Acoustic Side Channel Attack

AKSHAY PALURI, University of Texas at Arlington
HYUN HO KIM, University of Texas at Arlington
KUNDAN SINGH MAHATO, University of Texas at Arlington

Acoustic Side channel attacks uses noise generated by our computers which could potentially expose them to attackers who want to access confidential information from our computers. In this report, we include the acoustic side channel of guessing the amount of words the victim typed by using the audio recorded from our phones. We will use several experiments from our data and generate spectrogram analysis through which we can detect any pattern to distinguish between normal key stroke and space bar, and by counting space bar, our code will guess how many words were typed.

CCS Concepts: • **Acoustic Side Channel attacks**; • **Side Channel analysis and countermeasures**;

Additional Key Words and Phrases: Acoustic Side Channel

ACM Reference Format:
Akshay Paluri, Hyun Ho Kim, and Kundan Singh Mahato. 2023. Acoustic Side Channel Attack. In . ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

As part of our ongoing effort to comprehend and reduce the dangers posed by physical side-channel assaults, we are focusing our research on learning more about one particular category: acoustic side-channel attacks. Acoustic side-channel attacks are distinct from traditional vectors in that they take use of the noise that electrical equipment produce while they are in use, rather than focusing on power usage, heat emissions, or electromagnetic signals. Specifically, our study focuses on keystroke analysis, a method where the distinctive sound patterns generated by typing on a PC become possible points of vulnerability. The complexities of keystroke analysis in the context of acoustic side-channel incidents require a thorough examination of the noises produced during typing operations on personal laptops or computers. This investigation is helped by the use of the gadgets' built-in microphones. Each keystroke produces a different sound signature, and a prospective attacker can identify information about the letters being typed by methodically examining these acoustic patterns. The growing usage of personal computers in numerous aspects of everyday life, ranging from professional work and communication to the storing of sensitive information, highlights the sensitivity of this avenue of attack.

As the amount of dependence on computers grows, becoming essential to activities such as online banking and professional communication, an acute grasp of the complexities connected with acoustic side-channel assaults becomes critical. The consequences of competitors listening in on keystrokes via sound recordings on home computers are considerable, posing a real danger to user privacy and data security. In this context, our research aims to unravelthe complexity of keystroke analysis via microphone recording, diving into the potential risks and consequences of this

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
Manuscript submitted to ACM

largely unknown aspect of acoustic side-channel crimes. Acoustic side-channel attacks have evolved as a powerful and creative method of collecting sensitive information from electronic equipment by using the unwanted acoustic emissions produced during their operation. These assaults constitute a paradigm change in cybersecurity since they bypass standard protections by targeting the physical features of devices rather than exploiting software flaws. Acoustic side channels originated in the early days of computers, but recent advances in signal processing technology and the increased sensitivity of microphones have propelled these attacks into a higher level of complexity.

An acoustic side-channel assault, at its heart, makes use of the noises produced by electrical components throughout their different computational processes. Once recorded and studied, these emanations can provide extensive insights about a device's internal activities. Attackers can obtain important information by interpreting keystrokes and retrieving cryptographic keys, which typical security methods fail to protect against. Acoustic side-channel attacks are versatile in that they may target a wide range of devices, from old computers to current smartphones and networked IoT gadgets. The threat presented by acoustic side-channel assaults has grown in importance as the cybersecurity landscape advances, forcing researchers and industry specialists to focus on the creation of solid defenses. Addressing this unique danger necessitates a multifaceted strategy that includes both hardware improvements and novel noise production techniques to mask sensitive auditory signals. Understanding the complexities of acoustic side channels is critical for developing effective security tactics in the continuous effort to safeguard information in a more linked and technologically sophisticated society.

2 BACKGROUND

Acoustic side-channel assaults are a sophisticated and subtle cyber threat that takes advantage of unwanted sound emanations created by electrical equipment to collect sensitive information. These attacks make use of the fact that electronic components emit acoustic signals while they function, and these emissions can accidentally transmit information about the device's internal activities. Because it operates fully beyond the sphere of regular cybersecurity protections, this concealed type of information extraction offers a huge challenge to established security procedures. Acoustic side-channel assaults have their roots in the early days of computers, when researchers realized that electrical components generate sound waves during normal operation. However, it wasn't until the introduction of advanced signal processing techniques and the rising complexity of microphones that attackers were able to fully exploit the possibilities of acoustic side channels. Acoustic side-channel assaults are distinguished by their ability to target a wide range of devices, including desktop computers and laptops, cellphones, and Internet of Things (IoT) devices.

