Acoustic Surveillance of Physically Unmodified PCs

Michael LeMay
Department of Computer Science
University of Wisconsin-Eau Claire
Eau Claire, WI, U.S.A.

Jack Tan
Department of Computer Science
University of Wisconsin-Eau Claire
Eau Claire, WI, U.S.A.

Abstract— *Computer equipment produces a wide range of emanations in the visible, electromagnetic, and acoustic spectra. It is well known that electromagnetic emanations can reveal information about the operations being performed by a machine, and it has recently been shown that acoustic emanations can do the same. Additionally, techniques have been developed to manipulate electromagnetic emanations so that they surreptitiously transmit data to a remote receiver. In this paper, we similarly demonstrate how acoustic emanations can be manipulated to transmit arbitrary data and show how this technique can be used to create a practical, software-only acoustic keylogger. Finally, we recommend countermeasures to prevent such an attack from being performed.*

Keywords: TEMPEST, emanations, keylogger, acoustic, side channel

1. Introduction

Computer equipment emits a variety of optical, electromagnetic, and acoustic emanations. Much work has been done to determine the extent of the information that can be remotely ascertained by monitoring optical and electromagnetic emanations, and a variety of attacks with varying levels of practicality have been devised. As a result of this research, many modern systems prevent the leakage of these compromising emanations.

In contrast, much less attention has been devoted to acoustic side channels, due largely to the assumption that the clocks governing the operation of modern machines operate at far too high a frequency to reveal information through low-frequency acoustic channels. Recent studies have challenged this assumption, and in this paper we demonstrate an actual application in which the acoustic emanations of a physically unmodified personal computer can be manipulated to surreptitiously transmit arbitrary data in a repeatable fashion.

2. Related Work

Our work was inspired by the experiments performed by Shamir and Tromer [1], where they showed that the motherboard power smoothing capacitors supplying a Celeron 666MHz processor emit acoustic emanations when the processor is active. They also observed that these emanations could be easily collected and analyzed using an inexpensive microphone and an open-source audio analysis suite, Baudline [4]. Furthermore, these emanations exhibit interesting characteristics that can be correlated to the type of operation being performed by the processor at the time. In particular, they were interested in the potential for discovering information about cryptographic material that is leaked through these emanations. They include several demonstrative spectrograms in their analysis, including some that clearly demarcate between two phases of a common public-key operation based on the Chinese Remainder Theorem [6]. Their analysis also includes several experiments which effectively demonstrate that the signals being analyzed are actually acoustic and not electromagnetic. The most important aspect that is lacking from their analysis is any practical application of the emanations they analyzed.

Kuhn and Anderson [7] showed that electromagnetic side channels can be manipulated to surreptitiously transmit data. They demonstrated that a variety of prominent tones could be transmitted to an ordinary AM shortwave radio by displaying specially constructed patterns on a standard CRT. A noticeable shortcoming of this technique is that it caused strange patterns to be displayed on the CRT, which would of course alert an informed user that something is not right. Thus, it was best suited for use after hours. However, Kuhn and Anderson then presented more advanced techniques to manipulate the dithering on screens such that the human eye is made to see one image, while another, attacker-defined image is clearly transmitted to a standard video TEMPEST receiver. These techniques could potentially be used even while a human user is present at the terminal. Finally, Kuhn and Anderson explored how screen fonts can be subtly modified to eliminate TEMPEST emanations while minimally changing the font's on-screen appearance. In our work, we show how an attacker can surreptitiously transmit data using acoustic channels, and then provide best practices to minimize the effectiveness of these channels.

Asonov and Agrawal [2] of IBM Research developed an effective attack system that analyzes the sounds emitted by various keys on a single keypad to capture PINs and passwords from computers, ATMs, and other keypad-equipped devices. Their attack is entirely non-invasive, as sounds are collected from a long distance using a parabolic microphone. The sounds are analyzed using a neural network that can be trained to distinguish different key sounds, even if those sounds are indistinguishable to the human ear.

