# RSA Key Extraction Using Acoustic Cryptanalysis

Kevin Au, Omar Jatoi, Jennine Nash

**Abstract**

The paper *RSA Key Extraction via Low-Bandwidth Cryptanalysis* (2013) showed that an older version of GnuPG is vulnerable to an acoustic side-channel attack capable of leaking an entire RSA secret key. However, some of the computers and operating systems that were tested were older than what an average user might be carrying. This paper draws upon this research and attempts to recreate the attack on newer computers and operating systems using the same vulnerable version of GnuPG. Two approaches were used, one with a cellphone to mirror the cellphone attack detailed in the reference paper and one with an unmodified lab-grade microphone. The data from the experiment indicates that newer models are difficult to attack with both approaches. Though acoustic leakage was measured, it proved too difficult to extract an RSA key. However, this paper provides background for further research in this field.

## I. Introduction

Most people who are familiar with computer security and cryptography know that the strength of the RSA cryptographic scheme comes from the difficulty of factoring large numbers. The scheme is believed to be secure because for sufficiently large numbers, the best known methods to crack the scheme are much too slow. However, though the algorithm is currently theoretically sound, the amount of security that schemes like RSA are actually able to provide depends on their implementations -- both in software and in hardware. Thus, rather than trying to attack a scheme by solving a difficult problem, one can look for ways to attack the manner in which it is written and runs.

Side channel attacks use information leaked unintentionally from a piece of software or hardware to gain knowledge about the algorithm running or data being handled. This is one way to break RSA without actually breaking the RSA algorithm. Side channels can take many forms: cache activity, power consumption, file size, network activity, timing of operations, even sound waves [1]. In this paper, we focus on a side channel of leaked sound wave information.

Acoustic cryptanalysis is the process of recording sounds made by a computer as it is performing some cryptographic function, and using the resulting

information to infer sensitive information: the algorithm running, the key being used, and other potentially sensitive pieces of information. The paper *RSA Key Extraction via Low-Bandwidth Cryptanalysis* by Daniel Genkin, Adi Shamir, and Eran Tromer (referred to henceforth as "the reference paper") describes a procedure for doing just this [1]. In particular, they claim that this attack strategy can be used to extract entire RSA keys under certain conditions. Additionally, they discuss some other ways acoustic cryptanalysis can be used, such as to determine what operations are being run on a computer or identifying which of two keys is putting a digital signature on a document [1].

The "certain conditions" that the reference paper describes are on older laptops (e.g. Lenovo Thinkpad T61), on older operating systems (e.g. Windows XP), and with microphones ranging in quality from expensive, lab-grade setups to easily obtainable smartphones [1]. In the interest of furthering this research, we decided to attempt to duplicate their results on newer equipment. The goal was to determine whether the attacks outlined in the paper are feasible for the average hacker wanting to steal secrets from the average victim.

Our objectives were to first be able to reliably determine what operations were being performed on a target computer using only the sound waves emanating from the hardware. Then, we would attempt to differentiate between two different RSA keys signing the same document. Finally, if possible, our ultimate goal was to extract a full RSA key from the target computer.

First, we do a literature review of the reference paper in which we discuss their claims of the attack's effectiveness, based on their results. Then, we describe the equipment we used in our own experiments and the different configurations we tried. After that, we present the results of our experimentation and analyze their implications. Next, we compare the outcomes of our experiments with those of the reference paper. This will involve discussing which claims we were able to corroborate and which need further study. Finally, we discuss some possible directions for future work that we feel may be beneficial to this area of study.

# II. Literature Review

The reference paper details a successful attack against GnuPG 1.4.14 RSA using acoustic cryptanalysis. The attack hinges on recording the sounds made during a chosen ciphertext attack and analyzing them to determine entire 1024 and 2048-bit RSA secret keys. It does this by crafting one ciphertext for each bit in the key (each ciphertext dependent on the results of the previous attack), and based on the acoustic emanations, determining whether the current bit is a 1 or a 0. On the

way to executing the full attack, the authors of the paper were able to obtain various other pieces of information about the target computer, including what operations it is running and which of a number of keys it is signing with [1].

The reference paper has a much wider breadth than our experiments did, and also got further into the attack than we were able to. As a result, we will not be looking at all of the claims in the paper. Some involved side channels other than acoustic cryptanalysis, and others involved different ciphers (besides RSA) or different versions of GnuPG. We are focused only on acoustic cryptanalysis of GnuPG 1.4.14's RSA algorithm, using a lab-grade microphone and smartphone microphone. In addition, some of the analysis done in the reference paper, for example on the crafting of advanced ciphertext, would have been incorporated into our paper only if we had been more successful with earlier parts of the attack. Therefore, these claims will also not be enumerated.

The claims we will be investigating as stated in the reference paper are as follows [1]:

1. Acoustic emanations can be used to distinguish between the different computer operations sleep, multiply, and memory access.
2. In signing the same document with two different keys, it can be reliably determined from the audio output that the two keys were different.
3. In signing a document with the same key twice, it can be reliably determined from the audio output that the keys were identical.
4. Full 1024 or 2048-bit RSA secret keys can be extracted one bit at a time by executing chosen ciphertext attacks and recording the sounds made while decrypting the ciphertexts.
5. When attacking each individual bit, there is a distinct difference in frequency depending on whether that bit is a 0 or a 1.
6. The highest-quality and most reliable recordings were made with a lab-grade setup using a Brül&Kjær microphone using three different capsules: the 4939 with 350 kHz sampling, the 4190 with 40 kHz sampling, and the 4145 with 21 kHz sampling. In addition, this setup used a Brül&Kjær 2669 preamplifier, a Mini-Circuits ZPUL-30P amplifier, and a Brül&Kjær 2804 power supply. It also used a 10 kHz RC high-pass filter cascaded with a 150 kHz RC low-pass filter while recording the audio. The microphone was placed about 1 meter away from the target computer and aimed at the fan or ethernet port.
7. The attack was also possible using a cell phone microphone, though the data was of lower quality. The cell phones they used were a Samsung Galaxy S2 and a Samsung Note S2. Both of these were running a custom

Android application to record. The cell phones were placed approximately 30 centimeters from the target computer's fan or ethernet port. The specifications of the cell phone microphones were not available.

8. The target laptops which had the strongest signal (and therefore the most effective attacks) were the Lenovo Thinkpad T23 (2001), the Lenovo Thinkpad T61 (2007), and the Compaq Evo Notebook N200 (2002), all running Windows XP.
9. The reference paper contains the following quote: "Our observations apply to many laptop computers, of various vendors and models, running various operating systems."

Later we will compare our results to those of the reference paper and, in the process, decide if their claims are corroborated, refuted, or need further examination and research.

# III. GnuPG Vulnerability

The key component of this project which makes exploitation possible is the existence of a vulnerability in GnuPG version 1.4.14, which causes it to leak telling acoustical signatures while performing cryptographic operations. The essential source code to observe in GnuPG is that of its modular exponentiation algorithm on very large numbers (represented by arrays of 32-bit integers called limbs). Figure 1 presents the pseudocode found in the reference paper.

