SPyAudio: A Configurable Framework for Sending Data Through a Covert Audio Channel

Arjun Rawal
arjunrawal4@uchicago.edu
University of Chicago
Chicago, IL

ABSTRACT

Preventing the external control of devices is an essential part of modern security systems. More specifically, there has been an increase of focus on ways to exploit covert communication channels to transfer data between computers that are seemingly "air-gapped". These approaches have mainly focused on the exfiltration of data from secure systems, and are limited by the slow and unreliable nature of alternative communication methods: light, sound, vibration, etc. One advantage of such communication channels is a lack of tracking mechanisms and standard protocols to analyze and detect attacks after the fact. For example, very few modern security systems for computer hardware include audio recording to determine if data is being transferred through sound, although network monitoring is very common. Previous works have demonstrated that ultrasonic communication across machines is possible, but a open source configurable package has not been released. We propose and release SPyAudio: a package that encodes and decodes audio signals using Hamming codes, and variable rate, number of signals, and frequencies. We demonstrate that different rates and signal densities are optimal in different scenarios, and therefore an configurable approach is best for achieving best performance. We release the code at https://github.com/arjunrawal4/spyaudio/.

KEYWORDS

security, covert channel, acoustics, signal processing, receivers, codecs

1 INTRODUCTION

Covert channels have been an area of increased focus for security analysis, as they represent relatively unexplored areas of potential threats. Audio channels and the transmission of data through sound have been studied from various angles such as steganography and NFC communication. Air-gapping, or the separation of machines with respect to network and data links, is commonly considered to be the best practice when trying to isolate systems. This technique is used to prevent external traffic or data from entering a private system. However, the idea that data cannot be transmitted without the use of traditional bands such as Wi-Fi, Ethernet, Bluetooth, USB, etc. has been shown to be fundamentally untrue, as covert channels transmitting data through light, sound, and other media has been shown to be effective.

1.1 Phone to Computer Channel

We study the specific use of an audio channel to convey covert messages from a source to a receiver. Analysis of covert audio channels has often focused on transmitting data from computer to computer, but here we transmit data from a phone to a computer, reducing the conspicuousness of having a secondary computer open next to an air-gapped system. Since phones are ubiquitous, and are often carried on a person, having easy access to a phone is an extremely low barrier for the attack. Although some systems may not have microphones, or disable sound collection for security purposes, the vast majority of devices allow for both playing and recording sound.

This paper makes the following contributions to covert audio channels and their analysis.

- We present the attack model in detail, and show that an attack using a mobile phone is harder to detect and easier to implement.
- We develop and release a framework for data transmission through audio with configurable parameters to allow adaptation to a wide variety of environments and objectives.
- We evaluate the framework in different scenarios to show that it is both effective and configuration enables better transmission.

The rest of the paper is organized as follows. Section 2 discusses related work in the field of audio channels and covert side channels. Section 3 introduces our signal processing pipeline and discusses the challenges that transmitting binary data through sound poses. Section 4 documents the systems on which our evaluation was conducted. In section 5 we present our configurable framework, and discuss the design choices that enable easy user configuration. Section 6 presents the experiments we ran and their results. We conclude in Section 7, and provide some discussion of potential countermeasures and future work in the area.

2 RELATED WORK

This work builds on previous work by Guri et al. [3] that uses sound to transmit data across a covert audio channel, such as from desktop computers that only have no network connection. Although this study provided one of the first major publication discussing the use of high pitched audio as a covert channel, their analysis of the effect of tone density and data transmission speed was minimal. Furthermore, their work did not produce any open source artifacts.

Additional work by Novak et al. [4] focused mainly on transmitting data using sound over very close devices. Although they were able to achieve a high rate of transfer, 4.9 kB/sec, they did not attempt to transmit data across a distance, an essential component of a covert audio channel. Additionally, they only claim that their work is nearly inaudible, an assumption that will not work when the purpose is a covert channel, not simply just simple communication.

