Keyboard Emanations in Remote Voice Calls: Password Leakage and Noise(less) Masking Defenses

S Abhishek Anand
University of Alabama at Birmingham
anandab@uab.edu

Nitesh Saxena
University of Alabama at Birmingham
saxena@uab.edu

ABSTRACT

Keyboard acoustic side channel attacks to date have been mostly studied in the context of an adversary eavesdropping on keystrokes by placing a listening device near the intended victim creating a *local* eavesdropping scenario. However, being in close physical proximity of the victim significantly limits the applicability of the attack.

In this paper, we study the keyboard acoustic side channel attacks in *remote* attack settings and propose countermeasures in such attack settings. Specifically, we introduce an *offense-defense* system that: (1) highlights the threat of a *remote* adversary eavesdropping on keystrokes while the victim is on a VoIP call, and (2) builds a way to *mask* the leakage through the use of system-generated sounds. On the offensive side, we show the feasibility of existing acoustic side channel attacks adapted to a remote eavesdropper setting against sensitive input such as random passwords, PINs etc. On the defensive side, we demonstrate a software-based approach towards masking the keystroke emanations as a defense mechanism against such attacks and evaluate its effectiveness. In particular, we study the use of white noise and *fake keystrokes* as masking sounds and show the latter to be an effective means to cloak such side channel attacks. Finally, we discuss a novel way of masking by virtually inserting the masking signal in remote voice calls without distracting the user.

ACM Reference Format:
S Abhishek Anand and Nitesh Saxena. 2018. Keyboard Emanations in Remote Voice Calls: Password Leakage and Noise(less) Masking Defenses. In *CODASPY '18: Eighth ACM Conference on Data and Application Security and Privacy, March 19–21, 2018, Tempe, AZ, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3176258.3176341

1 INTRODUCTION

Acoustic side channel attacks have been shown to be successful at decoding keystrokes by exploiting keystroke emanations from the victim's typing, by placing a covert listening device in vicinity of the victim. [1]. The fundamental insight of these attacks is that, due to the mechanical characteristics of the keyboard, each key on the keyboard produces a unique sound on press and release. Studies have indicated that such attacks are reasonably accurate at decoding typed words by using methods described in [5–7, 11, 16]. As such, they pose a viable threat to highly sensitive information such as (random or non-random) passwords and other natural language text input by the user.

Given the huge concern about privacy issues in the digital world, these threats pose a serious risk to a user's interactions in an insecure environment. However, the existing acoustic emanation attacks have all been studied only in a *local setting* where the victim is being eavesdropped by a covert listening device placed in its vicinity. Such a setting may limit the applicability of the attack since the attacker is forced to be in close physical proximity of the victim user. The proximity-based attacks may also be detected/prevented via physical means, for example, with closer inspection of the space around the user such as through bug sweeping [3]. It is also important to note that, in such proximity settings, there may be other, simpler and more effective mechanisms to learn sensitive information from the user, such as simple shoulder-surfing attacks that monitor the user as she types her confidential information.

In this paper, our goal is to expand the research on keyboard acoustic emanations by exploring other attack avenues for the adversary and investigating defensive scenarios, especially when the victim is typing sensitive information such as passwords and PINs. First, since acoustic emanation attacks have already been studied extensively in a local setting where the victim is being eavesdropped by a covert listening device placed in its vicinity, we investigate the behavior of such attacks in a more broader *remote eavesdropper* setting. In this setting, the victim is unknowingly being eavesdropped by an attacker from a remote location while the victim is on a call with a malicious or compromised entity, or by a wiretapper who is sniffing the VoIP traffic containing the keystrokes between two honest parties. We also concentrate on attacks against random passwords and PINs as they demand more security and are supposed to be harder to decode due to their randomness. When compared to the localized attack setting, the remote attacks give immense flexibility to the attacker and are harder to detect, and therefore can be more devastating in practice.

Second, while the study of acoustic side channel attacks have been extensively highlighted, there is a scarcity of adequate defense techniques that can thwart such attacks. Most of the proposed defense revolves around improved hardware and a sanitized environment to suppress the acoustic emanations. However, these methods come with an expensive implementation making them impractical in assisting a common user, especially in a remote eavesdropping setting. In the face of such challenges, we examined the efficiency of system generated sounds in suppressing or masking the acoustic emanations to make the proposed remote acoustic side channel attacks ineffective. We investigate two types of defense

[1]We use the terms "victim" and "user" interchangeably throughout this work.

CODASPY '18, March 19–21, 2018, Tempe, AZ, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5632-9/18/03…$15.00
https://doi.org/10.1145/3176258.3176341

mechanisms: noisy defense mechanism that generates masking signal at the victim's end using speakers and noiseless defense that virtually mixes the masking signal with the victim's microphone output and directs it seamlessly to the remote calling application. While noiseless defense is user transparent, noisy defense can be utilized to counteract the threat of a local eavesdropping adversary (in addition to a remote attacker).