**Algorithm 1** GnuPG's modular exponentiation (see function mpi_powm in mpi/mpi-pow.c).

**Input:** Three integers $c$, $d$ and $q$ in binary representation such that $d = d_n \cdots d_1$.
**Output:** $m = c^d \mod q$.

```
1: procedure MODULAR_EXPONENTIATION(c, d, q)
2:     if SIZE_IN_LIMBS(c) > SIZE_IN_LIMBS(q) then
3:         c ← c mod q
4:     m ← 1
5:     for i ← n downto 1 do
6:         m ← m^2
7:         if SIZE_IN_LIMBS(m) > SIZE_IN_LIMBS(q) then
8:             m ← m mod q
9:         if SIZE_IN_LIMBS(c) < KARATSUBA_THRESHOLD then        ▷ defined as 16
10:            t ← MUL_BASECASE(m, c)                  ▷ Compute t ← m · c using Algorithm 3
11:        else
12:            t ← MUL(m, c)                           ▷ Compute t ← m · c using Algorithm 5
13:        if SIZE_IN_LIMBS(t) > SIZE_IN_LIMBS(q) then
14:            t ← t mod q
15:        if d_i = 1 then
16:            m ← t
17:    return m
18: end procedure
```

Figure 1. Modular exponentiation pseudocode from the reference paper [1]

The chosen ciphertext that the reference paper authors found to be most effective used the correctly identified first $i$ significant bits of the secret key's prime factor $q$ (the most significant bit of $q$ is always 1). Given these previously extracted bits, the ciphertext $c$ would be the following: $q_0q_1...q_i011...1$. That is, the first $i$ bits would be the already extracted bits of $q$, the target bit would be a 0, and the remaining bits would all be set [1].

The reason why this ciphertext uncovers a vulnerability in GnuPG is because of the pseudocode presented above. Notice that on line 3, the ciphertext is altered depending on whether or not $c$ has more limbs than $q$. Also note that, looking at the chosen ciphertext, $c$ will always be greater than or equal to $q$ if $q_{i+1}$ is 0 and always less than $q$ if $q_{i+1}$ is 1. Therefore, depending on the targeted bit of $q$, the $c$ used in the code will either be the long and very repetitive chosen ciphertext or a shorter random-looking bit string caused by the modulus on line 3 [1]. What is not apparent from the pseudo code, but which the reference paper shows experimentally, is that the multiplication algorithm used by GnuPG behaves differently on longer repetitive strings than on shorter, random-looking strings [1].

Because of this difference in multiplication behavior, the acoustic leakage from the laptop performing modular exponentiation will be distinctly different depending on whether the bit currently being attacked is a 0 or a 1. Making one ciphertext for each of these bits and observing the acoustical outputs can therefore reveal to us the entire prime factor $q$ [1]. Though the private key includes both

primes $q$ and $p$, knowing $q$ is sufficient for obtaining the entire secret key since the product $N=pq$ is public.

# IV. Project Setup and Procedure

For the sake of consistency, we followed the reference paper's setup as closely as possible. Although we were unable to acquire the exact same equipment as the reference paper, we were able to obtain pretty reasonable equipment that can surely be classified as a lab grade setup. Since our scope was to test the attack on as many newer devices as possible, we got a pool of four different modern devices running different operating systems. This section will first outline every piece of hardware on which we conducted the experiment, along with every piece of software we used. Next will be a detailed description of the project setup and the steps taken per iteration of the experiment.

The experiment consisted of two core hardware components: the target device itself and the recording equipment. Though the paper mentioned the possibility of this attack being conducted on desktops, we restricted our focus to laptops as the reference paper did. The laptops used were as follows (in descending order in terms of product age): Macbook Pro (2014), Surface Pro 3 (2014), Lenovo Thinkpad L520 (2011), and the Sony VAIO PCG-81312L (2011). The acoustic leakage of each laptop was tested on more than one operating system, all of which will be described under the software section of the setup. All four of these devices represent a good sample of modern computing, including a compactly designed Apple device, a 2-in-1 tablet-laptop ultrabook by Microsoft, and two portable workstations by Sony and Lenovo.

The second major component for the project was the audio recording equipment. The reference paper mentioned the feasibility of standard mobile phone microphones which we initially relied upon. We used a Samsung Galaxy S5 and tried several different recording applications before settling for Pure Audio Recorder Free, a basic recorder that was capable of recording .wav files to be later analyzed by an audio analysis program. After making little progress with the cell phone, we acquired a lab grade setup consisting of a MOTU UltraLite-mk3 audio receiver and a Audix TM1 microphone. The MOTU audio receiver is capable of high quality audio at sample rates up to 192 kHz. The Audix TM1 is advertized as an accurate and sensitive microphone used by sound engineers [2].

As did the reference paper, we decided to use the open-source audio analysis software Baudline. Baudline is capable of receiving a live stream of

audio from the sound equipment and graph a spectrogram with time as the y-axis and frequency as the x-axis [3]. Having a live spectrogram from our microphone proved useful for visual analysis. We encountered situations where we needed to keep records of audio recordings and decided to use another open-source audio recording software called Audacity. Audacity is capable of a lot more than what we used it for - but for our purposes we were able to save 192 kHz recordings directly from our audio setup [4]. Figure 2 shows both Baudline and Audacity.

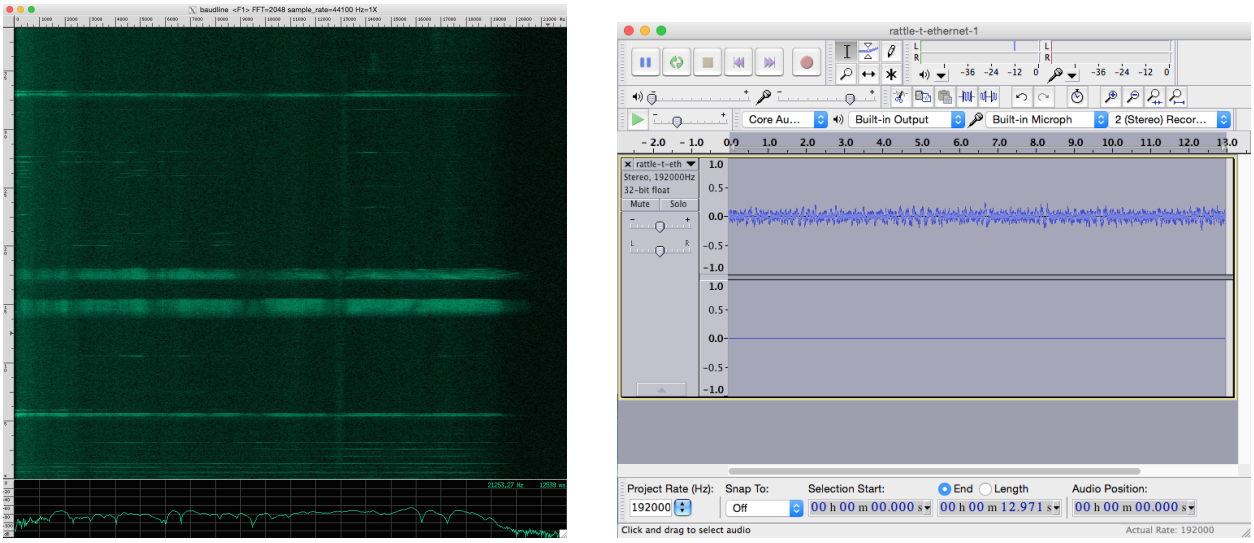



Figure 2. The left image is what a spectrogram looks like in Baudline, and the right image shows a recording in Audacity.

To prepare our devices for multiple iterations of experiments, we used the following operating systems on the following devices:

- Macbook Pro: Windows 7, Ubuntu 14.10, Arch Linux
- Surface Pro 3: Windows 8.1, Ubuntu 14.10
- Lenovo Thinkpad L520: Windows 7, Ubuntu 14.10
- Sony VAIO PCG-81312L: Windows XP

We were also fortunate to have been in contact with one of the authors of the reference paper (Daniel Genkin), who gave us two custom executables. The first was called rattle, which looped over various commands (HLT - x86 sleep call, MUL - multiply, MEM - memory access), with the purpose of first looking for acoustic leakage for some basic commands before looking for acoustic leakage for more fine grained algorihtms [1]. The second was run2, which looped over a decryption process using the vulnerable version 1.4.14 of GnuPG. For testing acoustic leakage on the Windows operating system we used these two

programs, whereas for testing acoustic leakage on devices running Linux we tried one of two methods: either running a slightly modified version of GnuPG version 1.4.14 source code which forced it to decrypt a hardcoded chosen ciphertext, or signing a document with a private key. In all cases we analyzed the data visually based on the spectrogram in Baudline.