One of the key techniques by which these assaults function is the capture and analysis of a target device's acoustic emanations produced throughout its different computing tasks. To discover patterns within the auditory waves, attackers use specialized equipment such as very sensitive microphones and complex signal processing techniques. These patterns may correlate to particular activities such as keystrokes, data processing, or cryptographic computations, allowing attackers to deduce sensitive data such as passwords, encryption keys, or other secret data. Acoustic side-channel attacks are effective because they may bypass typical security measures focusing on safeguarding data at rest or in transit. Unlike traditional assaults that target software or network weaknesses, acoustic side-channel attacks target the physical features of electrical devices. As a result, even air-gapped systems, which are traditionally thought to be more secure owing to their isolation from external networks, are vulnerable to this new type of infiltration.

Researchers and cybersecurity experts are hard at work creating solutions to the risk presented by acoustic side-channel assaults. These countermeasures range from hardware changes to reduce acoustic emissions to the creation of noise production methods that cover up sensitive acoustic signals. The arms race between attackers and defenders in

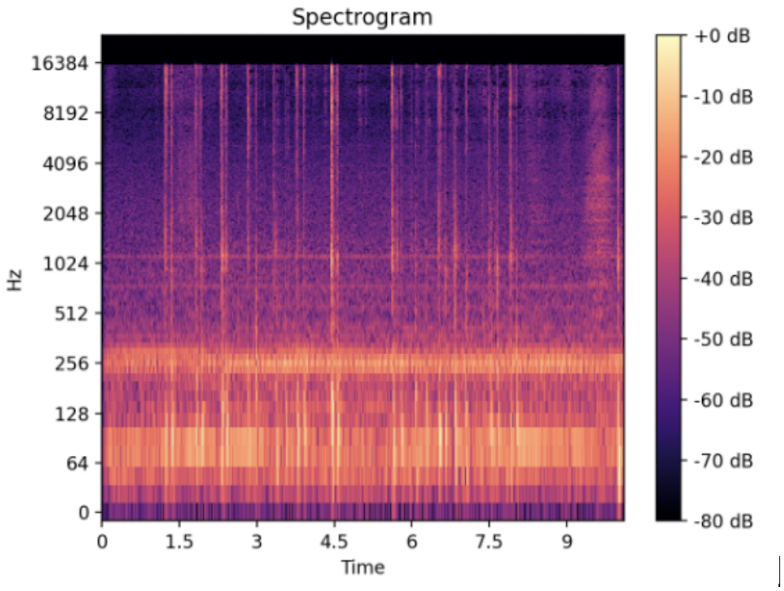

Fig. 1. Example of Spectrogram

the field of acoustic side-channel assaults promises to be an exciting and demanding frontier in the continuing war for information security as the threat environment evolves. Keystroke analysis, a subset of acoustic side-channel attacks, is understanding the sound patterns created while a person types on a keyboard. Each key press creates a distinct auditory signature, which attackers might use to deduce the actual keystrokes produced during typing sessions as signal processing capabilities progress. This poses a substantial risk to user privacy, particularly on devices such as personal iPhones, which frequently hold sensitive personal and financial data.

Because keyboards are so prevalent in our everyday lives, they are an appealing target for attackers looking to exploit user data. As people rely more and more on their laptops, PCs, and smartphones for activities such as online banking, shopping, and communication, the potential impact of successful acoustic side channel assaults on these devices cannot be overstated. The goal of this research project is to thoroughly investigate the feasibility and effects of doing a keystroke analysis using sound recordings from personal iPhones. We intend to provide useful insights to continuing efforts to improve the security of personal devices against emerging cyber threats by studying the subtleties of acoustic side-channel assaults in this unique scenario.

3 PROJECT DESCRIPTION

3.1 Purpose

In our project, 'Acoustic Side Channel Attacks: Keystroke Sound Analysis,' we aimed to count words by detecting 'space bar' keystrokes, which serve as indicators of word separators. Our approach involved analyzing keystroke sounds using an acoustic frequency table to pinpoint the distinct pattern of 'space bars.' For example, in a phrase like "this is a test," our system would identify three 'space bar' keystrokes, indicating a total word count of four. The focus on the space

bar was due to its larger size and its unique, hollower acoustic signature, which makes it stand out from other keys. Typically, the sound of a space bar keystroke is louder and lasts slightly longer than other keystrokes.