Won et al. [7] worked on a similar problem of an audio covert channel, and looked at how the room configuration and background noise affected their signal processing and data transfer speed. Their analysis determined optimal devices and placements for the sound transfer, but did not evaluate the effect on data transfer and reliability that varying number of tones per second, and at a time, can provide.

Finally, work by Shwartz and Birk [6] demonstrates the capabilities of a covert audio channel between a virutal machine and a host, bypassing sandboxing. This overcomes the traditional audio challenge of unreliable communication as the sender and receiver are on the same machine. Although this attack may work in some cases, and is much faster at transmitting data, it is limited in scope. Furthermore, this approach required the use of expensive audio monitors to achieve good performance, an assumption that we do not require.

3 SIGNAL PROCESSING DESIGN

The main challenge of an audio transmission challenge is the unreliability and noise associated with an audio signals sent through open air. To combat human detection, we require that all sounds are above the commonly accepted threshold for adult (above age 21) human hearing, 18000 Hz. It is possible that some humans can detect sounds above 18000 Hz, but at the decibels used for our transmission this is unlikely, and has been verified with individuals age ranging from 21 to 35 (n=10). At these frequencies, audio processing on general purpose devices is fairly unreliable, as the hardware is not designed for high fidelity processing at ultrasonic frequencies.

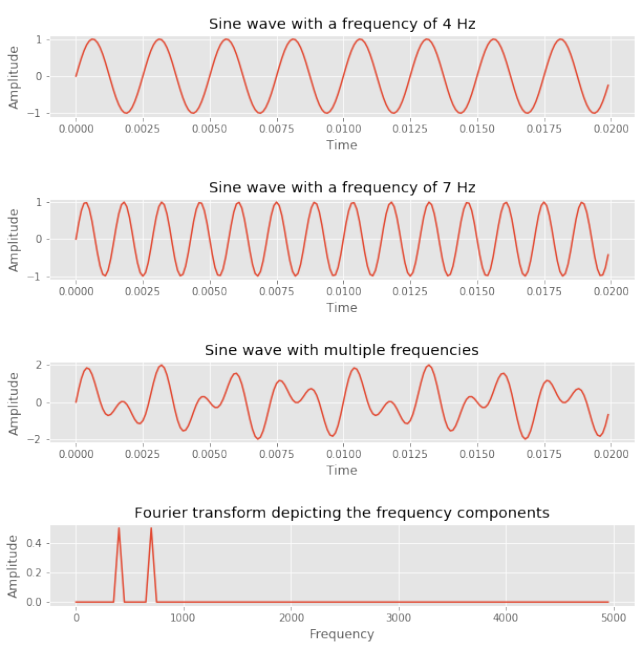

Figure 1: Demonstration of Signal Encoding and Decoding

3.1 Fourier Transformation

We utilize a Fast Fourier Transform to analyze the incoming sound data for features which we then transform to binary signal which can be converted to plaintext data. When encoding data, we layer signals at different frequencies on top of each other to increase throughput. We see that using an FFT we can decode the presence of individual signals at certain frequencies as shown in Figure 1.The Fourier Transform is used to decompose a sample of a signal over time to a set of frequencies and their relative strengths. Simply put, this allows us to take a small sample of an audio signal and determine which frequencies are being played. This is highly noisy process, as the signals are inexact and transfer over air distorts a portion of them. When transfer is done only through high fidelity audio, such as 3.5 mm audio cables, the signal accuracy approaches 100%. The challenges with the Fourier transform is to properly align the start and stop of the signals to properly process each time interval. For example, if the sliding window used by the receive is off by 50%, it will receive half of the audio signal from each of the two sender windows it views, and produce a noisy and inaccurate response. We solved this problem by introducing a starting and ending tone, which are fully distinct from the tones used in sending actual data.