Our Contributions: In this paper, we show that keyboard acoustic eavesdropping is possible in remote voice calling applications like VoIP scenarios, where the attacker is eavesdropping while the victim is typing during the call. Moreover, we show that such attacks can be thwarted with acoustic masking mechanisms. We also introduce the idea of noiseless masking where the masking mechanism is used without distracting the victim from the important task of password entry. Our contributions are summarized below:

- **Leakage of Password Keystrokes:** We utilize similar attack methodology as in [8] and show in Section 5 that traditional attack vectors (like *time-frequency distance* and machine learning tools) provide a reasonable accuracy at decoding single keystrokes (74.33% for lowercase alphabet keys and 77.33% for numeric keys), 6-character length random passwords (46.67%), and 4-digit PIN (70.00%) using the keystroke sounds transmitted over VoIP calls.
- **Noisy Masking Defense:** We propose a noisy defense mechanism based on sound masking in Section 6 to hide the keystroke acoustic emanations during password (and PIN) entry. We study the effectiveness of white noise and *fake keystrokes* in hiding the keystrokes typed by the victim.
- **Noiseless Masking Defense:** We introduce the idea of a novel defense mechanism in Section 7 that silently injects masking signal into the audio stream of a voice call during password (or PIN) entry. This setup allows a noise free environment at victim's end while preventing an eavesdropping attacker at the other end from exploiting leaked password keystroke emanations.

2 BACKGROUND AND RELATED WORK

Acoustic eavesdropping over keystrokes has been a well researched area in computer security. The acoustic emanations resulting from the keystrokes were identified as leaking significant amount of information about the keys being pressed. Asonov et al.[5] used neural networks with labeled keystroke samples to identify a keystroke using Fast Fourier Transformation (*FFT*) as a feature from the key press regions. They were able to identify the keystrokes with an accuracy of 80%. Zhuang et al.[16] improved upon this work by using non labeled training samples and *Cepstrum* features to achieve accuracy up to 96%.

Berger et al.[7] exploited the correlation between keystrokes based on their physical location on the keyboard to decode single words of length 7-13 using dictionary attack. Halevi et al.[11] studied keyboard acoustic emanations attacks by using *time-frequency decoding* that provided improved accuracy over previous methods. They also explored the effect of a user's typing style on keystroke identification in the context of decoding random passwords. They inferred that different typing styles affect the accuracy of keyboard acoustic attacks and random passwords are less vulnerable to such attacks as language modeling techniques and dictionary based attacks are not applicable. Context free attack was proposed by Zhu et al. [14] that utilized multiple microphones placed strategically around a keyboard to identify keystrokes based on time difference of arrival (TDoA) of the keystroke emanation at each microphone.

Compagno et al. [8] explored the issue of a remote adversary eavesdropping on keystrokes through VoIP calls. They used *MFCC* features for keystroke detection and investigated the impact of bandwidth as well as presence of speech on the accuracy of their classifier. Our work re-investigates this issue with a focus on random passwords and PINs while proposing a defense mechanism against a remote eavesdropping adversary. Similar to [8], our work does not consider geometry based attacks such as [14] because remote eavesdropping removes the distance factor between the eavesdropping device and the keyboard. This renders these class of attacks infeasible in our threat model as distance i.e. TDoA can not be used as a unique identifying factor.

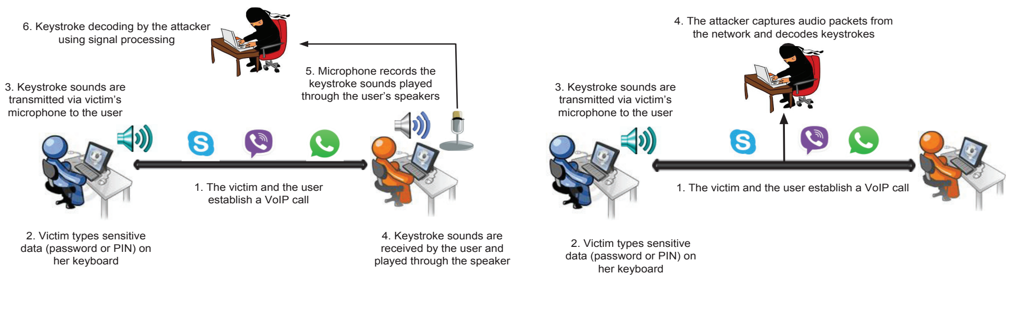
3 ATTACK MODEL AND OVERVIEW

In this section, we give an overview of the acoustic eavesdropping attack system, which is a reincarnation of existing localized keyboard acoustic attacks [5, 7, 11, 16] for remote voice call setting. We describe in detail our threat model for the voice call scenario elaborating the capabilities of the attacker.

In our threat model, we consider the victim to be in a voice call while typing sensitive information such as random passwords, PINs, or credit card information. We assume the user at the other end of the call is using speakers to listen to the victim. This allows the attacker to remotely eavesdrop on the voice call. In this scenario, the attacker could be a malicious application exploiting the microphone on the other end of the victim's call or it could be a dishonest user itself communicating with the victim. The attacker could also install a hidden listening device near the benign user (using speakers) who is on call with the victim. The voice call in our model could be a VoIP call, a video call or a normal phone call. In this work, we will limit ourselves to the VoIP scenario that can easily be extended to any types of calls with the victim.

Our threat model assumes that the victim is typing sensitive information during the call. One way to ensure this scenario could be using social engineering trickeries to influence the victim into logging to a website or authentication system while on the voice call. For example, in a customer service call, the customer service agent may ask the user to login to test if his account is accessible. The victim may also log into an authentication service even without the attacker's prompt while on the call.