3.2 ASSUMPTION

In conducting our project, "Acoustic Side Channel Attacks: Keystroke Sound Analysis," we worked under several key assumptions crucial to our results' integrity and accuracy. One significant assumption was the type of keyboard used during data collection. We chose standard keyboards like the 'Logitech Keyboard K120' and 'MacBook Scissor switch keyboard,' which are commonly used in everyday scenarios. This choice ensured our findings would be relevant to the most used hardware, avoiding specialized keyboards such as mechanical, silicon, or laser keyboards. Another critical assumption related to our typing methodology was ensuring single key presses during data collection. This minimized the potential for overlapping acoustic signatures, which could complicate the analysis and possibly skew the results. Finally, all data collection was conducted in a quiet environment to mitigate external noise interference.

3.3 Project Process

Our project followed a series of steps, each important for our study of sound from keystrokes. We were careful and thorough, making sure we looked at every part of how keystrokes sound in detail.

The first step was capturing keystroke sounds using an iPhone microphone for its high sensitivity and wide availability. These sounds were then converted into .wav format for detailed analysis.

After initial data capture, we focused on refining the acoustic data quality. This involved using the AudioSegment library for audio normalization, enhancing recording clarity by reducing background noise and normalizing volume levels.

The next phase was digital signal processing, converting acoustic inputs into digital signals for more detailed analysis.

A key part of our methodology was developing a space bar detection algorithm. This algorithm identified 'space bar' keystrokes based on their louder volume and longer duration compared to other keys.

The final step was continuously analyzing the sound files, monitoring the entire duration of each recording to count keystrokes meeting the 'space bar' condition.

3.4 Project Outcomes and Implications

To achieve more accurate and realistic results in our project, we conducted testing with a variety of sound files and keyboards. By using various keyboard models and sound conditions, we were able to gather a broad dataset. This diversity in our test data helped us refine our system to be adaptable and accurate across various keyboard types and acoustic settings. In addition to testing different keyboards, we adjusted the threshold values and time durations within our analysis algorithm. These adjustments were necessary to accurately capture each keystroke sound. We also attempted to count individual letter sounds from typing. This part of our research was aimed at assessing the potential for data leakage through keystroke sounds. By understanding how individual letters could be identified acoustically, we gained insights into the security vulnerabilities that could arise from acoustic side channel attacks. The space bar typically has a distinct sound profile, often being louder and having a longer duration than other keys. Recognizing these characteristics was crucial in distinguishing the space bar keystrokes from others. This analysis not only aided in word count accuracy but also added a layer of depth to our understanding of keyboard acoustic dynamics. This comprehensive approach enabled us to successfully count words in each audio segment.

```
# keystroke file
loud_then_quiet = AudioSegment.from_file("Test48.wav")
normalize_then_quiet = loud_then_quiet.normalize()
normalize_then_quiet.export("NormalTest48.wav", format="wav")
# Load the keystroke file
audio_data, sample_rate = librosa.load("NormalTest48.wav", sr=None)

# Generating the spectrogram
hop_length = 512
n_fft = 2048  # Adjust the number of FFT points
fmax = 10000   # Maximum frequency value

spectrogram = librosa.feature.melspectrogram(y=audio_data, sr=sample_rate, n_mels=128, n_fft=n_fft,
                                             hop_length=hop_length, fmax=fmax)
log_spectrogram = librosa.power_to_db(spectrogram, ref=np.max)
```

Fig. 2. Code to load and process audio file

4 EXPERIMENTS AND RESULTS

In order to identify a keystroke pattern, we first needed to be able to record a key stroke and then translate from audio file to be able to process it in code. We first took sample recordings from our phone and then used those audio files into our code. We use pydub module from python, and imported AudioSegment function which works with different audio files, and then we use it to normalize the sound of the audio file. We used the normalize function located in the object returned by AudioSegment from opening the audio file. In order to generate the specrtrogram graph, we imported the libarosa and matplotlib modules, which are going to be key to giving it some key parameters to determine the plot of the graph. This can be seen in figure 2, where "`Test48.wav`" is the name of the audio which has the audio recordings of the keystrokes. The spectrogram variable takes in the audio data, and the sample rate features of the normalized audio file, and then creates a spectrogram. Then, the log spectrogram variable used the libarosa's power to decibel conversion with the spectrogram variable to generate the spectrogram analysis graph, with the needed values of decibels, frequency, and time.

The next needed is to generate the spectrogram analysis graph, and identify an patterns that exist that helps in identifying a difference between a normal keystroke and a space bar. In order to start, we will just input numbers from macbook keyboard, to see if we can identify any patterns. In figure 3, we notice that there are several lines, with different audio patterns, that indicate a keystroke. We notice that they have a much higher dicibel values, and as we approach a much higher frequency, the sound values are much more cleaner with less background noise. As per the assumptions we make, we do expect the keyboard to make a much higher and more consistent decibel sound compared to a normal keystroke. Since every keystroke has a press and release event, we can also see that every single keystroke event has two very close lines to each other. Hence, our algorithm should also make sure that these time differences are also accounted for when detecting a keystroke. We should also notice that the sound or decibel values generated in the graph are also dependent on how the keystrokes were typed. If the keystrokes were typed in a lower sound, or typed much faster, the values that would be produced by the spectrogram analysis would be much more lower or higher in decibel values, and would be harder to identify a pattern.