3.2 Phase Synchronization

We first record all audio input to the receiver over a time during which a message is sent. To synchronize the audio signal, we run a Fourier transform on the tone length from 10 intervals within each tone length. For example, with tone length 100, we process an FFT for interval $(0, 100), (10, 110), \ldots (100, 200)$. Once we find a subsection where the value of the target start frequency, $F_s > T_s$, where T_s is a threshold used to distinguish actual signal from noise, we advance to the end of the section and begin processing the signal. The subdivision by 10 means that at most we are 10% off from the correct window divisions, leading to higher fidelity with lower processing overhead. However, this is a value which can be adjusted for particular cases. During the standard signal processing pipeline for actual data, we process each chunk of time of the length of one tone, and then convert it to text using a process described later. We additionally test if the value of the target end frequency, $F_e > T_e$, where T_e is a threshold used to distinguish actual signal from noise, and finish processing the signal.

3.2.1 Threshold Selection. From our experimentation, the thresholds used to determine valid signals from noise are somewhat variable, and depend on the signal strength and environmental conditions. For this reason, we decided to use a parameter search over the possible values in our decode pipeline, and return the one that succeeds the best. Specifically, our code searches over all the frequencies that could have valid data transmitted, and determines if a signal is present there using the following metrics. We denote the signal strength at x with F_x, the strength threshold T_r, the step coefficient k, and difference coefficient D.

(1)

$$F_x > T_r$$

(2)

$$D * F_{x(1+k)} < F_x$$

(3)

$$D * F_{x(1-k)} < F_x$$

This checks a few different constraints to validate that a signal is being sent. First, it requires that there is some significant signal at that frequency. Then we adjust the value of the frequency by scaling it by 1 ± k. A constant shift produces difficulties when frequencies are close together, so a scale was chosen. We then multiply the strength at these two "boundary" signals by D, and ensure that they still remain lower than the frequency at F_x. This ensures that at some frequencies bordering the chosen frequency, the strength of signal is significantly lower than the signal at the chosen frequency. In practice, we search over values of D and T_r to find the best split which minimizes errors in the signal processing.

4 METHODOLOGY

We use an iPhone 7 to play sounds using a WAV format. The audio device uses a stereo speaker, and is capable of playing sound at up to 73.2 DB [1]. On the recieving side, we use a 2016 MacBook Pro. According to Apple [2], this device has "Studio-quality three-mic array with high signal-to-noise ratio and directional beamforming". Both of these devices are commonly used, demonstrating the ease of this attack on standard commercial hardware. The software is developed fully in Python 3.7.6, with PyAudio 0.2.11, numpy 1.18.1, scipy 1.2.1 and the default wav file. We use a sample rate of 44,100 Hz, a common sampling frequency for audio processing. The analysis was done in a private study room approximately 5 feet by 10 feet, with decent sound isolation.

5 SOFTWARE PACKAGE

We develop and release a configurable framework to enable covert audio channels in a variety of environments and objectives. We expose the following configurable parameters, sample length l, base frequency F_0, tones per sample N, and frequency interval s. We use a standard procedure using Big-Endian representation and utf-8 to convert text into a binary stream. This binary stream can then be divided into chunks of size N, and signal sent at frequency F_i represents a 1 at that value, and the absence a 0 at that value. We use biological mechanisms as an inspiration for our error correction. We use a fixed time offset to deal with deletion or insertion errors, as even if the signal for a time interval is blocked (via white noise or other environmental factors), values will still be decoded. This will possibly effect one text character, but will not cause cascading problems. Point changes, as in the flipping of a single bit during transmission are harder to detect and correct and lead to incorrect characters being printed in the final output.

We see that the audio signal is quite noisy, and furthermore missing bits can cause extreme complications when the data is then converted to text. To deal with the prevalence of random errors, we use an error correcting code. We implement the Hamming(7,4) code in numpy using matrix multiplication to add redundancy to the communication, increasing reliability. This procedure allows for one bit to be flipped and still maintain perfectly accurate final output. This approach is still not highly accurate, but we see that other less efficient approaches such as triple modular redundancy could be used if high data reliability is required. Other techniques include interleaving to prevent an environmental factor from fully obfuscating a section of the plaintext.

