay required for successful communication. This process is later repeated for the opposite direction with side B initiating the handshake process.

Algorithm 3: Heat delay handshake

```
   Input  : T - initial time length, δ - step size, W - pause time
   Output: H
 1 if side A then
 2     while T > 0 do
 3         generateHeat(T)
 4         t ← currentTime()
 5         res = ← MonitorTemperatureRise(T)
 6         if res = TIMEOUT then
 7             return T, L
 8         end
 9         else
10             L← res - t - W
11             T ← T − δ
12         end
13     end
14 end
15 else
16     counter ← 0
17     res ← MonitorTemperatureRise(T)
18     if res = TIMEOUT then
19         if counter > 0 then
20             return T − δ·counter
21         end
22         else
23             return FAIL
24         end
25     end
26     else
27         counter ← counter+1
28         MonitorTemperatureRise(T)
29         MonitorTemperatureFall()
30         Sleep(W)
31         generateHeat(T)
32         GOTO 17
33     end
34 end
```

The time it takes side A to receive the response signal (response time) is sum of the cooling time of B and the heat propagation delay for A ($H + L$). once the process is completed for both directions and the heat propagation delays are known, each side may subtract its own delay from the response time in order to get an estimate of the other side's cooling time ($(H + L) - H =$ L).

VII. Countermeasures

Defensive countermeasures for general TEMPEST threats [13] can serve as a source of inspiration for the implementation of countermeasures for the thermal channel. The prevailing standards are aimed at limiting the level of detectable information-bearing RF signals over a given open-air distance, or its equivalent, when insulating materials are used. Practically, certified equipment is classified by 'zones' that refer to the perimeter that needs to be controlled to prevent signal reception [28]. As a countermeasure against attacks like BitWhisper, the "zones" approach should be used to define the physical distances required by potential heat emitting and heat sensing components connected to different networks. In some cases, mainly due to the space limitations, keeping minimal distances between computers is not practical and obviously, managing physical distances between different networks is complicated in terms of space and administration overheads that increases with each air-gap network used. One solution may be to place thermal sensors between computers that contain sensitive information and all other computers in order to detect anomalous thermal emissions or to use existing administrative monitoring solutions [29] [30] in order to detect thermal communication attempts. These monitoring systems, typically used for servers, already monitor CPU usage and thermal sensors for maintaining availability. The logs and alerts from these monitoring systems can be used to detect communication attempts if they are made available to an aware analyst or if appropriate rules are put in place.

VIII. Conclusion and Future Work

This work introduces a new type of covert channel codenamed BitWhisper that enables communication between two air-gapped computers. Our covert channel exploits the thermal radiation emitted by one computer, operating within permissible heat boundaries, to deliver information to a neighboring computer, equipped with standard heat sensors. Our method does not require dedicated or modified hardware, and is based solely on software. We describe a complex yet feasible attack model, based on gradual contamination of computers from isolated networks and public networks, located within one spatial zone, advancing through 'thermal pings' until eventually two spatially adjacent computers belonging to separate networks are found. BitWhisper allows those two air-gapped computers to communicate with one another. We implement a working prototype of BitWhisper to evaluate the capabilities and boundaries of the physical layer. A variety of settings, configurations and parameters, with varying distances between computers and several types of target computers were evaluated. We also discuss modulation methods along with the initial handshaking protocol, aimed at delivering narrow-band communication between the two computers, at close distances. BitWhisper provides a feasible covert channel, suitable for delivering command and control

(C&C) messages, and leaking short chunks of sensitive data such as passwords.

We expect this pioneering work to serve as the foundation of subsequent research, which will focus on various aspects of the thermal channel and improve its capabilities. In particular, we consider the following open issues: 1) exploring the feasibility of full-duplex communication, i.e., allowing both sides to send and receive at the same time, and 2) examining the feasibility and efficiency of broader bandwidth, utilizing multiple temperature sensors for the MIMO model. Another research direction concerns collaboration between multiple heat-emitting computers, located within the same room, orchestrating their operations to increase the overall effective transmission range.

IX. REFERENCES

[1] R. Shirey, "RFC4949 - Internet Security Glossary, Version 2," IETF, 2007.

[2] NIST, "Special Publication 800-53R4 Security and Privacy Controls for Federal Information Systems," 4 April 2013. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf. [Accessed 15 September 2013].

[3] DISA, "Top Secret/Sensitive Compartmented Information service," [Online]. Available: http://www.disa.mil/Services/Network-Services/Data/TS-SCI-IP. [Accessed 15 September 2013].

[4] S. Karnouskos, "Stuxnet worm impact on industrial cyber-physical system security," in *IECON 2011-37th Annual Conference on IEEE Industrial Electronics Society*, 2011.

[5] B. Knowlton, "Military Computer Attack Confirmed," 2010. [Online]. Available: http://www.nytimes.com/2010/08/26/technology/26cyber.html?_r=2&adxnnl=1&ref=technology&adxnnlx=1423562532-hJL+Kot1FP3OEURLF9hjDw.

[6] ICS-CERT, "malware infections in the conrol environment," 2012.