Through experimentation, they were able to conclude that the variations in key sounds were caused by the keys striking different parts of the plate underneath the keyboard. Basically, the plate was acting as a drum that resonated differently depending on where it was struck. In the final analysis, this work was useful in its own right because it contributes a practical attack technique that is still as applicable now as it was at the time of its publication, and it also served as a motivation for further research into more subtle acoustic emanations generated by computers.

The government has a long history of research on attacks against various electronic emanations, but most of it is classified. In fact, there exist a few known TEMPEST documents, NACSIM 5103, 5104, and 5105 relating to acoustic emanations, but they remain classified, according to the partially declassified NACSIM 5000 [9].

3. Experimental Setup

1. Personal Computer

For all of these experiments, we used a computer that was constructed from modern components in December 2004. It was powered by an Intel Celeron D 320 (2.40GHz, Prescott core, no hyperthreading) hosted by a Jetway PM9MS microATX motherboard and cooled by a prepackaged water cooling system. We chose a high frequency processor to demonstrate that the emanations observed by Shamir and Tromer are still present at higher operating frequencies.

We performed a standard installation of Gentoo Linux on our test machine, and executed all of our experiments on it. None of the underlying operating system was modified so as to provide a realistic, multitasking environment. We also ensured that no intensive tasks were running during our experiments. In the future, it may be interesting to determine what effect an intensive background process such as any of the distributed computation projects would have on our results. However, most systems sit idle when not being actively used, so our results should accurately predict those that would be obtained from a typical system.

2. Microphone #1

To achieve a flat frequency response from our recording equipment, we selected an omnidirectional measurement microphone produced by Behringer, the ECM8000, which is readily available for about $50. This is an electret condenser microphone that has a linear frequency response curve from about 15Hz to 20kHz. It has a sensitivity rating of -60dB.

The microphone was coupled with a Behringer UB802 Eurorack 8-input mixer that is also readily available for $50. To complete our ensemble, we obtained cables both to connect the microphone to the mixer and the mixer to the computer, which brought our total investment to just under $150. This is well within the reach of any potential attacker.

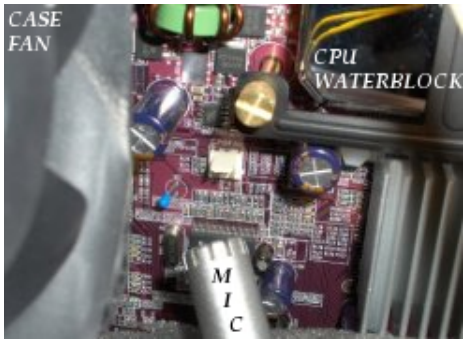

Fig. 1. Microphone positioning for first set of experiments

3. Microphone #2

We also conducted some experiments using a wireless spy camera with an integrated microphone. Again, we wanted to constrain ourselves to a budget accessible to even the most limited attacker, so we purchased X10 wireless spy cameras. For only $180, we purchased a "special package" comprising two wireless spy cameras, an A/V wireless receiver, a USB adapter for converting the video input, two Pan'n'Tilt robotic camera bases, remote controls for the same, plus a large number of other accessories. For another $20, we purchased a AA battery pack that can be used to power one of the cameras without plugging it into an AC outlet or robotic base. What makes this equipment interesting is the fact that the entire camera can be concealed inside of a standard ATX mid-tower case, and transmit audio and video to a remote receiver. Of course, we have no use for the video feed in this application, which indicates that a more compact microphone-only unit could be concealed with even greater ease.

4. Recording Station

To record the audio produced by our microphones, we enlisted a standard Pentium 4-M laptop. It contains an integrated 16-bit audio controller with a single microphone port. Thus, it is hardly a professional recording device, but would be in any reasonable attacker's toolkit. We ran Linux on the laptop during the experiments, and used the same recording package as Shamir and Tromer, Baudline.

For more advanced spectral analysis, we used the FAWAVE [5] software package, created and maintained by James Walker of the University of Wisconsin-Eau Claire. FAWAVE runs only on Windows, so we used a standard Windows XP desktop for that portion of our analysis.

4. Initial Experiments

One of the objectives of this project was to replicate many of the tests performed by Shamir and Tromer on a different computer, to lend additional support to the idea that acoustic emanations from computer internals are common phenomena, not simply a strange characteristic of their particular machine.