	r=0.02	r=0.04	r=0.08
n=2	0.1013	0.1142	0.1875
n=4	0.0854	0.0417	0.0042
n=8	0.399	0.1094	0.0586

Table 1: Accuracy of Data Transfer (1ft)

	r=0.02	r=0.04	r=0.08
n=2	0.1336	0.1207	0.0216
n=4	0.1521	0.0062	0.0375
n=8	0.1133	0.1191	0.1094

Table 2: Accuracy of Data Transfer (3ft)

	r=0.02	r=0.04	r=0.08
2	0.5065	1.0	1.0
4	0.2375	0.1896	0.0792
8	0.1816	0.1875	0.1699

Table 3: Accuracy of Data Transfer (9ft)

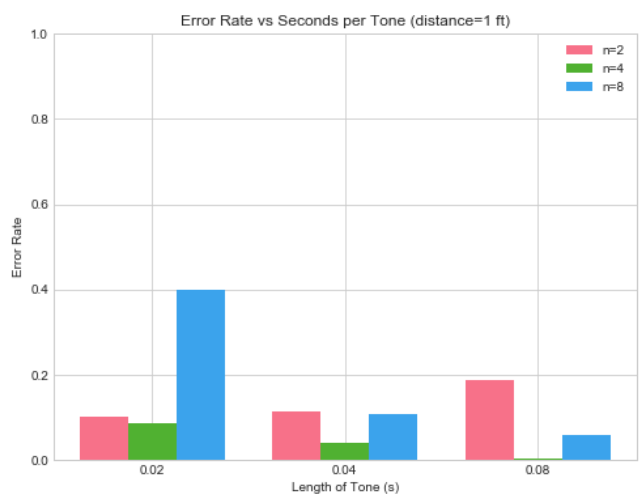

Figure 2: Accuracy of Data Transfer (1ft)

6 EVALUATION

We evaluate the effectiveness of various configurations of the following parameters: sample length l, base frequency F_0, tones per sample N, and frequency interval s. During these experiments, we maintain a parameter search over the set of possible combinations of T_i, the threshold coefficients, and D, the difference coefficient. We consider the maximum value for transmission accuracy over the parameter choices for T_i and D. We then run the following experiments.

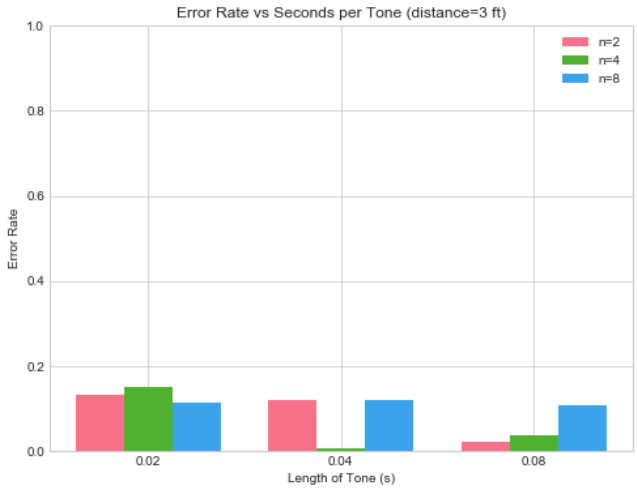

Figure 3: Accuracy of Data Transfer (3ft)

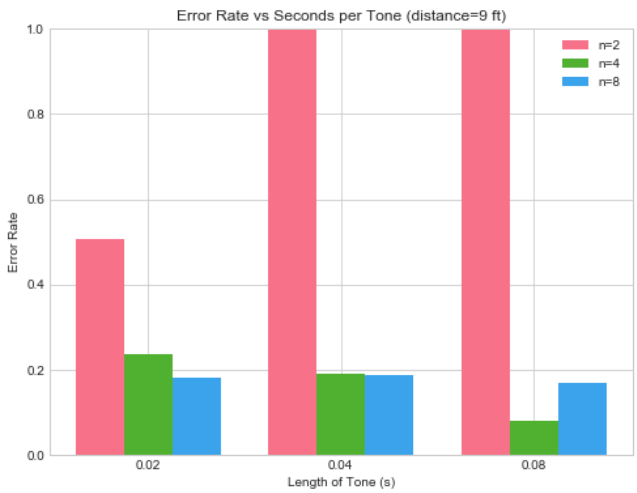

Figure 4: Accuracy of Data Transfer (9ft)