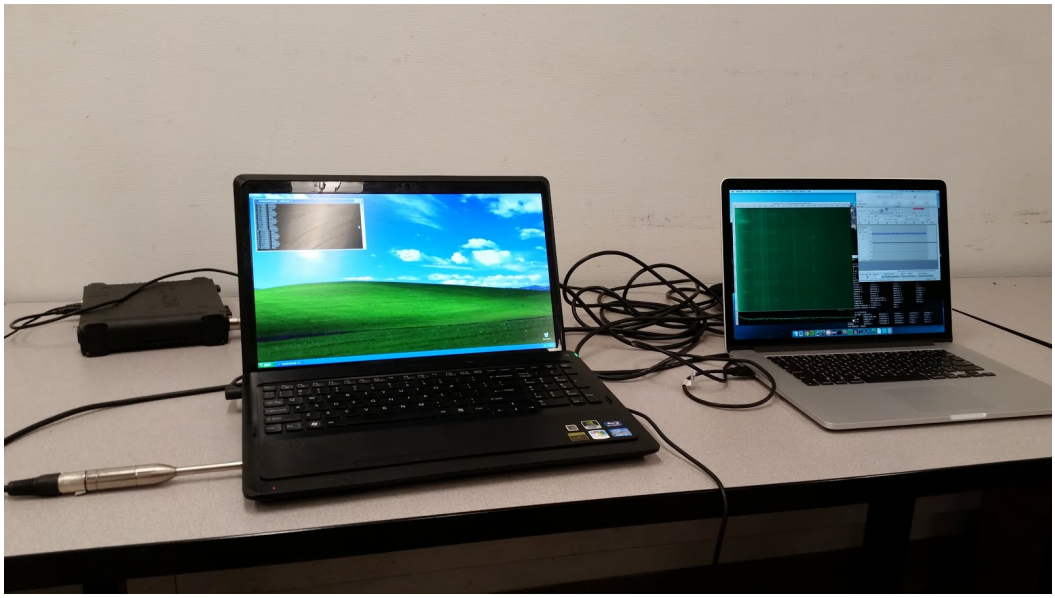


Figure 3. The Sony VAIO running the rattle program, hooked up to our lab grade microphone setup. Macbook Pro used as analysis device.


Figure 4. The phone setup for recording acoustic leakage.

Figure 3 shows the optimal setup that we used to get our final results using the lab grade microphone. Figure 4 shows the cellphone setup. Out of the four computers sampled, only one (the Sony VAIO) yielded any results. We mimicked the procedure from the reference paper as closely as possible, trying to recreate the setup from the pictures in the reference paper. Specifically, for the cell phone setup, the paper suggested the cell phone be approximately 30 cm away from the target computer's ethernet port or fan exhaust. The lab grade setup had the microphone pointing at the ethernet port or the fan exhaust and from 1 meter away [1]. We initially copied the setup exactly as suggested in the reference paper. This did not work, because our microphone was not sensitive enough. The cell phone microphone was also not sensitive enough to pick up signals from 30 cm away. However, it is likely that the laptops we tested had weaker acoustic leakage and therefore variations from the distance presented in the paper were expected in our setup. To account for these variations we tested many distances between the target device and the microphone, starting with a meter for the lab-grade setup and halving the distance for each separate test. Similarly, for the cell phone we started with 30 cm away and halved the distance each iteration. We found that there was little to no acoustic leakage at lengths over a half a centimeter away from the target computer. Results were only found when our lab grade microphone was inserted inside the ethernet port and when the cell phone's microphone was placed directly outside the port. This suggests two things: first that the laptops we used had little to no acoustic leakage, and second that the microphone and overall audio setup was not sensitive and/or powerful enough to get acoustic leakage from the device. The reference paper made use of a custom amplifier to tackle signal attenuation, but unfortunately we did not have access to this hardware.

Figure 5. On the left is the microphone placement that yielded results. On the right is the microphone only a few centimeters outside of the ethernet port, and this setup yielded no results.

# V. Results

Eventually we were able to see results on one of the four laptops tested. It is important to note that only the Sony VAIO gave us any leakage that was visually apparent. The Macbook, Thinkpad, and Surface Pro 3 all failed to leak any visible data. One possible reason for this is because neither the Macbook nor the Surface Pro 3 have an ethernet port, so getting as close to the inside of the machine as possible posed a challenge. These devices were tested with the microphone pointing directly towards the fan exhaust.

The Sony VAIO leaked an acoustic signal only when our lab grade setup microphone was inserted directly into the ethernet port. We qualified a "result" as visually seeing a pattern in the spectrogram. Once we had done this, we recreated the results and made audio recordings to analyze them. We saw distinct leakage when running both the run2 and rattle programs, and very low quality signal leakage during key signing. It was still enough to allow us to distinguish separate keys, but only barely. With the given quality, it was difficult to differentiate between two acoustically similar keys to determine whether or not the keys themselves were identical. This may be done using a machine learning algorithm given some basic data, though this was outside the scope of our project. Figure 6 shows two keys signing the same document.

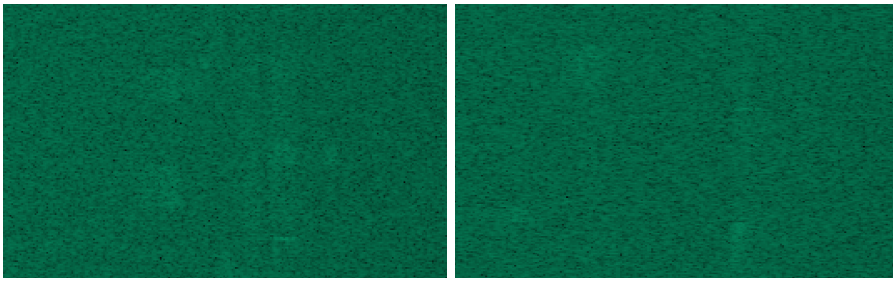


Figure 6. On the left is key one signing a document, and one the right is key 2 signing the same document.

The rattle program running a loop of basic instructions yielded significant results. We were able to repeat the experiment over numerous iterations and got the same results each time. A clear pattern can be seen between each system call, and though the quality is not excellent, it is substantial enough to guess the kind of program running. It may even be possible to use advanced algorithms or analysis techniques to reverse engineer the code being executed based on just the system instructions. In figure 7 is an image of the spectrogram showing this acoustic leakage pattern along, and figure 8 shows a screenshot of the frequency

analysis. The significant leakage for the multiply (MUL) and memory access (MEM) instructions can be seen mostly in the 50-75 kHz range, though the sleep instruction (HLT) seemed to create a pattern in the lower frequency range as well (15-35 kHz). All leakage was very quiet, in the -60 to -78 dB range.