For these initial tests, we used our studio microphone to record the audio emanating from inside the computer's case during the execution of the test program. To capture the emanations more effectively, we removed the side panel from the case and positioned the microphone as shown in Figure 1. The cylindrical objects surrounding the tip of the microphone are power supply capacitors. Our observations support the hypothesis that these are the sources of the dominant acoustic emanations.

The first experiment we performed was inspired by Shamir and Tromer's analysis of GnuPG operations, which led to the appearance of the CRT's distinct phases in their acoustic spectrogram. For our experiment we decided to roughly emulate the operation being run within GnuPG using parameters that we chose, to allow us to distinguish the effects of various parameter selections on the acoustic emanations.

We used GMP (GNU Multiple Precision Arithmetic Library) [11] to perform the simple computation and assignment $c = x^y (mod\, m)$. Our hypothesis, suggested by Shamir and Tromer's experimentation, is that fixing the values of x and y but providing different values for m should noticeably change the acoustic emanations of the system, since m influences the size of the intermediate values used during the computation. We fixed x to be 499,999 bits of alternating zeros and ones, and y to be 125,001 ones followed by 374,999 zeros. We selected these numbers simply because they are large, so that they would produce easily discernable acoustic emanations. We ran our experiment with three values for m, and obtained the results shown in Figure 2.

There is one final, crucial preparation for our experiment that we must explain. Our Celeron D can adjust its effective frequency in 300MHz steps, all the way from 300MHz to its rated 2.40GHz. It doesn't actually scale its clock, but instead shuts down for minuscule periods of time. The user is able to control what fraction of time the processor operates, and thus adjust its effective frequency. In this experiment, we used the Celeron's power management functions to scale its effective frequency down to 300MHz from 2.40GHz. As we will discuss more thoroughly later in this paper, acoustic emanations are most apparent at lower effective frequencies (duty cycles).

Figure 2 is a condensed screenshot taken of Baudline after recording the emanations produced during the experiment. We inserted long pauses between computations, which were removed from the figure to conserve space. This spectrogram represents frequency along the horizontal axis, advancing time from top to bottom, and increasing amplitude as increasing darkness. The computations are clearly visible as dark horizontal bands of high-amplitude audio. They are clearly different from each other, both in terms of the duration of the computations and the frequency characteristics of each computation. As you may have already ascertained, the modulus of the first exponentiation was significantly larger than that of the

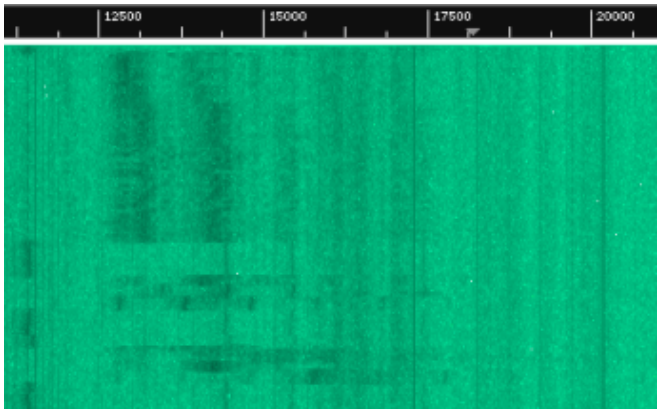

Fig. 2. Condensed spectrogram from first experiment with GMP

latter computations. Specifically, the first computation, which appears at the top of the spectrogram, used an arbitarily-selected 144-bit exponent, chosen solely for its size. The latter two computations used 12-bit exponents that are represented in binary as `100101001010` and `100000000011`, respectively. In a cryptographic application, we would have to consider the primality of many of these parameters, but this is simply a demonstration in which primality does not have any special effect on the outcome.

With this experiment, we have shown that acoustic emanations are emitted by modern personal computers, and that they are not just a strange feature of Shamir and Tromer's hand-picked system. In fact, the authors have detected acoustic emanations from other systems at their workplace with their unaided ears when the side panels are removed from the systems, suggesting that a large number of machines have the potential to emit these sounds. Additionally, we have shown that acoustic emanations can be used to determine the duration of intense computations, and perhaps characterize the parameters controlling those computations if something is known about the algorithm being executed.