To obtain training samples for the victim's keyboard, we assume the attacker has access to prior calls from the victim that allows building a keystroke feature set to be used for keystroke identification. Other ways for the attacker to build a training model for the victim's keyboard involve an exchange of chat messages between the attacker and the victim, online editing of a shared document (e.g. Google Docs) or coercing the victim to send an email while on call. In all of these scenarios, the attacker can record victim's keystrokes through VoIP call and match them against the written text. This is in contrast to prior keyboard emanations attacks [5, 7, 11, 16] where the attacker had physical access to victim's keyboard. As an analogy, this constitutes a type of known-plaintext attack against


6. Keystroke decoding by the attacker using signal processing
5. Microphone records the keystroke sounds played through the user's speakers
3. Keystroke sounds are transmitted via victim's microphone to the user
1. The victim and the user establish a VoIP call
2. Victim types sensitive data (password or PIN) on her keyboard
4. Keystroke sounds are received by the user and played through the speaker
4. The attacker captures audio packets from the network and decodes keystrokes
3. Keystroke sounds are transmitted via victim's microphone to the user
1. The victim and the user establish a VoIP call
2. Victim types sensitive data (password or PIN) on her keyboard

(a) Malicious end point attacker **(b) Network sniffing attacker**

Figure 1: Eavesdropping attack scenarios

encryption systems where the attacker has the general knowledge of the keys on the victim's device but not the actual typed keys.

We also consider a scenario where the attacker builds the training model by eavesdropping locally on the victim's keystrokes and then using the obtained training model to recognize the remotely eavesdropped keystrokes of the same victim. This scenario assumes that attacker may have one-time access to victim's keyboard (similar to lunch-time attack) that allows him to build a training model using the exact keyboard of the victim. This training model is later used to decode victim's keystroke over VoIP call.

In our model, we limit ourselves to eavesdropping on random passwords and numerical data (like PINs, credit card number, and date of birth). Hence, we eliminate the effect of individual typing style from the recorded keystrokes as such data is generally not entered using touch typing[2]. Also, HMM (Hidden Markov Model) language-based models and dictionary attack can not be useful in this case as random passwords and numbers are devoid of language features that make such attacks possible. To give the attacker a realistic chance for decoding the keystrokes, we allow multiple samples of the sensitive data to be collected by the attacker (e.g., the same password typed multiple times).

In line with prior acoustic side channel attacks, we design our environment to be free of noise to evaluate the maximum likelihood of a successful attack. While it is realistic for the victim to converse during the call while entering the sensitive information, we do not consider this scenario in our threat model. Note that the attacker side noise/speech does not affect our attacks, only the victim's side is important. Another important scenario in our threat model involves the attacker sniffing on the network traffic of the VoIP call between two honest parties. When a VoIP call is made to or from the victim, the attacker monitors the network traffic for VoIP packets. One amendable setting for this attack involves unencrypted VoIP communication, or third-party encryption (who themselves can be the attacker). However, even if VoIP services employ end-to-end encryption, VoIP packets have been shown to be vulnerable and can still be decoded as shown by Wright et al.[13]. Thus, our work may apply to both encrypted and cleartext VoIP communications.

4 ATTACK SETTINGS & TECHNIQUES

In this section, we lay out the design of the experiment that was conducted to study the feasibility of the attacker eavesdropping remotely on an unaware victim during a voice call while she enters some sensitive information (password or PIN).

4.1 Experiment Design

In our experiments, we study the following scenarios based on the threat model: a) An attacker (a malicious user) communicating with the victim, or a compromised machine on call with the victim, b) an attacker sniffing and capturing the audio packets directly from the network, and c) the attacker obtains a training model of the keystrokes locally and later eavesdrops remotely intending to use the training model for decoding. The first scenario is depicted in Figure 1a and the second scenario is depicted in Figure 1b.

4.2 Keystroke Processing

The keystroke processing done by the attacker for audio signals eavesdropped over voice calls is in line with keystroke extraction and processing done in previous works [5, 7, 11, 16]. The process is divided in two phases: keystroke extraction, and keystroke classification and recognition. We restrict our signal processing (that uses Fast Fourier transform) to the frequency band 400Hz-12kHz as we obtained best results in this band. For classification and recognition, we compared single character detection accuracy for existing methods in literature such as MFCC with neural networks [16], cross-correlation [7], frequency distance measure [7], frequency-time distance measure [11] and machine learning methods described in [8], and chose the method that gave us the best accuracy.

Our threat model differs from previous works in the sense that the eavesdropping is done remotely and not in the proximity of the victim, the eavesdropped signal is *not* the same signal that emanated from the keystrokes at the victim's end. The eavesdropped keystroke signal is a result of an encoding-decoding (defined by the underlying VoIP protocol) of the original keystroke emanation from the keyboard that is transmitted over the network. Therefore we implement and test all the previously described methods and choose the method that provides us with the best accuracy.

5 ATTACK EXPERIMENT AND RESULTS

This section describes the setup details of our experiments and the results that show the threat of an acoustic eavesdropping attack.

Table 1: Single key classification using time-domain and frequency-domain distance estimates

Classification Method	Accuracy (in %)
Alphabet keys (a-z)	
Cross-correlation	56.00
Frequency Distance	64.70
Frequency Time	**67.30**
Numpad keys (0-9)	
Cross-correlation	73.30
Frequency Distance	70.00
Frequency Time	**83.30**

Table 2: Single key classification using FFT coefficients

Classification Algorithm	Accuracy (in %)
Alphabet keys (a-z)	
J48	15.79
Random Forest	32.35
Linear Nearest Neighbor Search	20.92
SMO	21.95
Simple Logistic Regression	**39.54**
Multinomial Logistic Regression	**38.89**
Numpad keys (0-9)	
J48	39.67
Random Forest	49.67
Linear Nearest Neighbor Search	42.33
SMO	38.67
Simple Logistic Regression	**53.00**
Multinomial Logistic Regression	**53.67**

Table 3: Single key classification using MFCC

Classification Algroithm	Accuracy (in %)
Alphabet keys (a-z)	
J48	34.40
Random Forest	66.88
Linear Nearest Neighbor Search	53.91
SMO	**74.33**
Simple Logistic Regression	**73.17**
Multinomial Logistic Regression	62.52
Numpad keys (0-9)	
J48	44.67
Random Forest	**73.67**
Linear Nearest Neighbor Search	62.00
SMO	**77.33**
Simple Logistic Regression	**70.67**
Multinomial Logistic Regression	56.33

5.1 Single Character Detection

Single character detection involves determining the accuracy of correctly recognizing a keystroke. For this purpose, two sets of keystroke samples are used: a labeled training set and a testing set. The testing set is evaluated against the training set using the methods described in Section 4.2 for determining the single character detection accuracy.