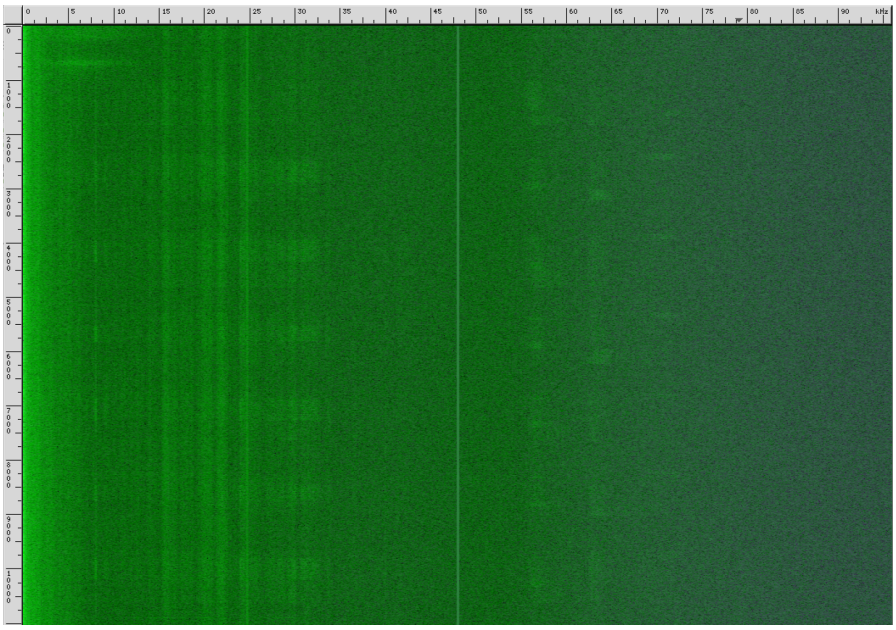


Figure 7. Leakage from the rattle program running on the Sony VAIO. Microphone inserted directly into ethernet port.

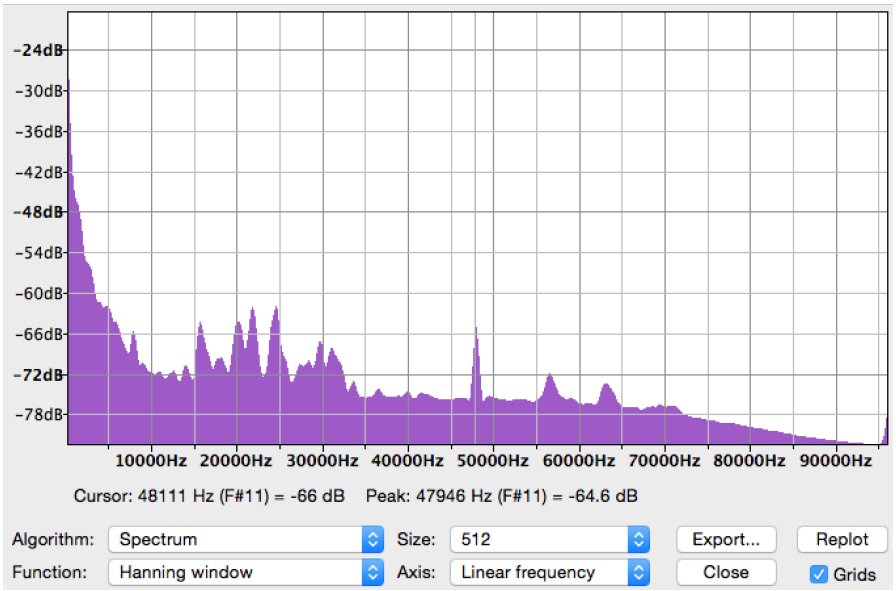



Figure 8. Graph of the frequency analysis from a recording of the rattle program running on the Sony VAIO.

The other quantitative results we saw were during the execution of the run2 program on the Sony VAIO. This program essentially looped through the decryption process from a slightly modified version of GnuPG version 1.4.14, with the sole purpose of making it relatively easy to spot acoustical leakage, and was provided to us by one of the authors of the reference paper. During this experiment, the acoustic leakage pattern we saw was in the 55-75 kHz range, and the frequency analysis of a recording showed volume range for the leakage similar to the execution of rattle. Specifically it was very quiet in the range of approximately -60 to -78 dB. Again, it is important to note that the pattern seen in the spectrogram was repeating very consistently. To illustrate this, in figures 9 and 10 are two spectrograms from two separate recordings, followed by the frequency analysis of only one of them in figure 11 (the analysis did not differ, and so the slight variations in the graph were just from ambient noise).

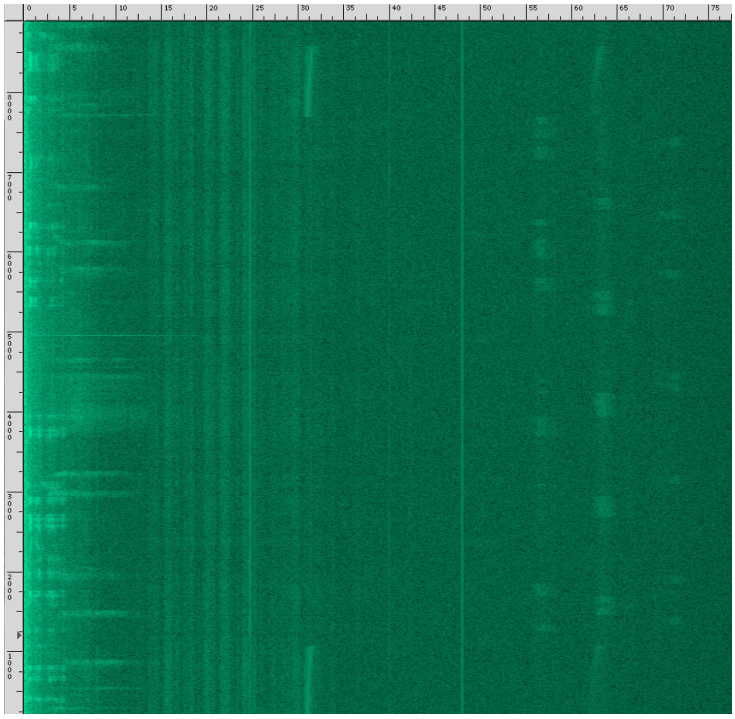


Figure 9. Iteration 1 of the run2 program recording, showing a clear pattern.

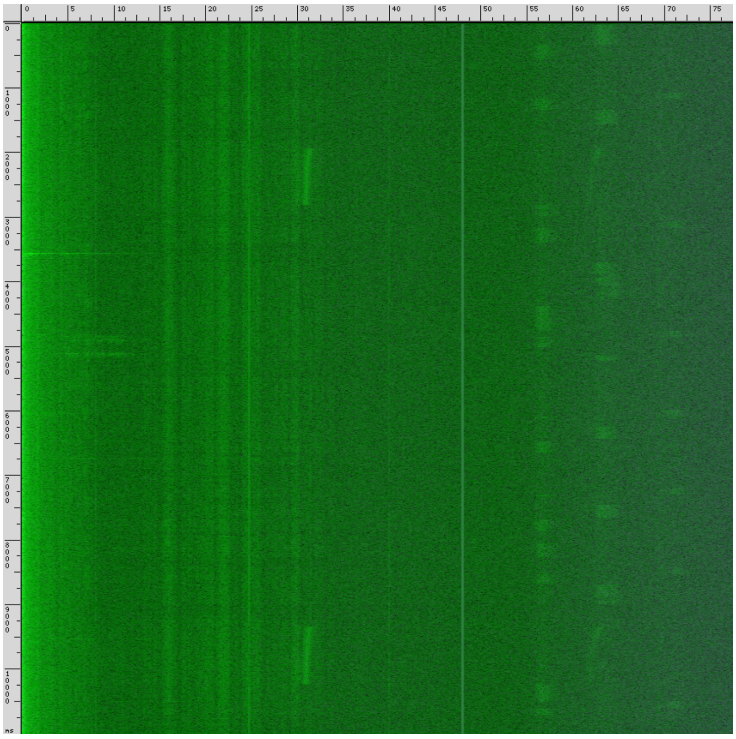


Figure 10. Iteration 2 of the run2 program recording, showing clear similarities to the pattern in the first iteration.