5. Instruction Sequence Tests

Our second test comprised a sequence of instruction loops, encoded using inline assembly language in a GCC C source file. A sample loop copied from our test is reproduced here:

```
// cmpxchg8b
u_int64_t ref = 0x1234567812345678ull;
asm("top_cmpxchg8b:\n"
    "cmpxchg8b %0\n"
    "loop top_cmpxchg8b\n"
    : "=m"(ref)
    : "d"(0x12345678u),
      "a"(0x12345678u),
      "c"(repCnt),
      "b"(0x78787878u)
    );
```

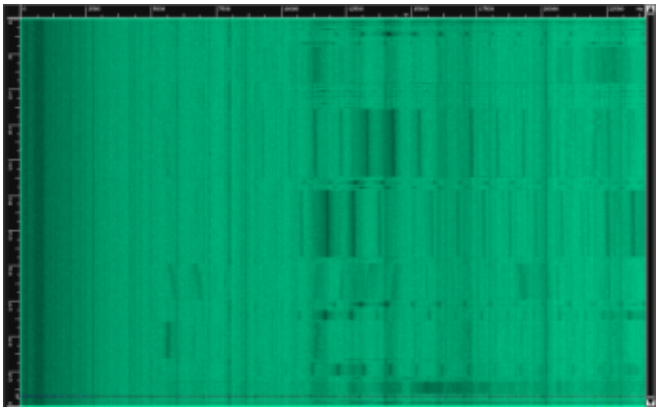

Fig. 3. Acoustic emanations from instruction sequences at 300MHz

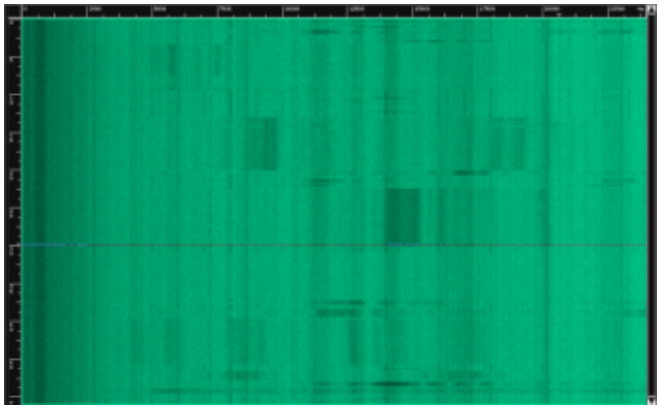

Fig. 4. Acoustic emanations from instruction sequences at 600MHz

```
HPT_MARK(cmpxchg8b);
SLEEP;
```

This loop is exercising the `cmpxchg8b` instruction. This instruction compares `EDX:EAX` with its memory operand. If equal, it sets `ZF` (the zero flag) and replaces the operand with the contents of `ECX:EBX`. Otherwise, it clears `ZF` and loads the operand into `EDX:EAX`. In short, this is a complicated, obscure instruction that is likely to induce some interesting emanations if anything will. In this particular loop, the instruction is not performing any useful work. It is simply changing the value held in the 64-bit variable `ref`. The variable `repCnt` determines how many times the loop will be executed. It must be set relatively high if we are to distinguish the acoustic emanations from different loops.

At the end of the loop, we record the time at which the loop completed. `HPT_MARK` stands for High-Performance Timer Mark, and is a simple inline assembly macro we constructed to read and record the value of the Pentium's timestamp counter using the `rdtsc` instruction. This instruction returns the number of clock cycles that have elapsed since the computer was started. Our macro records this value in a pre-allocated memory location and labels it with the name of the instruction being measured. The final line of code, `SLEEP`, is another macro that simply releases the processor for a short time using the `usleep` UNIX system call.