[7] Z. Z. X. a. H. W. Wu, "Whispers in the Hyper-space: High-speed Covert Channel Attacks in the Cloud," in *USENIX Security symposium*, 2012.

[8] G. V. Jie Chen, "CC-Hunter: Uncovering Covert Timing Channels on Shared Processor Hardware," in *MICRO-47 Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture*, 2014.

[9] H. W. a. H. W. Ki Suh Lee, "PHY Covert Channels: Can you see the Idles?," in *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI '14)* , Seattle, 2014.

[10] A. Madhavapeddy, R. Sharp, D. Scott and A. Tse, "Audio networking: the forgotten wireless technology," *Pervasive Computing, IEEE,* vol. 4, no. 3, 2008.

[11] M. a. G. M. Hanspach, "On Covert Acoustical Mesh Networks in Air," *arXiv preprint arXiv:1406.1213,* 2014.

[12] J. Loughry and A. D. Umphress, "Information leakage from optical emanations," *ACM Transactions on Information and System Security (TISSEC),* vol. 5, no. 3, pp. 262-289, 2002.

[13] M. G. Kuhn and R. J. Anderson, "Soft Tempest: Hidden data transmission using electromagnetic emanations," in *Information hiding*, Springer-Verlag, 1998, pp. 124-142.

[14] M. Guri, G. Kedma, A. Kachlon and Y. Elovici, "AirHopper: Bridging the Air-Gap between Isolated Networks and Mobile Phones using Radio Frequencies," in *9th IEEE International Conference on Malicious and Unwanted Software (MALCON 2014)*, Puero Rico, Fajardo, 2014.

[15] R. Callan, A. Zajic and M. Prvulovic, "A Practical Methodology for Measuring the Side-Channel Signal Available to the Attacker for Instruction-Level Events," in *Microarchitecture (MICRO), 2014 47th Annual IEEE/ACM International Symposium on*, IEEE, 2014, pp. 242-254.

[16] S. J. Murdoch, "Hot or not: Revealing hidden services by their clock skew," in *ACM conference on Computer and communications security*, 2006.

[17] B. Snyder, "Snowden: The NSA planted backdoors in Cisco products," 2014. [Online]. Available: http://www.infoworld.com/article/2608141/internet-privacy/snowden--the-nsa-planted-backdoors-in-cisco-products.html.

[18] S. Stasiukonis, "social-engineering-the-usb-way," 2006. [Online]. Available: http://www.darkreading.com/attacks-breaches/social-engineering-the-usb-way/d/d-id/1128081?.

[19] A. von Meier, Electric Power Systems: A Conceptual Introduction, Wiley, 2006.

[20] H. N. Weste and M. D. Harris, Cmos Vlsi Design: A Circuits And Systems Perspective, Pearson Education India, 2006.

[21] Intel, *CPU Monitoring With DTS/PECI,* M. Berktold, Ed., 2010.

[22] Accredited Standards Committee, "Working Draft AT Attachment 8 - ATA/ATAPI Command Set," 2006.

[23] ozone3d, "FurMark: VGA Stress Test, Graphics Card and GPU Stability Test, Burn-in Test, OpenGL Benchmark and GPU Temperature," 2013. [Online]. Available: http://www.ozone3d.net/benchmarks/fur/. [Accessed January 2015].

[24] GIMPS, "Great Internet Mersenne Prime Search," April 2014. [Online]. Available: http://www.mersenne.org/download/. [Accessed 2015].

[25] REALIX, "HWiNFO, HWiNFO32 & HWiNFO64 - Hardware Information, Analysis and Monitoring Tools," 2015. [Online]. Available: http://www.hwinfo.com/. [Accessed 2015].

[26] Microsoft, *The OSI Model's Seven Layers Defined and Functions Explained,* 2014.

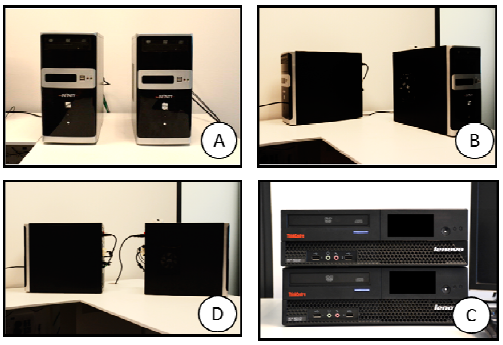
[27] Institute for Telecommunications Sciences, *1037c. telecommunications: Glossary of telecommunication terms,* 1996, p. N10.

[28] R. J. Anderson, "Emission security," in *Security Engineering, 2nd Edition*, Wiley Publishing, Inc., 2008, pp. 523-546.

[29] Nagios Enterprises, "Nagios overview," [Online]. Available: http://www.nagios.org/about/overview/. [Accessed 2015].

[30] Microsoft, "System center Operations Manager," 2013. [Online]. Available: https://technet.microsoft.com/library/hh205987.aspx. [Accessed 2015].

APPENDIX A



Layouts: (A) parallel layout; (B) quadrature layout; (C) stacked layout; (D) face-away (back-to-back) layout.