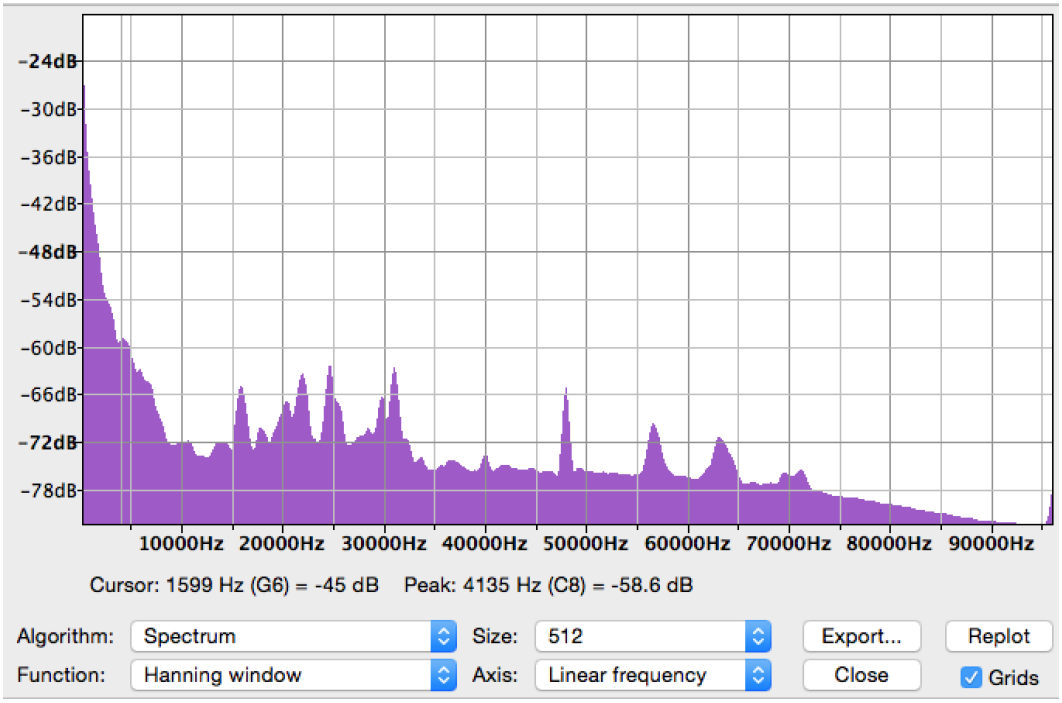



Figure 11. Frequency analysis of the run2 program, showing the loudness of the signals that we care about in the -60 to -78 dB range.

The cell phone setup (using a Samsung Galaxy S5) yielded results as well, though the results were a lot less impressive and we were not able to differentiate between different system instructions (though a pattern was evident). Using the cell phone as the microphone yielded no results with the execution of the run2 program, but the rattle program gave us a recognizable pattern that led us to conclude that the cell phone was able to pick up patterns in the lower frequency range between 10-25 kHz. The cell phone microphone was only capable of a sampling rate of 48 kHz, so any leakage in the higher frequency range was simply cut off. Another limitation from the cell phone was noise; our recordings were much noisier since we were still recording within human audible ranges and ambient noise was able to attenuate our acoustic leakage signals a lot more than in the lab grade setup. Figure 12 presents a spectrogram from an iteration of the rattle program executing on the Sony VAIO. Again, the cellphone had to be very carefully placed to have the microphone lined up exactly with the opening of the ethernet port.

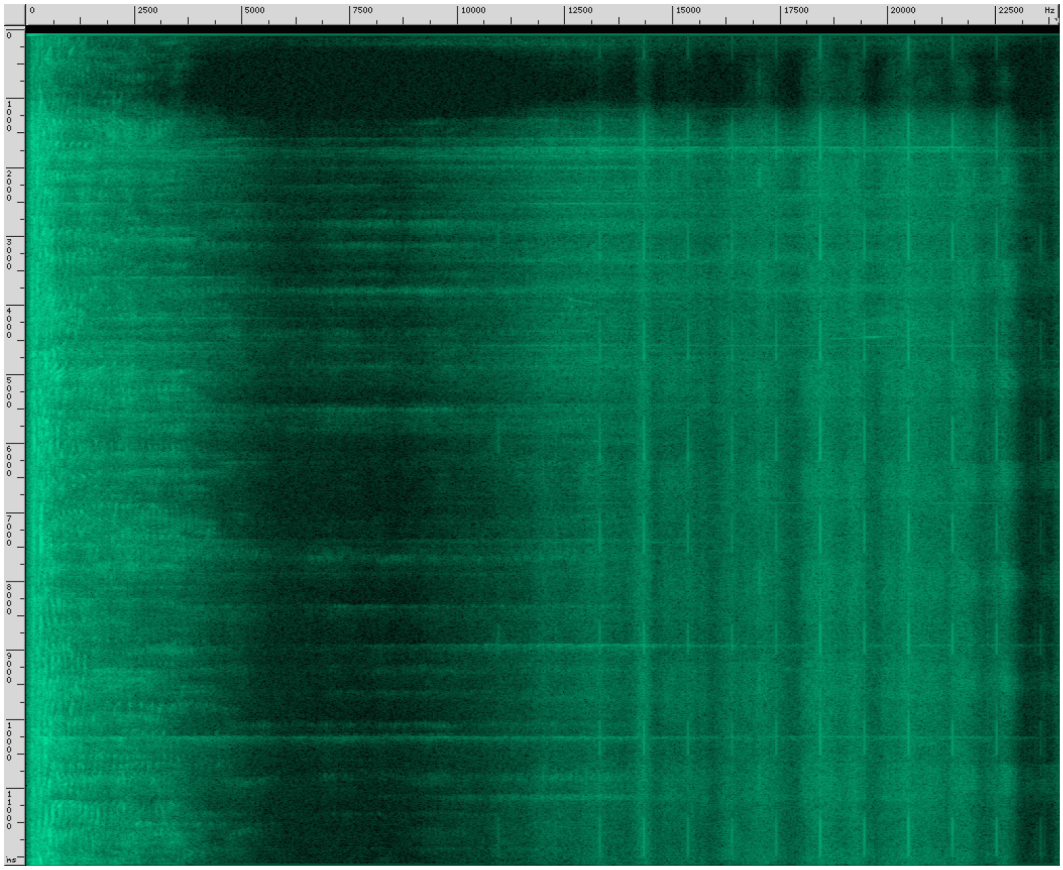



Figure 12. Spectrogram from a recording of the rattle program executed on the Sony VAIO, recording using a Samsung Galaxy S5 built in microphone.

# VI. Analysis of Results

The results show that the attack has limited success on newer computers. In particular, given the limitations of the setup used, no acoustic leakage was measured on three of the four computers. Furthermore, the acoustic leakage that was measured was not high enough in quality to extract a key. However, it is notable that on the computer which showed acoustic leakage, an attacker could find out to some extent when specific acoustically significant function and being called and with some degree of certainty know which key is being used to sign a document at a given point of time.

As indicated in the reference paper, even coarsely grained leakage of sensitive data can be useful to an attacker. For instance, being able to see specific cryptographic operations may indicate to an attacker what information would be useful to target. One might imagine that if an attacker knows when a document was signed using a specific key, that attacker would be better equipped to determine what was signed. It would at least limit the search space to a specific time frame. In addition, one can imagine that if a private key is shared among multiple computers, the acoustic signature could be used to identify such sharing.