We sequentially tested a large number of both prefixed and unprefixed Intel instructions. Unfortunately, we don't have the space to explain the purpose of all these instructions. We selected them to provide a fairly comprehensive representation of the Intel architecture's diverse levels of instruction complexity and many functional units. Specifically, we execised the FPU and the MMX, SSE, and SSE2 subsystems, as well as the processor's cache lines. Again, we executed this test with the processor set to a duty cycle of 12.5%, or 300MHz, and the resulting spectrogram is shown in Figure 3.

The various instruction loops can be easily distinguished in the spectrogram. It is very clear that many of the instructions take much, much longer to execute than others, since all of the instruction loops had identical iteration counts. It is also clear that different instructions induce different audio frequencies. We exploit these differences later in this paper to allow surreptitious data transmission.

6. Effects of Power Management

Before we present a potential application for the predictability of these acoustic emanations, we must discuss the factors that affect them and their probable root cause.

First, we repeated our second experiment at a higher effective frequency, 600MHz. The resulting spectrogram is displayed in Figure 4. There is a drastic difference between the 300MHz and 600MHz spectrograms. The basic outlines of the instruction sequences are still easily discernible, but most frequency bands have "smudged" and shifted, and several have disappeared entirely. Thus, acoustic emanations collected at the lower effective frequency are much more revealing and potentially useful.

We repeated this experiment at all frequency levels supported by our processor. Acoustic emanations are discernible at all frequencies except 2.40GHz, when no power management is being performed. The emanations uniformly grow more faint (lower amplitude) as the effective frequency is increased. We are unable to conclude that no emanations are being produced at 2.40GHz, but we are also unable to collect any evidence for their existence with our modest equipment.

Due to our observations, we are able to hypothesize about the root cause of these acoustic emanations. As Shamir and Tromer noted, the power supply capacitors on a motherboard are the likely sources of acoustic emanations. Since we now know that power management options have an extremely dramatic effect on acoustic emanations, we hypothesize that the plates of the power supply smoothing capacitors on the motherboard are serving as acoustic transducers.

Capacitors are passive electrical devices that store small

Fig. 5. Four frequency modulation levels

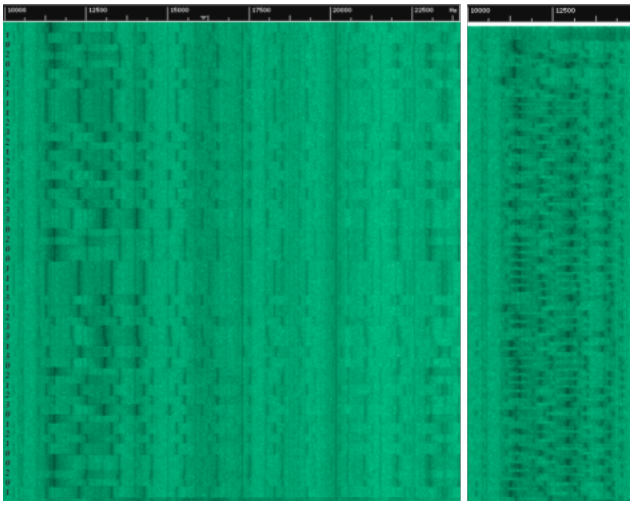

Fig. 6. NRZ-encoded quaternary version of ASCII "Hello World!" on the left, versus a modified encoding of the same string on the right that eliminates the clock recovery problem. The spectrogram on the right has been vertically scaled, and actually takes approximately twice as much time as the spectrogram on the left.

amounts of electricity. The power smoothing capacitors on motherboards are connected in parallel to the power input, and store enough energy to provide the small surges of energy required by the processor when computations intensify. These particular capacitors are known as electrolytic capacitors. They consist of two thin sheets of conductive foil, each connected to one wire that protrudes from the bottom of the capacitor. The two foil plates are separated by a dielectric paste. These plates are tightly wound and are unable to move very far. However, it is possible that increasing the voltage difference across the capacitor could cause a slight physical motion in the plates in response to the stronger electrical attraction. Likewise, a lessening of the potential could allow the plates to return to their original position. In fact, this movement is much more of a possibility in electrolytic capacitors than in other types because electrolytic dielectrics are semi-liquid.