In figure 4, an audio file with one space bar at 4.5 seconds and the difference between that sound is noticeable. But also the width of the time event is much more than a normal keystroke at any other time. With higher decibel values, and much more consistency across several frequencies, we can detect a space bar event. We now require to print the values of the spectrogram analysis to identify patterns and group the patterns to detect space bar.

In order to print the necessary values, we need to first determine the minimum decibel value we need to use to print since we don't want to capture any quiet sound, or sounds where there weren't any key strokes. If we use the spectrogram analysis from figure 4, we shall can use -20dB sound as a threshold, and shall only generate sounds that

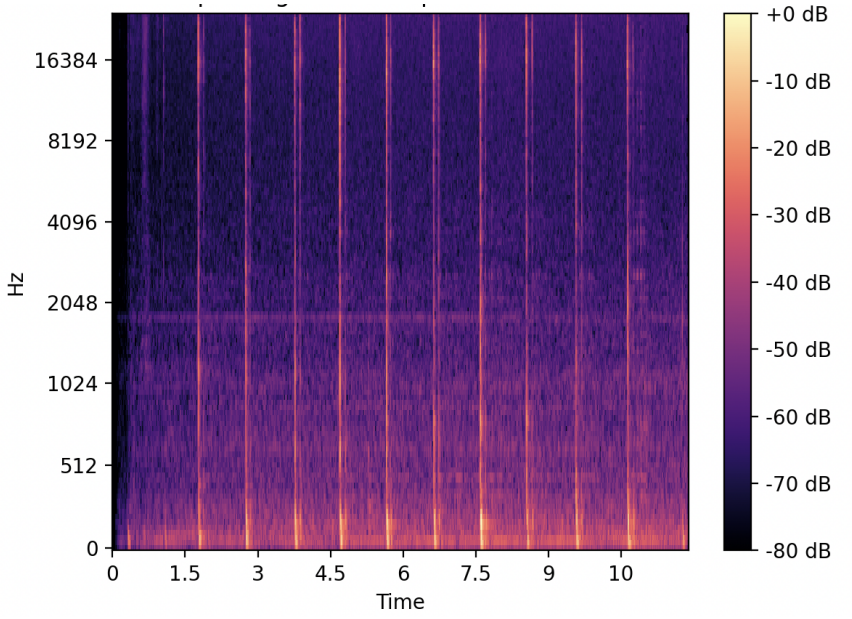

Fig. 3. Spectrogram with numbers as input

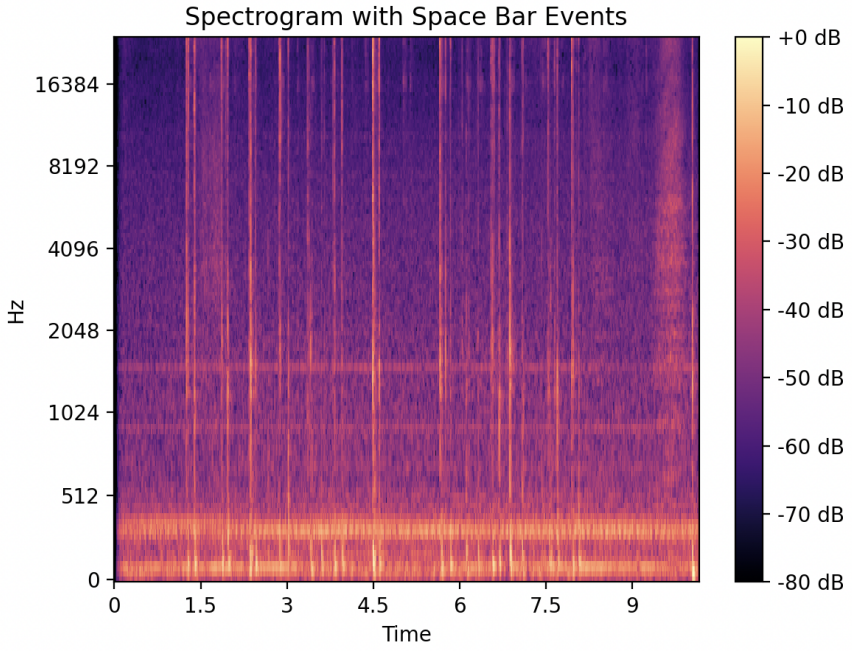

Fig. 4. Keystroke at 4.5 seconds

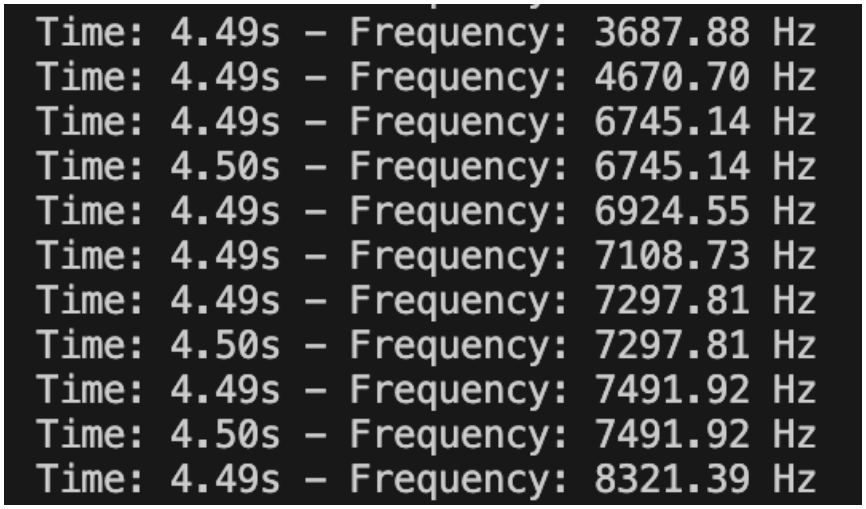

Fig. 5. Time and frequency values more than -20dB

have higher values. But, we can notice that the sounds at lower frequencies, are producing a much higher sound although it may not be a keystroke and could be a background noise. So, we are not going to consider frequencies lower than 2000Hz. This will give us frequency and time values that are going to be more consistent and have much higher chances of being a keystroke.

In figure 5, the value shows the time and frequency values more than 2000Hz that were caught which were more than -20dB. We can also see that the values are more concentrated around 4.5 seconds. This also indicates that there is a keystroke at that area. To detect how many space bars we encountered, we need to first identify how often these time events has occurred so that if there too few, then we can consider them as a regular keystroke rather than a space bar. In order to group these values, we are going to use a time array, which will hold in all the time values that were printed, and will identify the frequently occurring time values. Once we store them, we need to have a count on how many times a certain time values should occur to make them count as a space bar rather than a regular key stroke. After doing several experimentation, we identified that the best option would be to use 15 occurrences of a particular time slot for that to be identified as a space bar event. Once we group