However, there are some notable drawbacks to such an attack. Both attacks proved physically difficult to implement. The microphone had to be very close the target computer to record any acoustic leakage on tested computers. One possible reason for this (which is outside the scope of this project) is that newer computers have more efficient processors, so perhaps there is less acoustic energy to leak. This problem, along with the high quality audio recording necessary to effectively pull off an attack, lead to a couple conflicting considerations for the attack. The microphone must be placed very close to the internal components of the target's computer, yet be sensitive enough to discern the acoustic leakage to a high enough level. Our setup would probably not be sufficient for a real attack, but perhaps a funded attack could be.

# VII. Limitations

The limitations the group faced came in two flavors: limitations in equipment and setup and limitations in time. Our limitations in equipment narrow the scope of the project because they reduce the generalizability of any results of the project. In addition, limits in equipment mean that perhaps with a different setup it would be possible to obtain much better results on newer computers. On the other hand,

limitations on time gave rise to simplifications to the project and an inability to build a fully functional attack even with the guidance of the authors.

One of the main limitations we came up against ourselves while doing this project was that of access to recording equipment. The reference paper described the use of a very expensive microphone with capsules of varying sensitivities. They also had a preamplifier, amplifier, filters to apply to their signals while recording, and a custom Android application to use in the cell phone attack [1]. While we may have been able to obtain some of this equipment for our experiments given more time, much of it we could not realistically get. It may also be the case that the average attacker would also not be able to obtain this equipment. Furthermore, there is reason to believe that the low quality of our results may have been due to this lack of more advanced recording equipment. With a more sensitive microphone and an amplifier giving us a stronger signal, we may have been able to observe the frequencies of the acoustic leakage more precisely, and therefore had more success in extracting full RSA keys.

The other main limitation that held us back throughout this project was the lack of access to many different types of laptops, particularly older ones. While our goal was to see if acoustic cryptanalysis was successful on newer laptop models, it still would have been best if we had first perfected our procedures on older models. However, these were difficult to find. Also, since we did not put any money into this project, we were not able to get a very wide variety of laptop models, even modern ones. We had to make due with the laptops each of us already owned. In the reference paper, it was noted that the quality of acoustic signals was directly proportional to the target laptop's age [1]. Also, very specific laptops were put forth as the most effective targets for the attacks, as discussed in the Literature Review section. Therefore, it is reasonable to expect that had we had access to more and older computers, the cryptanalysis would have gone much more smoothly.

Not only was the above issue, target laptop availability, a major limitation for conducting our own experiments, but we expect that it would also be a significant problem for any attacker trying to use acoustic cryptanalysis in the real world. In most scenarios, an attacker would be targeting a specific person's computer, or a specific corporation (many of which give all their employees only a few different models of computers). Therefore, they would also be limited in what types of laptops they could choose to attack. If it happens that all of the attacker's potential targets are using laptops which are not vulnerable to acoustic cryptanalysis, then the exploit will be infeasible. This would be especially concerning to the attacker if, as may be the case, the vast majority of laptops are not vulnerable enough to this attack to leak full RSA keys.

The final limitation we encountered, which would mostly affect an attacker and not a team working in a lab environment, is the distance that the microphone must be from the target computer to get any results. As discussed in the Setup and Results sections, in order to get any acoustic signals at all on our spectrograms, the lab-grade microphone needed to be inside the target's ethernet port, and the cell phone microphone propped up to be touching directly against the ethernet port. Even a slight shift in position away from the target caused the spectrogram to go completely blank. Obviously, this is no good for an attacker, if even partial information cannot be obtained by recording from a reasonable distance. It is extremely unlikely that a target would not be suspicious of someone sticking a probe into their computer's ethernet port, or touching their laptop with their cell phone. There is a possibility that using a better microphone would have fixed this issue in our lab-grade setup, and that using the same custom Android application as used in the reference paper would have fixed it in our cell phone setup. It is also possible that the issue lies again in what target laptops are available, which as we discussed before is a major issue for an attacker. No matter what the root of this limitation was, it is the case that an attacker attempting to perform acoustic cryptanalysis would come up against this issue during the course of their attack.

# VIII. Comparison with Reference Paper

Though our results were not as crisp or reliable as those in the reference paper, and we were not able to execute the full secret key extraction, we were still able to reproduce many of the paper's intermediate results.

We used newer target computers and less advanced recording equipment (for the most part) than that used in the reference paper, so that may be the explanation for some of the instances in which we were not able to corroborate their results. However, we still see these inconsistencies as important, because part of our purpose was to investigate how practical this attack was on the general user, who may not be using a very old laptop. In addition, the recording equipment we used in our experiments was much more easily obtainable than the equipment used by the reference paper authors due to there being less of it and it being less expensive.

Claim 1, that different computer operations could be distinguished from each other, was validated in our experiments. As noted in the Results section, we were able to clearly see differences in the spectrograms of the SLEEP, MUL, and MEM operations. However, these signals were fairly noisy and spanned over a

relatively wide range of frequencies, causing each operation's acoustic signature to look like a collection of smudges rather than a collection of crisp lines. Figure 13 compares the spectrograms of the individual operations we took, with the spectrograms of individual operations in the reference paper. However, the noise in these recordings was not enough to make the operations indistinguishable, so we consider this as having successfully supported the claim made in the reference paper.

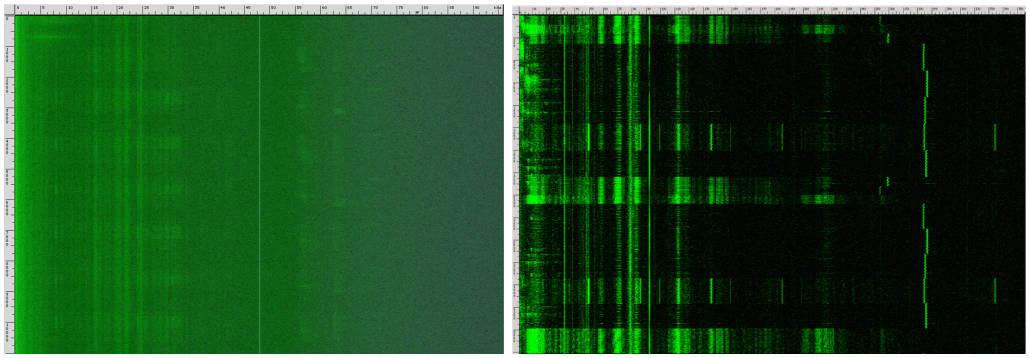


Figure 13. On the left is our spectrogram of individual computer operations; on the right is the same spectrogram from the reference paper [1].

Claim 2, that two RSA keys signing the same document would have distinctly different signatures, was confirmed by our experiments. As shown in the Results section, when signing the same document with two different keys, the two spectrograms definitely have clear differences. The only problem, however, is that the spectrograms are very faint and hard to see. They are relatively noisy, more noisy than the spectrograms for decryption and individual operations. So, that may cause problems if there is a lot of background noise or the two keys just happen to have very similar spectrograms. It is a stark comparison in figure 14 when one looks at how difficult it is to see our signing spectrograms as compared to the signature spectrograms from the reference paper. However, in most cases we believe that one would be able to see the difference between two signing keys, so this claim is true even for newer computers on less advanced recording equipment.