5.1.1 Experimental Setup. For our experiment, we collected 20 keystroke samples for each of the lowercase alphabetical keys *(a-z)* and each of the numpad keys *(0-9)*. The keyboard used was Dell SK-8125 and the recording was performed by a PC microphone (DX-USBMIC13). The scenario involved victim typing each key *(a-z; 0-9)* twenty times while communicating with the attacker on Skype. The typing was done in hunt and peck style where the victim used right hand index finger to press each key individually. The victim and the attacker are not present on the same network connection.

As per our attack model, no other sounds were present except the keystroke sounds. The keystroke sounds generated at the victim's end were transmitted through the microphone of the victim to the attacker by Skype. The microphone was placed at a distance of 15cm from the victim's keyboard to allow the attacker to have the best quality recording. The attacker recorded the victim's keystroke using a microphone from the other end of the Skype call. This captured our first attack setting involving a malicious end point.

5.1.2 Processing and Results. We divided the collected samples for each keystroke into training set (14 samples) and testing set (6 samples). We extracted the key press and the key release regions from each keystroke sample as detailed in Section 4.2. To get the best possible accuracy, we built models for each of the methods described in Section 4.2 and trained them on the obtained training set for each of the keys *(a-z; 0-9)*. The testing set for each key was then used to determine the accuracy of the models. The results are enumerated in Table 1.

For applying supervised machine learning, we used FFT coefficients and MFCC as classifying features for each keystroke. We calculated FFT features with a window of 441 samples with overlap of half the window size. We also used *"melfcc"* code provided at [4] with frequency wrap of type htkmel and used the first 32 channels in our calculations. We performed 10 fold cross validation on the collected samples of keystrokes with following algorithms: Simple Logistic Regression, Multinomial Logistic Regression, J48, Random Forest, SMO and Linear Nearest Neighbor Search.

The results are compiled in Table 2 and 3. From the results, we observe that MFCC provide a better classification accuracy than FFT coefficients. The best accuracy achieved by FFT coefficients was 39.54% using Simple Logistic Regression while using MFCC as features, the classification accuracy was 74.33% (SMO) and 73.17% (Simple Logistic Regression).

We compared the results obtained using multiple machine learning algorithms that use MFCC features against time domain methods like cross-correlation or derived features in frequency domain such as frequency-distance or frequency-time distance estimate. We clearly observe from Table 1 and Table 3 that machine learning method using MFCC delivers better results for a single keystroke accuracy for both alphabets and numpad keys. Hence, we utilize machine learning using MFCC features in our subsequent analysis of decoding random passwords and PINs over VoIP calls.

5.2 Random Passwords and PIN Detection

We now focus on the feasibility of decoding random passwords and PINs using same technique as in single character detection. Random passwords are difficult to decode as the accuracy of the attack method depends only on the single character detection rate as we can not use language models and dictionaries to deduce the correct word from a partially decoded word.

5.2.1 Experimental Setup. In order to test for random passwords, we allowed the victim to type 5 randomly generated 6-character passwords consisting of only lowercase alphabets *(a-z)*. The length of 6 characters is the minimum requirement for a password on most of the authentication systems. The victim typed each of the 5 passwords for a total of 10 times with a time interval of 5 seconds between successive password inputs for the same password. For testing PINs, we allowed the victim to type 5 randomly generated PINs of length 4, which is the standard length in most PIN authentication systems. Each PIN was typed 10 times allowing for an interval of 5 seconds between successive PIN entries for the same PIN similar to random passwords.

5.2.2 Processing and Results. For decoding random passwords, we used MFCC features extracted from password keystrokes, eavesdropped over the remote call. We used previously collected samples of single keystrokes (30 instances per key) to build the training model and keystrokes from each password attempt were used as the testing set. We used all ten attempts of each of the five random passwords against our training model and recorded the best obtained accuracy for each password over ten samples.

The results for password decoding is shown in Appendix Table 4 that indicate that SMO and Simple Logistic Regression algorithms were able to decode, on an average, more than 40% of the password over ten samples for each of the five random passwords. Single keystroke accuracies observed in Table 3 collaborate the effectiveness of these two algorithms when used with MFCC features. In terms of search space, SMO algorithm is able to decode almost half of the password, averaging about 3 characters for a 6-character length password. Since the experiments were performed over ten samples for each password, the required search space is $C_1^{10} \times C_3^{6} \times 26^3$ guesses. A brute force attack would require 26^6 guesses which means our attack reduces the search space by a factor of 88, for a 6-character length password.

If we consider a 10-character password, a brute force attack would require 26^{10} guesses. Using our best accuracy for password decoding (approximately 46.67%), the required number of guesses would be $C_1^{10} \times C_5^{10} \times 26^5$ reducing the search space by a factor of 4716. While the accuracies are not very high, they still show an underlying threat that has the potential to compromise user's security if the password is not truly random.