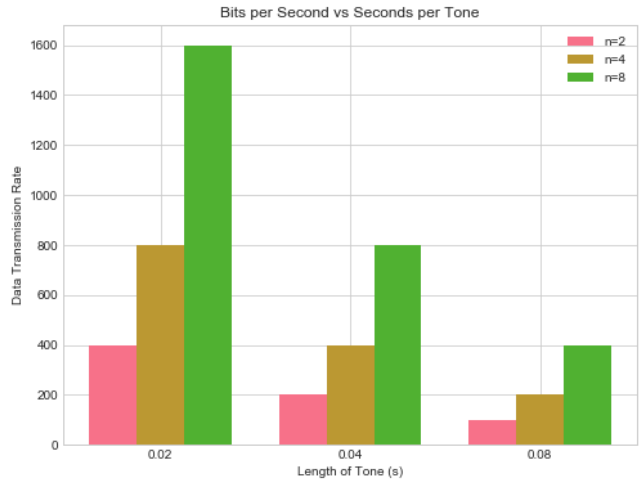

Figure 5: Maximum Data Transfer Rate

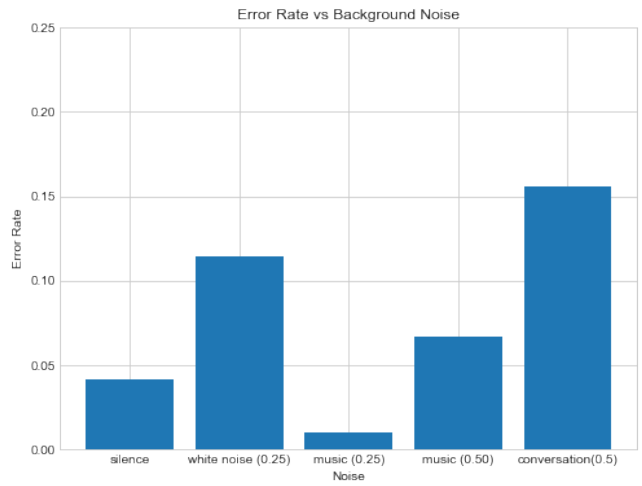

Figure 6: Accuracy of Data Transfer with Background Noise

(1) Effect of increasing rate of tones per second on throughput and accuracy (Silent background, fixed 3 ft distance)
(2) Effect of increasing number of tones played on once on throughput and accuracy (Silent background, fixed 3 ft distance)
(3) Effect of distance on accuracy (Silent background)
(4) Effect of background noise on accuracy
(5) Effect of error correcting codes on accuracy

To evaluate, we use the message "Oh, it's such a perfect day. I'm glad I spent it with you", and proceed using the experimental setup described in Section 4. We use 3 trials and take the best result, as the data transfer is quite noisy.

6.1 Rate of Tones

We first adjust the rate of tones per second, and analyze the effect that this has on both data transmission rate, and the accuracy of the data sent. As Figure 5 demonstrates, the shorter the tones (r), the faster the possible data transfer can occur. However, as Tables 1, 2, and 3 demonstrate, the effective error rate for shortening the tone duration is not very clear. Generally, the longer tones lead to greater accuracy, which follows intuition. Simply put, a longer tone enables a more accurate and less noisy signal, which can then be decoded more effectively. However, due to general variance, this trend is also somewhat noisy.