Thus, we hypothesize that the rapid variations in power load caused by various instruction sequences could induce capacitor plate vibrations at audio frequencies. When the processor's duty cycle is decreased, these power fluctuations are accentuated, since the processor is switched off for a short period and then suddenly switched back on, causing tiny power surges in the power supply capacitors.

7. Acoustic Data Transmission

We have shown that acoustic side channels may hold potential for revealing unintended information about the operation of a computer. However, the acoustic emanations of a computer can also be manipulated to surreptitiously transmit arbitrary data.

It is possible to transmit data acoustically by modulating the frequency of acoustic emanations. Our instruction sequence experiments at 300MHz show that it is possible to reliably distinguish between the frequency spectra of at least four distinct instruction sequences, those of `bswap`, `cmpxchg8b`, `bound` and `bt`. Their respective spectra are displayed in Figure 5.

Almost certainly, more levels than these could be distinguished, especially with machine assistance such as that provided by neural networks, but even these are sufficient for our experiments. Using these four levels, it is possible to transmit quaternary data, rather than binary data. This allows us to double our transmission rate over what it would have been if we had only two levels at our disposal.

Initially, we wrote a short program to transmit the ASCII sequence corresponding to "Hello World!" Below, we show both the binary and quaternary ASCII codings for the first word, as an example:

```
=====BASE2===BASE4
H: 0100 1000: 1020
e: 0110 0101: 1211
l: 0110 1110: 1232
l: 0110 1110: 1232
o: 0110 1111: 1233
```

Referring back to Figure 7, we assign each of the instructions' acoustic outputs to a single quaternary digit: `bswap = 0`, `cmpxchg8b = 1`, `bound = 2`, and `bt = 3`. Each of these instructions has a different complexity and thus they all take different amounts of time to execute. To compensate for this, we vary the number of loop iterations for each instruction, to produce approximately equal execution durations. The quaternary acoustic encoding of our simple message is shown in Figure 6.

This simple encoding is similar to the binary encoding known as Non-Return to Zero (NRZ) in computer networking. Thus, it shares NRZ's biggest weakness, the clock recovery problem. It is possible (and common) to have long runs of the same level, which makes it difficult to determine precisely where one digit ends and another begins. This problem is apparent in Figure 6.

In networking, a number of alternative codings were suggested to eliminate this problem. They all introduce

some level of inefficiency, and thus incur a slight bandwidth penalty. For our system, we also propose an alternative coding that solves the clock recovery problem. It is likely not be the most efficient encoding possible, but it only requires twice as many quaternary transmissions as the NRZ scheme. Unfortunately, we don't have space in this paper to include the full definition of the encoding function. Its most important characteristic is that no quaternary character is ever transmitted more than once sequentially, completely eliminating the clock recovery problem. For example, if the character `H` were originally to be transmitted as `1020`, the expanded sequence `0123 0202` would be transmitted instead. There are no adjacent, identical digits, nor would there be if more characters were transmitted. Again, we transmitted "Hello World!" using this encoding, as is shown in the spectrogram at the right side of Figure 6.

8. Acoustic Keylogging

Having defined an effective scheme for transmitting data using the acoustic emanations from our system, we explored an actual application that may be used by an attacker, an acoustic keylogger. One of the problems attackers face with traditional keyloggers is retrieving the keystrokes after they have been collected and stored, even if they are able to install the keylogger in the first place. By harnessing acoustic transmissions and long range acoustic detection, they may be able to attack more effectively.

We modified a free keylogger for Linux, aptly named the Linux KeyLogger [8], to transmit keystrokes using the enhanced encoding scheme developed in the previous section. Each character takes about two seconds to transmit using our current scheme, which is far too slow to transmit typing in realtime. However, the attacker could configure the keylogger to capture and transmit only critical information, such as system usernames and passwords. This would also be beneficial from the attacker's viewpoint because it would reduce the probability that a legitimate user on the system would notice the keylogger.