The decoding accuracy for 4-digit PINs is also shown in Appendix Table 4 demonstrating average accuracies using SMO, Simple Logistic Regression and Linear Nearest Neighbor Search. For a 4-digit PIN, the attacker can reveal as many as 3 digits contained in the PIN. The search space for this attack would be $C_1^{10} \times C_3^{4} \times 10^1$. When compared to a brute force attack (a search space of 10^4), the attacker's search space has been reduced by an order 10^3.

5.3 Network Sniffing

For this scenario, we assumed that the attacker had direct access to the network and can sniff audio packets from it. To emulate this scenario, we installed the recorder plugin for Skype named Supertintin [1] that can directly record the audio from Skype instead of relying upon the user's speaker to get access to the keystroke sounds. This also removes the need for the use of a microphone for the attacker as the recording was now done through the software.

We performed the experiment with no explicit limit imposed on the network bandwidth. Compagno et al. [8] showed that classification of remote keystrokes using MFCC suffers noticeable loss at and under 40 Kbits/s. Thus, we perform the experiment where the Skype conversation was working at almost best possible quality thereby creating the most favorable scenario for the attacker.

Top 5 Matching Character List: In this technique, we find the 5 most frequently matching characters for each of the characters *(a-z)*. We match each of the samples for each keystroke against the samples of remaining 25 keystrokes using our classification algorithms and arrange the matching candidates in decreasing order of frequency during the matching process. The matching character list describes the similarity of a given alphabetical key with other alphabetical keys. Thus, for each decoded character in our attack, we get a possible search space of 5 characters that could replace the decoded characters. If the replaced character resulted in a successful match, i.e. it was the same character typed by the victim, we count it as a correctly decoded character in the 6-character password.

For random passwords decoding, the attack was able to decode 2 out 6 characters for random passwords with a single character accuracy of 8.0% for the raw samples and after the application of top 5 matching list, it rose to 24.0%. For the 4 digit PIN, it was able to decode 3 out of 4 characters with the single character detection rate of 20.0% for the raw samples that rose to 50% after applying the top 5 matching character list. These results are in line with those of the end point attack setting and again demonstrate the feasibility for a network sniffing attacker, which will give rise to a significant factor reduction in the search space for passwords and PINs.

5.4 Local and Remote Eavesdropping Attacker

We used the same setup as used in the experimental setup described in Section 5.1.1. However, on this occasion, we also recorded the keystrokes locally at the victim's end by placing a microphone near the victim's keyboard. This setup mimics the threat scenario for eavesdropping on keystrokes locally as studied in previous literature ([5], [7], [9], [11], [15],[16]).

We build the training model as per Section 4.2 by using the keystrokes signals recorded locally at the victim's end. We then tested the remotely recorded keystrokes against this local eavesdropping based training model. We found out that using the top 5 matching character list as described in Section 5.2.2, we could only achieve a single character detection accuracy of 18.6% as compared to 32.7% as achieved in Section 5.2.2. This indicates that the steps involved in the transmission of the audio signal from victim's to attacker's end affect the features of signal to an extent that is lowers the accuracy of the classifier. This is to be expected when VoIP applications need to compress and encode the signals that may result in loss of some information about the signals.

6 NOISY AUDIO MASKING DEFENSE

Audio masking is used to hide susceptible sounds by introducing a different sound in the environment that should be able to cloak the susceptible sounds. In acoustic side channel attacks, the attacker exploits the sounds emanated from the victim's device (keyboard) to recover sensitive information. In our setup, the attacker records keystroke emanations that are transmitted over the VoIP call and decode them to extract random passwords, PINs, etc.

The main principle behind audio masking is to decrease signal to noise ratio (SNR) making it hard to separate the signal from noise. The audio emanations from the keystrokes constitute the signal and the masking sound is the background noise. If the loudness of the background noise is comparable to the signal, it becomes hard for an adversary to filter out the noise. To implement audio masking, an audio signal could be introduced in the victim's environment coexisting with the keystroke audio emanations. The masking signal could be generated from the victim's device or it could come from an external source.

6.1 Types of Masking Sound

For our defense based on masking keystroke emanations, we explore the masking ability of two types of masking sounds: white noise and *fake keystrokes*. We evaluate the masking signals on their ability to cloak keystroke emanations by measuring the accuracy of the adversary in decoding the keystrokes typed by the victim.

6.1.1 White Noise. Theoretically, white noise is defined as a random signal of constant spectral density and infinite bandwidth. However, in practice the bandwidth of white noise is regulated for the noise generator and is dependent on the context of generation. In our work, we are interested in masking keystroke sounds so we limit white noise to the frequency range of keystroke sounds. White noise is often used in office environments to drown out distracting sounds. While white noise is a simple masking signal, it lacks sophistication due to uniform frequency distribution leading to poor frequency spectrum overlap with keystroke sounds. It is also susceptible to filtering by most of the VoIP applications for enhancing speech and hence merely serves a baseline in our work from which we build upon more sophisticated masking signals.

6.1.2 Fake Keystrokes. Fake keystrokes refer to a pre-recorded sound of keystroke emanations where random keystrokes are typed without any pause between successive keystrokes. *Fake keystrokes* have similar frequency spectrum as the keystrokes typed by the victim during the VoIP call hence they lie in the same frequency spectrum. Due to this property, it may become difficult to filter *fake keystrokes* from real keystrokes for an attacker. In order to have maximum efficiency at masking the real keystrokes, the *fake keystrokes* should aim to overlap with the real keystrokes as much as possible. The overlap of the real and *fake keystrokes* may change the frequency features of the real keystrokes causing the resultant keystroke sound to be different than the real keystroke sound. Most VoIP applications, in our knowledge, do not filter out keystrokes during a call making the *fake keystrokes* a viable defense mechanism.