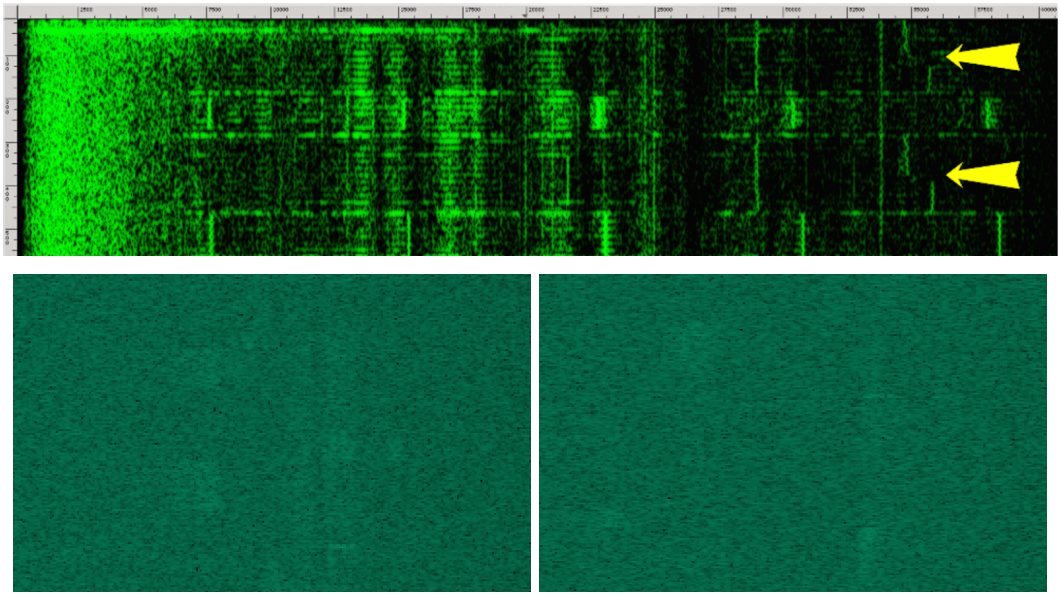


Figure 14. The top image is the comparison between two different signing keys in the reference paper (each yellow arrow points to the middle of each signing operation) [1]. The bottom image compares two different signing keys from our results.

Claim 3, that each RSA signing key has a distinct signature which is identifiable over multiple signatures, was only partially corroborated by our experiments. As discussed in the Results section, signing the same document twice with the same key resulted in two spectrograms which looked similar, but not exactly the same. This is in contrast to the spectrograms provided in the reference paper, which looked identical (to the human eye at least) for the same signing key. This can be seen in figure 15, which compares two of their signatures using the same key. It would most likely not be possible to mistake those two signatures as being from different keys, although this was not the case in our recordings. This is especially true if the two keys just happen to have similar spectrograms. Because of this, two identical keys could probably be identified in some cases. The discrepancy between our results on this experiment and the reference paper's results may have been due to noisier hardware in general.

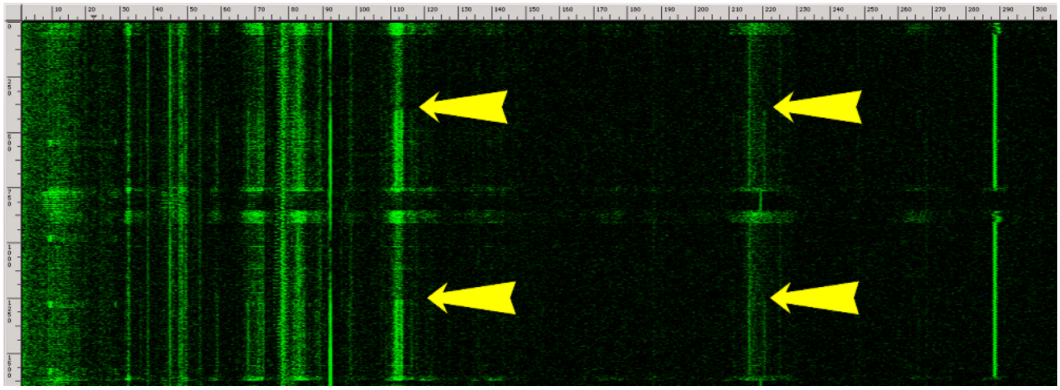
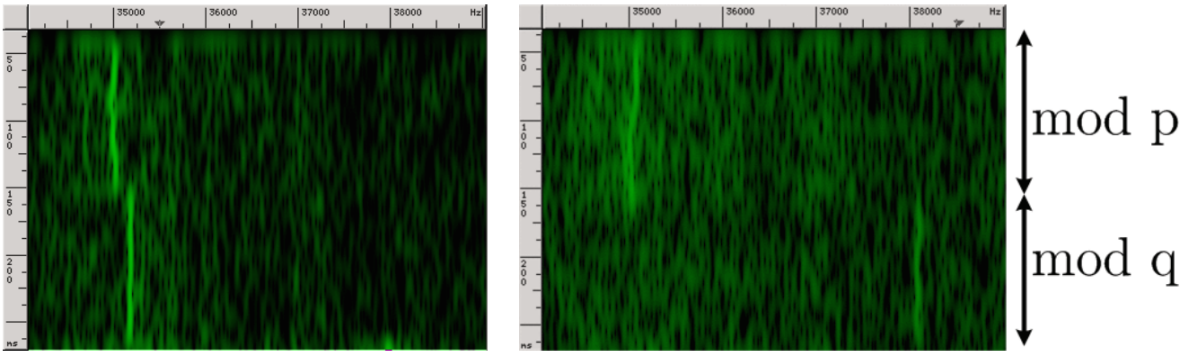


Figure 15. Two identical key signings from the reference paper. Each yellow arrow points to the center of one of the signings [1].

Claim 4, that full RSA secret keys can be extracted bit by bit using a chosen ciphertext attack, was one that we could not confirm. This is mostly because we were not able to get to the point in our project where we could test this. However, there are quite a few indications that we would not have been able to do it. During the course of working on these experiments, we contacted one of the authors of the reference paper, Daniel Genkin. After we had our spectrograms of individual operations and full decryptions, we emailed them to him and he told us that they would be too noisy to gain any specific information about the RSA keys. However, in those recordings we were using our lab-grade setup with the microphone placed as close to the source of the noise as possible (inside the ethernet port). Short of finding an older laptop, obtaining a more expensive microphone, adding amplifiers and filters to our setup, or taking apart the computer, it would have been very unlikely for us to have gotten a strong and clear enough signal to extract full RSA keys. Thus, we decided that with easily obtainable equipment, this part of the attack is not very likely to succeed. However, more study could show that there are other, simpler methods we could have used to get a clearer signal.

Claim 5, that there would be a distinct distance in the decryption spectrogram depending on whether or not the currently attacked bit was a 1 or a 0, could not be corroborated for many of the same reasons claim 4 could not be corroborated. Since determining the difference in spectrogram between a 1 and 0 is part of the procedure for extracting full keys, it would not be possible to even begin to confirm claim 4 without first confirming claim 5. However, because the difference between a 1 and a 0 on the spectrogram is the shifting to the left or right of a very thin line (as seen in figure 16), the thick smudges that appeared on our spectrograms were most likely not fine-grained enough to see a difference

between 1 and 0 bits, especially as we got past the first bits of the key and the difference got less pronounced [1].




(a) attacked bit is zero (b) attacked bit is one

Figure 16. Spectrograms from the reference paper showing the visual difference between the attacked bit being 0 and 1 [1].

Claim 6, that the reference paper's lab grade setup was the best for achieving results, was partially verifiable. For one thing, we did not have all the same equipment. However, since our lab-grade setup, with less advanced equipment, gave us the best results, it makes sense that theirs would as well. In particular, the microphone they used was more sensitive than ours, so it likely could pick up the acoustic signals better. Plus, using filters and an amplifier would definitely cut out a lot of the noise that was obscuring our spectrograms and make our intended signals stronger. The one aspect of this claim that makes us skeptical is the assertion that strong, clear signals of the type pictured in the reference paper could be obtained with the microphone placed a meter away from the target computer's exhaust fan or ethernet port. As noted in the Results section, we got absolutely no results when the microphone was any farther away than inside the actual ethernet port. While it is possible that an amplifier would enable us to place the microphone farther away, it seems doubtful that it would have such a drastic effect. It suggests that adding an amplifier would allow a signal to be picked up from a meter away, when no signal at all could be obtained from one one-thousandth of that distance. More study would definitely be needed to verify whether or not that is true, since being able to place the microphone farther away from the computer would make this attack much more practical.