6.2 Tones Per Interval

We also compare the effect of increasing the number of tones being used in parallel, and its effect on data transfer and accuracy of the data sent. We see that similarly to rate, the more tones per interval (n), the faster the possible data transfer can occur. Figure 5 demonstrates this trend. Additionally, for data accuracy, we see that the 4 tone system seems to perform the best overall, but there

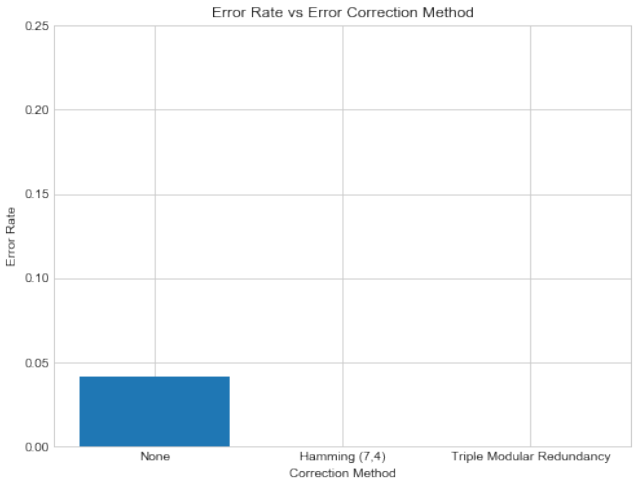

Figure 7: Accuracy of Data Transfer with Error Correction

are some variances in different cases. It would seem that adding additional tones would cause a decrease in accuracy as they could interfere with each other and cause signal distortion, but it appears that there is a "sweet spot" where this distortion is minimized. One possible explanation for the higher error rate with only two tones is the length of transmission is increased, and so air patterns and other environmental factors can cause a larger effect on a longer duration.

6.3 Distance

We analyze the effectiveness of this communication method over different distances. We compare 1 foot, 3 feet, and 9 feet. As Figures 2, 3, and 4 demonstrate, the accuracy generally decreases as distance increases. This follows our general intuition about sound, as the distribution of the sound waves decreases at a quadratic rate, and therefore accuracy will diminish rapidly. However, we see that by choosing the right parameters at a large distance, we can still maintain an error rate below 0.2. This demonstrates that correct configuration is essential to providing a reliable data transmission, as other configurations have an effect data transfer of 0. We attempted farther distances, and found that data transfer was fully impractical after around 20 feet, although with specialized audio equipment, this distance might be much further.

6.4 Background Noise

We evaluate the capabilities of data transfer in the face of external noise. Due to equipment constraints, we were unable to use sound detection equipment to accurately gauge the volume of background noise, so all values listed are relative to the maximum volume on a secondary iPhone 7. We used the secondary phone to generate audio signals aligned to be orthogonal to the transmission path. In spite of the experimental obstacles, we still demonstrate the the effect of background noise is not insurmountable. We experiment using an mid range configuration from the earlier experiments, $d = 1, n = 4, r = 0.04$. We see that music (Space Song - Beach House) provides minimal interference with the signal, at both low and high volumes. It appears that the lower error rate with 0.25 music is an anomaly. However, white noise and conversation both provide somewhat effective barriers to data transfer, although the underlying error rate increase (3-5x) still allows data transfer and can be assisted by the use of error correcting codes.

6.5 Error Correcting Codes

Error correcting codes have an effect on both the data transfer rate and the reliability of the data stream. We experiment using an mid range configuration from the earlier experiments, $d = 1, n = 4, r = 0.04$. As showin in Figure 7, Hamming (7,4) and Triple Modular Redundancy provide great increases in data reliability. In other experiments, Triple Modular Redundancy proved to be more effective than Hamming codes, as it can tolerate 1 error for every 3 bits whereas Hamming is only able to correct 1 error for every 7 bits. Both methods slow down effective data transfer drastically, with Hamming decreasing transfer rate by 43% and Triple Modular Redundancy cutting transfer by 66%. However, the benefit of a configurable and adaptive construction is that these codes can be turned on and off to deal with more and less reliable situations. Furthermore, when error rates are very high, more drastic correction such as 9x modular redundancy can be used if transfer rate is less important.