6.2 Experimental Setup

Our experimental setup for implementing the noisy defense mechanism was similar to the attack's experimental setup. The victim was in a VoIP call (Skype) with a malicious end user while entering sensitive information (passwords or PINs) at the same time on a computer terminal. The call from the victim was recorded at the malicious user's end using a PC microphone. In addition, we generated the masking sound using the victim's computer speaker. We divided the experiment in two stages: the first stage involved the use of white noise as the masking signal, and the second stage involved the usage of fake keystrokes as the masking signal.

The victim was instructed to type five unique 6-character random passwords, with each password being typed 30 times. Similarly, five unique 4-digit PINs were entered by the victim using the computer's keyboard, with each PIN being entered 30 times. We increased the number of samples for each password/PIN compared to the attack setup in order to test the defense against a more determined adversary. For the first stage of the experiment involving white noise, we generated white noise using the "wgn" function of Matlab in the frequency range 400Hz-12kHz at a sampling frequency of 44.1kHz. The resulting audio was played back using Windows Media Player by the inbuilt speaker at a sound level of 60dB. This is considered to be the sound level for a normal speech conversation and therefore any noise at the same level should probably be not distracting to the conversation. The keystroke sound was also measured and found to be around similar audio level.

For the second stage of the experiment, *fake keystrokes* were recorded offline by the victim locally, prior to the VoIP conversation with the malicious end point attacker. Since each keystroke lasts around 100ms and an average time duration between two successive keystrokes while entering a random password (we assume touch typing is not used for random passwords) is significantly more than 100ms, the victim presses random keys on the keyboard as fast as she can while recording the resulting keystroke sounds at the same time. Once the recording is complete, any part or whole of it can be played during the sensitive information entry event making sure that the masking sound is present during the entirety of this event.

6.3 Experimental Results

In order to test the viability of masking signals in cloaking acoustic emanations of keystrokes, we used our attack described in Section 5.2.2 to recover information from the eavesdropped signal.

6.3.1 In Presence of White Noise. Our results from the testing of white noise as a masking signal point out to the fact that white noise is recorded at a very low volume as compared to the keystrokes at the attacker's end. Hence in the presence of white noise, for the 6-character random passwords, we were able to recover on an average 5 out of 6 characters. Out of the 180 decoded characters (30 samples $\times$ length of a single password), we were able to correctly decode 9.2%, i.e, 16 characters correctly. After using the top 5 matching list, the percentage of correctly decoded characters went up to 35.9%.

For 4-digit PINs, we were able to recover almost all of the 4 digits of the PIN. Out of the 120 decoded numbers (30 samples $\times$ length of a single PIN), 12.5%, of digits were correctly decoded. Applying the top 5 matching list made the percentage of correctly decoded digits rise up to 67.2%. Thus, we can see that white noise has no effect upon the accuracy of the attack.[2] This is not surprising as most of the microphones and voice calling software implement some type of background noise suppression for improving the quality of the call. In other words, our experiments confirm that white noise as masking signal is not effective to defeat eavesdropping attack.

6.3.2 In Presence of Fake Keystrokes. Our recordings of the victim's keystrokes in the presence of *fake keystrokes* show that Skype does not filter out the fake keystrokes. This means that the real keystrokes typed by the victim may get obfuscated by the fake keystrokes that were played at the victim's end.

From the spectrum analysis, we observe that it may be difficult to identify *fake keystrokes* from real keystrokes as they lie in the same frequency spectrum. There may be cases of complete overlap of fake and real keystrokes, partial overlap between the fake and real keystrokes or no overlap between the fake and real keystrokes. In the first two cases, where there is a complete or a partial overlap between the fake and real keystrokes, the resultant keystroke signal will have different characteristics than the original real keystroke that was typed by the victim. In the case of no overlap between the fake and the real keystrokes, the *fake keystrokes* will still be mapped to some alphabetical key *(a-z)* and hence result in a false positive detection/insertion of a character.

[2]These accuracies are a bit higher compared to the ones reported in our attack section, which could be attributed to the independent setting in which this new data set was collected.

We tested the resultant keystroke audio signal captured by the attacker consisting of both real and *fake keystrokes*. The first challenge was to determine the correct number of samples. According to the experimental setup, 30 samples were recorded for a 6-character random password. However, the number of detected characters was fairly higher (261) than the number of characters actually typed by the victim (180). This results indicated that the recorded signal contains a high number of false positives in addition to the false negatives that may occur due to the overlapping of real and *fake keystrokes* as explained above.

Since there seems to be no viable way to identify the real keystrokes from the attack perspective, we can assume that the search space would increase drastically for an attacker trying to separate *fake keystrokes* from real keystrokes and the attack would fail. Also the randomness of *fake keystrokes* prevents the attacker from building up a profile of *fake keystrokes* over repeated trials. Thus we may say that *fake keystrokes* possess the capability of thwarting an acoustic side channel attack.

6.4 Keystroke Eavesdropping with Speech

While our threat model did not consider speech to be present during the eavesdropping of keystrokes, we examined the effect of speech on the accuracy of the remote eavesdropping attack. If speech has a detrimental effect on the attack's accuracy, it may have the potential to be used as a defensive measure against such attacks.

We tested the passwords and the PINs (2 instances randomly generated each) with 20 samples per instance. For each password/PIN entry on the victim's side, the victim also read aloud an English language text from a newspaper clip. The attack could deduce only 2 out of 6 characters for the password using the top 5 matching character list and 3 out of 4 for the PIN. This result indicates that the attack accuracy drops in presence of speech as compared to the scenario when no speech or masking signal is present. When compared to masking signals (white noise and fake keystrokes), speech may be more effective at masking keystrokes than white noise but less effective than the fake keystrokes over the VoIP channel.