The acoustic keylogger is potentially more conspicuous than a traditional keylogger since it consumes the computer's processor resources during data transmission, and because it causes the machine's internals to emit faint, structured sounds. Despite these impediments, acoustic keyloggers may still be advantageous to an attacker since it is not necessary to retrieve the keylogger or its data files at any point in the future; it must only be installed and then monitored until the desired data has been intercepted. It even allows eavesdropping on machines that are physically inaccessible and disconnected from the network, as long as the keylogger has been installed at some point in the machine's past.

Most attackers would probably be content with manually interpreting spectrograms and locating the data pulses produced by our methods. However, highly motivated attackers may attempt to minimize the keylogger's "footprint", which would involve minimizing both the keylogger's use of aggressive power management and the duration of data transmissions. Such transmissions would require more advanced techniques for reliable detection. Thus, serious attackers may use neural networks to distinguish levels, just as they were used to distinguish keystrokes in [2].

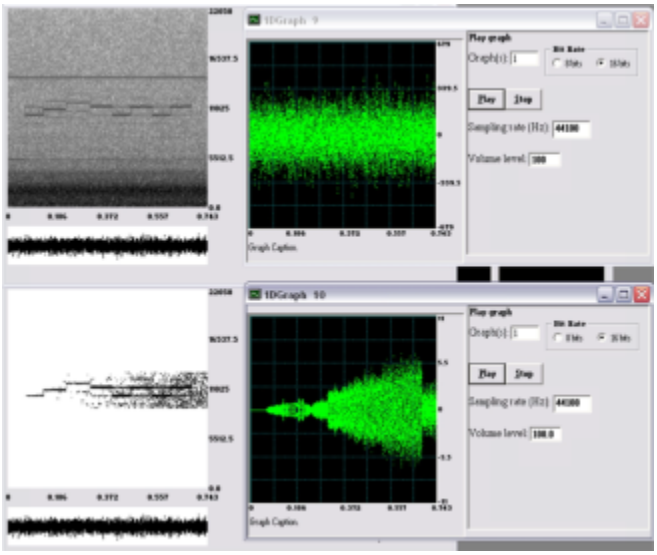

Fig. 7. Screenshot of FAWAVE, showing acoustic waveforms corresponding to an `h` typed on a system equipped with the acoustic keylogger and captured via a wireless X10 spy camera

Attackers may also improve their reception accuracy by filtering the audio before processing. In fact, this is a very simple task that can be automated using a Windows freeware program, FAWAVE [5]. The top window in Figure 7 shows an acoustic waveform that we cut from a larger file collected during a keylogger run. This waveform corresponds to the letter `h`, encoded as `01320202`. As can be seen from the spectrogram, the signal is very noisy, which would probably make it difficult to automate detection of the levels. In particular, the low-frequency noise from the speakers and hard drive shows up as a thick, dark band along the bottom of the spectrogram. To eliminate these spurious elements, we used FAWAVE to perform both bandpass filtering, to eliminate all frequencies outside a 4kHz band centered on 11kHz, the approximate central frequency of our levels, and all signals whose amplitude was beneath a threshold value. After this processing, the levels are much more clearly defined, as can be seen in the lower window in Figure 7.

In the experiment just described, the signal was collected using an X10 wireless camera placed inside the computer's case. This demonstrates that even an inexpensive, easily-concealed microphone can record signals with sufficient fidelity to enable effective eavesdropping.

9. Recommendations

We have demonstrated that acoustic emanations from computers can both reveal information about general computations being performed on a machine, and can be

manipulated to provide a fairly reliable, surreptitious data channel. Thus, sensitive systems should be configured to minimize acoustic output. As we have shown, more aggressive power management amplifies acoustic emanations, and should thus be disabled on sensitive systems. It may also be necessary to acoustically isolate sensitive systems by sweeping their interiors for hidden microphones and wrapping their exteriors in sound-absorbing foam.

10. Future Work

Our study should motivate future work on minimizing the usefulness of acoustic emanations from computer components. It may be possible to develop new compiler techniques that minimize the usefulness of acoustic emanations by carefully arranging instruction sequences. Operating systems should also be designed to carefully control access to power management functions.

We explored only a single attack application that could make use of data transmission via acoustic emanations. Any application that wishes to surreptitiously transmit narrow-bandwidth data from a physically unmodified computer could benefit from the techniques presented in this paper. More work should be done to identify those applications and evaluate the additional risks that the existence this side channel could introduce.