Claim 7, that the same attacks done with the lab-grade setup were possible with a cell phone microphone, could not be fully confirmed through our experiments. In the Results section we did discuss the fact that there was definite acoustical leakage picked up by the cell phone when recording both individual

operations and a full RSA decryption. However, it was not possible for us to distinguish between the different operations, and the decryption spectrogram was very faint. This is in contrast to the outcome for the reference paper authors, who were able to carry out the full attack using just a smartphone microphone. Additionally, our recordings were taken directly outside of the laptop's ethernet port, not 30 centimeters away, which the reference paper reported to be a suitable distance. Unfortunately, the reference paper did not include any spectrograms taken from the cell phone attack, so we cannot include a side-by-side comparison here. However, their images must have been relatively crisp in order to extract full secret keys. It most likely could not have been due to our recording equipment that there were such discrepancies, since the phone we used and the phones used in the paper were very similar (different versions of Samsung Galaxy phones, but there is little reason for that to be the culprit). Besides the phone itself, the authors did not use any other equipment. It could be that their custom Android recording application was better able to utilize the microphone than the one we downloaded, but we were not able to get a copy of the application they used, so that would require further study. Another possibility is that the cell phone attack is just not viable on the target computers we used. Though we did not exhaust the selection of possible laptops, this may suggest that newer models are not nearly as susceptible to this attack than older ones. More research would be needed on this front, however.

Claim 8, that there were certain target laptops running Windows XP which had the most reliable results, could not be fully corroborated because we did not have access to the same laptops for experimentation that the paper authors did. However, we did obtain strong evidence throughout our experiments that Windows XP was the most vulnerable operating system with respect to acoustic cryptanalysis. In fact, no other operating system showed us any acoustic leakage at all, and this included a slightly old Lenovo Thinkpad. While it was not one of the models used in the reference paper's experiments, it was of a similar type. Therefore, we agree with the authors that there is something particular about Windows XP which makes this attack more feasible.

Claim 9 states that acoustic cryptanalysis is effective on a variety of laptop models running a variety of operating systems. Though we cannot refute this claim because we only attempted to recreate the attack on a handful of different configurations, we are skeptical as to its truth. This is not only due to the fact that it was not true for us on the laptop computers on which we experimented. One of the main reasons for doubting this claim is because of a few things one of the paper authors, Daniel Genkin, told us when we contacted him. When we told him we were having trouble getting any results on the laptops we were using, he

stressed to us that it would be much easier if we found a Lenovo Thinkpad T23 to experiment on. He further stressed that it was important that we use Windows XP as the operating system on our target computer. It was stated in the reference paper that this configuration was one of the ones which yielded the best results, but it was by no me ans presented as the only one with results. It should not be the case that it was simply the only setup feasible without the reference paper's lab-grade setup, because the cell phone attack was reported to work on many laptop configurations as well. Despite this, we do not disbelieve what the reference paper reports on this matter. Genkin most likely suggested that setup to us in order to make it easier. However, we suspect that the variety of laptops vulnerable to acoustic cryptanalysis may not be as wide as the paper suggests. Instead, we expect vulnerabilities to be mostly limited to older laptops such as those tested in the reference paper. However, this is another hypothesis which would require further experimentation.

# IX. Future Work

There are many more equipment configurations we wish we could have tried, and many more experiments we wish we could have done in order to have more of a concrete answer on whether or not the attacks outlined in the reference paper are really feasible on modern computers and easily accessible equipment. However, there were many factors which made this difficult, which were discussed in the Limitations section. In this section we discuss some possibilities of future work in this area, or experiments that we believe would have been helpful in determining how realistic acoustic cryptanalysis is.

First of all, as far as we have been able to find, there have been no exact duplications of the research in the reference paper. Though duplicating the authors' work would not say anything about the practicality of this attack using different equipment, if the original results cannot be reproduced then it is probably not particularly useful to expand on them. So, an important aspect of future work would be to reproduce the reference paper's attack using the exact equipment described in the reference paper.

Probably the most important aspect of future work would be to make sure to include a much wider variety of modern laptops in the experiments than we had access to. Also, each of these laptops should have the ability to run GnuPG on multiple operating systems in order to fully test the capabilities of the attack. This would make sure the experiments were more exhaustive and, if it turns out that

the attack cannot be executed, one can be relatively confident that it was not just due to bad luck in choosing computers.

Another factor which may have contributed to full key extraction being infeasible for us was the lack of amplifiers or filters. Even though when asked, one of the authors (Daniel Genkin) noted that not having filters should not affect the quality of the results greatly, it should not be ruled out entirely. The lack of amplifier, however, may have actually been a large contributing factor to our poor signal clarity. One can assume that an attacker with access to a relatively sensitive microphone could also get their hands on an amplifier. Therefore, the practicality of acoustic analysis on modern equipment can not be ruled out until it is attempted with equipment that will strengthen and clarify the audio signals.

Adding an amplifier, however, will not help the cell phone attack, which was reported by the reference paper authors to need no additional equipment. There were a few differences between the paper's cell phone setup and our cell phone setup, though. First of all, the phone we used was a different model than the one they used (although similar). So, one possibility for future work would be to obtain a Samsung Galaxy S2 and Samsung Note S2 to perform the experiments with. Another possibility would be to obtain the custom Android application used for recording in the reference paper, since that may be a cause of difference in the signal quality. It should not be hard to obtain a copy of this. Since Daniel Genkin was willing to provide us copies of some of the executables used for testing in the reference paper, it is reasonable to expect he would provide us with the recording application. With the same phone and the same recording application, it would be made clear that any failure to extract RSA secret keys was the fault of the target computer being used. This is an especially important direction for future work, since in our opinion the possibility of acoustic cryptanalysis using a common smartphone was one of the more alarming claims made in the reference paper.

There are also more interesting experiments to run from a theoretical standpoint. For instance, it would be interesting to scientifically prove exactly what about newer computers and newer operating systems might make the attack less effective.

# X. Conclusion

As seen in the data gathered in our experiments, acoustic cryptanalysis proved possible on a newer machine with the setups we implemented, but at a resolution that was not adequate for full key extraction. In particular, it was shown that the MUL, MEM, and HLT operations could be distinguished acoustically and that

with limited success, two different keys could be distinguished when signing a document. The project shows that, despite the success of relatively low frequency acoustic cryptanalysis on older computers (as indicated by the success of a phone attack in the reference paper), it appears difficult to obtain similar results on newer computers. In addition, it appears that attacking a newer computer requires more specialized equipment than the lab-grade microphone that was available to us. However, the fact that acoustic leakage was measurable even with lower quality equipment indicates that there is a lot of promising unexplored territory to cover in respect to this approach.

# XI. References


[1] Genkin et al. “RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis.” *CRYPTO 2014.* http://www.cs.tau.ac.il/~tromer/papers/acoustic-20131218.pdf (visited on 12/7/2014).

[2] Feature Summary. http://www.motu.com/products/motuaudio/ultralite-mk3/summary.html (visited on 12/7/2014).

[3] Baudline. http://www.baudline.com/ (visited 12/7/2014).

[4] Audacity. http://audacity.sourceforge.net/ (visited 12/7/2014).