6.6 Evaluation Summary

Overall, we see that covert audio data transfer is a practical attack on systems. Furthermore, we demonstrate that a variety of parameters, length of tones, tones per interval, and error correction can be used to adapt to different distances, background noise, and environmental factors. Further analysis to show the effect of varying frequency intervals and starting frequencies could also be done with likely similar results. Our experiments demonstrate that background noise may not be an effective countermeasure, unless it reaches very high volumes. However, preventing close access to a machine would stop data transfers, as we were unable to maintain any transfer from more than 20 ft. Frequency limitation also dampens the possible data transfer rates and transfer reliability. Because we restrict ourselves to non-human hearable ranges, we lose a large part of the speaker range, and the most powerful range. iPhone speakers and MacBook Pro microphones are not designed to handle pitches outside of human hearing, and so their effective volume and capture is much lower than other tones. We did not formally study the transfer across all frequencies, but anecdotally, this would enable much higher values for n, potentially up to 64, increasing data transfer by more than 8x.

7 CONCLUSION

7.1 Possible Countermeasures

There are a number of possible countermeasures that systems can utilize to prevent these types of attacks. First, preventing other devices from being in the same room as the vulnerable system is becoming less and less effective, as devices become smaller and more capable. This attack, implemented on a phone, could likely also work from a watch or other small audio device. A better approach, therefore, is through the vulnerable device itself. Devices should not allow audio access whatsoever if proper security is to

be achieved. Attacks similar to this one have been shown to be possible even if a microphone is not available. Finally, a somewhat effective countermeasure is to have each device emit a form of sonic interference such as emitting white noise at a low volume.

7.2 Future Work

Although manual configuration is a good intermediate step for creating reliable sound channels, automated parameter configuration is the ultimate goal. Future work could apply machine learning to automatically configure transmission protocols based on the environment and adjust to changing sonic patterns or environmental interference. Other potential areas of future exploration include transmission from phone to phone, with the potential for covert and untraceable communication. Since the origin of a sound is hard to pinpoint, some form of secure voting could be implemented using this technique, as a receiver could tell that certain frequencies were present, just not exactly who they were coming from. There has also been work focusing on deep-learning based approaches to extracting data from noisy signals by Sajedian and Rho [5]. This work uses a neural network to deliver fast frequency predictions from a noisy signal, and could be used to provide a more accurate detection of signals.

7.3 Summary

We see that building an audio covert channel between commercially available devices is not only possible, but practical. We demonstrate data transfer from mobile phones to laptop computers using on board audio devices. Moreover, this technology can be configured to achieve good performance in a variety of situations.

REFERENCES

[1] 2016. Apple iPhone 7. https://www.gsmarena.com/apple_iphone_7-8064.php
[2] Apple. [n.d.]. MacBook Pro 16-inch - Technical Specifications. https://www.apple.com/macbook-pro-16/specs/
[3] M. Guri, Y. Solewicz, and Y. Elovici. 2018. MOSQUITO: Covert Ultrasonic Transmissions Between Two Air-Gapped Computers Using Speaker-to-Speaker Communication. (Dec 2018), 1–8. https://doi.org/10.1109/DESEC.2018.8625124
[4] Ed Novak, Zhuofan Tang, and Qun Li. 2018. Ultrasound proximity networking on smart mobile devices for IoT applications. *IEEE Internet of Things Journal* 6, 1 (2018), 399–409.
[5] Iman Sajedian and Junsuk Rho. 2019. Accurate and instant frequency estimation from noisy sinusoidal waves by deep learning. *Nano convergence* 6, 1 (2019), 1–5.
[6] O. Shwartz and Y. Birk. 2017. Sound Covert: A Fast and Silent Communication Channel through the Audio Buffer. In *2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*. 313–320. https://doi.org/10.1109/PDP.2017.13
[7] Kyoungpil Won, Sanggil Yeoum, Byungseok Kang, Moonseong Kim, Yeji Shin, and Hyunseung Choo. [n.d.]. Inaudible Transmission System with Selective Dual Frequencies Robust to Noisy Surroundings. ([n. d.]).