7 NOISELESS AUDIO MASKING DEFENSE

While the approach of generating a masking sound at the victim's end may be able to hide keystroke sounds from a potential eavesdropper, it may also distract the victim from typing password. Since the malicious eavesdropper is not in proximity to the victim and the only feedback he can get is the from VoIP application itself, it may be beneficial for the victim if the masking signal is injected directly into the audio stream of VoIP application while being inaudible to the victim. In this manner, the masking sound will only be heard by the attacker and the victim will face no distraction while typing.

A virtual audio driver that provides the capability of rerouting the audio from a media software (the masking signal) to the audio input of the VoIP software while mixing it with audio from the user's microphone would be ideal for setting up a noiseless masking defense mechanism. Such a system would allay the usability concerns that arise when the masking signal is played through speakers on the user's end and affects the microphone output. This may be distracting to the user especially while trying to enter sensitive information that requires user's full attention.

Masking Signal: Same type of masking signal could be used in this setup as in noisy defense. White noise was not found to be very efficient at masking keystrokes in our noisy defense setup as Skype filters the noise in order to provide speech clarity. In our noiseless setup, we aim to inject white noise directly into Skype without it being relayed to Skype via microphone. We also use fake keystrokes in a similar manner since fake keystrokes seemed to have provided efficient masking to actual keystrokes from the victim's device. The experiments and evaluation of this approach are deferred to the full version of this paper.

8 DISCUSSION AND FUTURE WORK

We studied the feasibility of a remote eavesdropping acoustic side channel attack and demonstrated that such attack was able to reduce the search space for guessing the correct password/PIN when compared to a random guess. We also demonstrated that it is possible for an attacker to directly sniff the audio packets from the network with similar accuracy as a malicious end point attacker. We proposed two defense mechanisms based on sound masking to mitigate the investigated attacks. In our noisy defense setup, we generated the masking signal at the victim's end while the victim was typing the sensitive information. In the noiseless defense setup, a virtual audio driver can be used to mix the masking signal with the output from the victim's microphone and transmit it over the communication channel.

We evaluated two types of masking signal: white noise and fake keystrokes. We found out that white noise was not a suitable candidate to defend against our attack system due to noise suppression mechanisms deployed by the microphone and the voice calling applications. *Fake keystrokes* proved to be a better candidate against our attack system by effectively masking the keystroke emanations from the victim due to similarity of the *fake keystroke* signal and the real keystroke signal in the frequency domain which increased the search space for an attacker (more false negatives) during the attack. However, they should be randomly generated with random time intervals (none lasting more than 100ms) between successive *fake keystrokes* for effective overlap with real keystrokes. Longer passwords also increase the effectiveness of *fake keystrokes* due to increase in typing duration and are also harder to guess.

Continuous speech may also have potential as a defensive measure as it performs better than white noise since VoIP applications are configured to allow speech to be transmitted and suppress background noise. Compagno et al. [8] showed that the accuracy of their classifier decreased when speech became louder than keystroke sounds. When speech was 20dB higher than the keystroke sounds, their classifier reached a random guess baseline. We measured the loudness of keystrokes for three different keyboards (Dell SK-8125, Dell L100 and) using a sound meter and the average sound pressure level was 62dB. If the speech needs to be around 20db higher than the keystroke sound, it needs to be at 80dB which is beyond normal conversation loudness level (around 65-70db). Compared with the fake keystroke sounds that only need to be as loud as the keystroke themselves, we believe that speech as a masking signal may suffer from usability issue as the victim would only be conversing at a normal loudness level most of the time.

Real-world Defense Implementation: The design of our noiseless defense requires the masking signal generation capability to be in-built within the system (the source of the audio leakage) and the ability to inject the masking signal directly into the communication channel of VoIP call while being inaudible to the user. One such tool for virtual mixing and injection of audio into VoIP channel has already been introduced and utilized in our experiments. However, there needs to be a mechanism that allows the user to activate and deactivate the defense by the click of a button without explicitly going through audio settings of each involved application.

An alternative and zero-effort design for a real life implementation of the noiseless defense would be to detect the first key press for the password or sensitive input entry that will act as a trigger for the defense mechanism to activate and start injecting masking signal into the audio stream. In scenarios such as web login through passwords and PINs, the trigger can also be bound to the URL of the website, in particular to the login webpage. In our experiments, a Java swing based UI was constructed to test the defense mechanism. However, the defense mechanism can also be deployed as a browser plugin that can generate the masking sounds based on the visited URL. It is also possible to allow the user to enable or disable the defense mechanism at his discretion (e.g., by typing in a special character sequence such as "@@" as in an existing password manager application [12]).

Beyond Random Passwords and PINs: The focus of this paper was centered on short and sensitive input (e.g., random passwords and PINs). However, any arbitrary input containing sensitive data can be protected by our proposed defense. This may include, for example, email messages or documents prepared by the user while on a call that may be sensitive to the user. However, given these tasks are longer in time duration, further usability studies need to performed to analyze the effect of background noise generation on arbitrary text input in case of noisy defense. Our proposed noiseless defense remedies this issue and therefore is capable of working with even longer inputs and gets better at thwarting attacks with longer time duration as discussed before.