these time slots, and count them, we can identify how many space bar events have occurred throughout the audio file. We are also going to sort our time array in increasing length so that we can keep resetting our count back to 1 if we see a new time value. The code in figure 6 shows how many times certain time value has occurred and if it sees more than 15 of them and within the range of 0.1 seconds, it will count them as one space bar event. In this specific example, it correctly detected a space bar, and indicated that 2 words were typed in the given audio file.

5 CONCLUSION

As noted in the introduction, acoustic side channel attacks are extremely dangerous which can cause confidentiality breaches. Noises are constantly generated by our keyboards and they give away much of the information that could be sensitive. The key stroke noises could also potentially reveal how many words were typed the victim and could shorten the amount of guesses the attacker can make with the word they typed. Also, the microphone used from our

```
def valExists(num, valArray):
    for values in valArray:
        if num < values + .10 and num > values - .10:
            return True
    return False
for i in range(len(time_ind)):
    print(time_ind[i])
    if(time_num < time_ind[i] + 0.10 and time_num > time_ind[i] - 0.10):
        time_count += 1
        if(time_count >= 15 and not valExists(time_ind[i],visited_time)):
            num_spaceBar += 1
            visited_time.append(time_ind[i])
    else:
        time_count = 1
        time_num = time_ind[i]
```

Fig. 6. Algorithm to process time values

experiment was a recorder from Iphone and there are potentially much better microphones that could be available that can capture every key stroke with great accuracy. There are multiple mitigation strategies that could be used to prevent such instances from happening. If there is a white noise across the background, it makes it harder for the attacker to be able to detect any noise for key stroke. Also, if there are any sensitive information, the person should work in an environment that is isolated and more familiar to them so that they are sure that there no acoustic attacks. Another mitigation strategy is to be able to detect any acoustic attack in your surroundings as these attacks require close proximity to the victim, and checking any microphones or any recording device that could be places near the victim.