There are many methods to detect acoustic emanations. We would be interested in exploring the feasibility of detecting emanations from a long distance, using a laser interferometer [10] or some other such device.

Multicore systems almost certainly have very different power consumption characteristics than their single-core counterparts, which could change the results of our experiments significantly. However, it will take some time for software developers to fully exploit the parallelism offered by these new processors, so they will likely continue to behave somewhat like single core systems for some time. Also, it may be possible for a resourceful attacker to synchronize the actions of a majority of cores and thus cause predictable variations in power consumption similar to those observed in this paper.

Virtualization is another influential technology that could affect our results. Virtual machines are often used to contain system vulnerabilities. Thus, they provide a barrier between the virtual machine (VM) and the actual system hardware that can be nearly impossible to breach if the VM monitor is properly configured. However, most virtualization products place a premium on performance and many even provide a way for virtualized instructions to execute unmodified on the system hardware. In these systems, our attacks would undoubtedly apply unchanged if no sensitive or privileged instructions were used, but it is not clear how effective they would be in a more thoroughly emulated virtual machine.

Finally, it would be interesting to push beyond acoustic emanations and explore ultrasonic emanations. Since even acoustic emanations from computers are concentrated around relatively high frequencies, there are almost certainly interesting emanations in the ultrasonic spectrum that could be analyzed using techniques similar to those in this paper.

11. Conclusion

In this project, we have shown that the acoustic emanations observed by Shamir and Tromer on a single system are in fact produced by other, more modern systems, and that they do reveal information about the computations being performed within a machine. We have also attempted to explain the origins of these emanations, using the additional observations we obtained from our system with aggressive power management features. We then designed and performed experiments to explore the full range of emanations that can be produced by a computer, and showed how they can be manipulated to surreptitiously transmit data. We also applied these techniques to a popular application that could compromise system security, a keylogger for Linux. That discussion was accompanied by suggestions for filtering acoustic data to accentuate the compromising emanations, and a general discussion of techniques that could be used to automate and enhance data reception. Finally, we recommended ways to minimize the risk of compromise from acoustic emanations.

ACKNOWLEDGMENTS

Michael LeMay was initially funded by the UWEC Center of Excellence for Faculty/Student Research Collaboration, and then supported by the Air Force Office of Scientific Research during the composition of this paper. Additionally, the UWEC physics department graciously provided their lab space and assistance during our initial interferometry experiments.

REFERENCES

[1] A. Shamir and E. Tromer, *Acoustic cryptanalysis, On nosy people and noisy machines*, http://www.wisdom.weizmann.ac.il/ tromer/acoustic/

[2] D. Asonov and R. Agrawal, *Keyboard Acoustic Emanations*, In *Proceedings of 25th IEEE Symposium on Security and Privacy*, May 2004, pages 3-11

[3] D.X. Song, D. Wagner and X. Tian, *Timing analysis of keystrokes and SSH timing attacks*, In *Proceedings of 10th USENIX Security Symposium, Wahington DC, USA*, Aug 2001

[4] *Baudline signal analyzer*, http://www.baudline.com/contact.html

[5] J. Walker, *FAWAVE*, http://www.uwec.edu/walkerjs/FAWAVE/index.htm

[6] S. Wolfram, *Chinese Remainder Theorem*, http://mathworld.wolfram.com/ChineseRemainderTheorem.html

[7] M.G. Kuhn, R.J. Anderson, *Soft Tempest: Hidden Data Transmission Using Electromagnetic Emanations*, *Lecture Notes in Computer Science*, Volume 1525, Jan 1998, Page 124

[8] C. Comin, *LKL Linux KeyLogger*, http://sourceforge.net/projects/lkl

[9] *NACSIM 5000 Tempest Fundamentals*, http://cryptome.org/nacsim-5000.htm

[10] *LASER Microphone*, Williamson Labs, http://www.williamson-labs.com/laser-mic.htm

[11] *The GNU MP Bignum Library*, http://swox.com/gmp/