Other Acoustic Side Channels Attacks and Defenses over VoIP: The idea of applying acoustic side channel attacks to remote voice call setting is not limited to keyboard inputs alone. CPU acoustic emanations [10] and printer acoustic emanations [6] would also be potentially significant threats in a general remote setting explored in the paper. Further work, however, would be needed to study the impact of VoIP transmission over voice channel to these audio emanations. In the similar vein, our defense idea involving masking sounds would be applicable to defense against these attacks too that need to be studied in future work as well.

9 CONCLUSION

In this paper, we highlighted and quantified the threat of keyboard acoustic side channel attacks in the context of remote eavesdropping over voice calls. We showed the feasibility of the attack against short-length sensitive input, random passwords and numeric PINs based on off-the-shelf signal processing techniques. In contrast to localized attacks considered in prior work, our remote attack presents a threat to users' private information inadvertently leaked over a simple phone call unbeknownst to the user. We also proposed defense mechanisms against such attacks that attempt to obfuscate the acoustic leakage by inserting system-generated sounds while the user provides any sensitive input. Our noisy defense generates user-audible masking signal while noiseless defense silently combines the masking signal with microphone output while being inaudible to the user. The results for noiseless defense are deferred to the full version of the paper. The significance of our work lies in systematically investigating a known threat in a broader application setting and coming up with a near practical and user-transparent defense against this threat.

REFERENCES

[1] 2016. Supertintin. The Most Advanced Skype Video Recorder. http://http://www.supertintin.com/. (2016). Accessed: 2016-06-07.

[2] 2016. Touch typing. Wikipedia, available at. https://en.wikipedia.org/wiki/Touch_typing. (2016). Accessed: 2016-05-08.

[3] 2016. USA Bug Sweeps: We Will Find Hidden Audio Microphones & Recorders! http://usabugsweeps.com/tscm-debugging-audio/. (2016). Accessed: 2016-06-07.

[4] 2017. PLP and RASTA (and MFCC, and inversion) in Matlab using melfcc.m and invmelfcc.m. http://www.ee.columbia.edu/ln/rosa/matlab/rastamat/. (2017). Accessed: 2017-08-07.

[5] Dmitri Asonov and Rakesh Agrawal. 2004. Keyboard Acoustic Emanations. In *IEEE Symposium on Security and Privacy.*

[6] Michael Backes, Markus DÃĺurmuth, Sebastian Gerling, Manfred Pinkal, and Caroline Sporleder. 2005. Acoustic Side-Channel Attacks on Printers. In *USENIX Security Symposium.*

[7] Y. Berger, A. Wool, and A. Yeredor. 2006. Dictionary Attacks Using Keyboard Acoustic Emanations. In *ACM Conference on Computer and Communications Security.*

[8] Alberto Compagno, Mauro Conti, Daniele Lain, and Gene Tsudik. 2016. Don't Skype & Type! Acoustic Eavesdropping in Voice-Over-IP. https://arxiv.org/abs/1609.09359. (2016).

[9] A.H.Y. Fiona. 2006. Keyboard Acoustic Triangulation Attack. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.100.3156&rep=rep1&type=pdf. (2006). Final Year Project.

[10] Daniel Genkin, Adi Shamir, and Eran Tromer. 2014. RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis. In *Advances in Cryptology - CRYPTO.*

[11] Tzipora Halevi and Nitesh Saxena. 2012. A Closer Look at Keyboard Acoustic Emanations: Random Passwords, Typing Styles and Decoding Techniques. In *ACM Symposium on Information, Computer and Communications Security.*

[12] Blake Ross, Collin Jackson, Nick Miyake, Dan Boneh, and John C Mitchell. 2005. Stronger password authentication using browser extensions. In *USENIX Security Symposium.*

[13] Charles Wright, Lucas Ballard, Scot Coulls, Fabian Monrose, and Gerald Masson. 2008. Spot me if you can: recovering spoken phrases in encrypted VoIP conversations. In *IEEE Symposium on Security and Privacy.*

[14] T. Zhu, Q. Ma, S. Zhang, and Y. Liu. 2014. Context-free attacks using keyboard acoustic emanations. In *ACM SIGSAC Conference on Computer and Communications Security.* 453–464.

[15] Tong Zhu, Qiang Ma, Shanfeng Zhang, and Yunhao Liu. 2014. Context-free Attacks Using Keyboard Acoustic Emanations. In *ACM Conference on Computer and Communications Security.*

[16] Li Zhuang, Feng Zhou, and J. D. Tygar. 2009. Keyboard Acoustic Emanations Revisited. *ACM Transactions on Information and System Security* 13, 1 (2009).

A APPENDIX

Table 4: Decoding accuracy using MFCC features

	Accuracy in %					
Password	**J48**	**Random Forest**	**Linear Nearest Neighbor Search**	**SMO (SMV)**	**Simple Logistic Regression**	**Multinomial Logistic Regression**
hfkgml	33.33	50.00	50.00	33.33	33.33	33.33
jotfmk	16.67	33.33	16.67	50.00	33.33	33.33
loughl	16.67	33.33	66.67	50.00	50.00	50.00
mlcabd	33.33	33.33	33.33	33.33	50.00	33.33
vaorkg	33.33	33.33	33.33	66.67	50.00	33.33
Average accuracy	**26.67**	**36.66**	**40.00**	**46.67**	**43.33**	**36.66**
PIN						
0075	75.00	75.00	75.00	75.00	75.00	50.00
1282	50.00	100.00	75.00	75.00	50.00	50.00
1446	50.00	50.00	75.00	50.00	75.00	75.00
3684	50.00	50.00	75.00	75.00	75.00	75.00
4793	50.00	50.00	50.00	50.00	75.00	50.00
Average accuracy	**55.00**	**65.00**	**70.00**	**65.00**	**70.00**	**60.00**