Don't Skype & Type! Acoustic Eavesdropping in Voice-Over-IP

Alberto Compagno*, Mauro Conti†, Daniele Lain† and Gene Tsudik‡
*Department of Computer Science, Sapienza University of Rome
Email: compagno@di.uniroma1.it
†Department of Mathematics, University of Padua
Email: conti@math.unipd.it, daniele.lain@studenti.unipd.it
‡Department of Computer Science, UC Irvine
Email: gts@ics.uci.edu

***Abstract*—Acoustic emanations of computer keyboards represent a serious privacy issue. As demonstrated in prior work, spectral and temporal properties of keystroke sounds might reveal what a user is typing. However, previous attacks assumed relatively strong adversary models that are not very practical in many real-world settings. Such strong models assume: (i) adversary's physical proximity to the victim, (ii) precise profiling of the victim's typing style and keyboard, and/or (iii) significant amount of victim's typed information (and its corresponding sounds) available to the adversary.**

In this paper, we investigate a new and practical keyboard acoustic eavesdropping attack, called *Skype & Type (S&T)*, which is based on Voice-over-IP (VoIP). S&T relaxes prior strong adversary assumptions. Our work is motivated by the simple observation that people often engage in secondary activities (including typing) while participating in VoIP calls. VoIP software can acquire acoustic emanations of pressed keystrokes (which might include passwords and other sensitive information) and transmit them to others involved in the call. In fact, we show that very popular VoIP software (Skype) conveys enough audio information to reconstruct the victim's input – keystrokes typed on the remote keyboard. In particular, our results demonstrate that, given some knowledge on the victim's typing style and the keyboard, the attacker attains top-5 accuracy of 91.7% in guessing a random key pressed by the victim. (The accuracy goes down to still alarming 41.89% if the attacker is oblivious to both the typing style and the keyboard). Finally, we provide evidence that *Skype & Type attack* is robust to various VoIP issues (e.g., Internet bandwidth fluctuations and presence of voice over keystrokes), thus confirming feasibility of this attack.

I. Introduction

Electronic devices, particularly smartphones and tablets, are some of the most personal objects in many people's lives, storing and managing private and sensitive information, such as photos, passwords, and messages. Protecting such sensitive data by encryption is a common approach to prevent unauthorized access and disclosure. However, there is no protection if data is leaked before encryption.

Eavesdropping on physical signals, such as acoustic or electromagnetic emanations, is one way to recover either: (1) clear-text data before encryption, e.g., during its input or visualization, or (2) encryption keys, e.g., during data encryption and decryption. Indeed, the history of eavesdropping on physical signals dates back to 1943, when a Bell engineer discovered that an oscilloscope can retrieve the plain-text from electromagnetic emanations of a Bell Telephone model 131-B2 – a mixing device used by the US Army to encrypt communications [8].

A common target for physical eavesdropping attacks are input peripherals, such as keyboards, touchscreen surfaces and printers. Some examples of physical eavesdropping attacks that were already exploited are: electromagnetic emanations of keyboards [29], videos of users typing on a keyboard [4] or on a touchscreen [25], and keyboard acoustic emanations [3]. In particular, the research community invested a lot of effort into studying keyboard acoustic emanations, showing that it is a very serious privacy issue. Eavesdropping on keyboard emanations allows an adversary to learn what a victim is typing, based on the sound produced by keystrokes. During the attack, the sound is usually collected either using microphones [3, 10, 11, 5, 36, 14, 30, 35, 18], or by exploiting various sensors (e.g., accelerometers of mobile devices [17, 32]) to reconstruct the same acoustic information. Once collected, the sound stream is typically analyzed using various techniques: supervised [3, 10, 11, 18] and unsupervised [36, 5] machine learning, or triangulation [14, 30, 35]. The final result of this analysis is a full or partial reconstruction of the victim's input.

To the best of our knowledge, all previous attacks on keyboard acoustic emanations require a compromised (malicious, i.e., controlled by the adversary) microphone near the victim's keyboard [3, 10, 11, 18, 5, 14, 30, 35]. We believe that this requirement limits applicability of the attack, thus reducing its feasibility in the real world. Although increasing universal popularity of smartphones might ease deployment of a compromised microphone close to the victim (e.g., one in the attacker's smartphone), the adversary still needs to either physically position it, or control one, near the victim. Otherwise, a keyboard acoustic eavesdropping attack cannot be performed. Moreover, some previous approaches have even more restrictive requirements: (i) a lot of training information to cluster [5], thus requiring long-term collection of keystroke sounds, or (ii) precise profiling of the victim's typing style and keyboard [3, 10, 11, 18].

In this paper, we propose a different type of the keyboard acoustic eavesdropping attack that does not require the adversary to control a microphone near the victim, and works with a limited amount of keystroke data. We call it *Skype & Type attack*, or *S&T attack* for short. As a basis for this attack,

we exploit Voice-over-IP (VoIP), one of the most popular and pervasive voice communication technologies used by great multitudes of people throughout the world. We premise our work on a very simple observation:

> People involved in VoIP calls often engage in secondary activities, such as: writing email, contributing their wisdom to social networks, reading news, watching videos, and even writing research papers. Many of these activities involve using the keyboard (e.g., entering a password). VoIP software automatically acquires all acoustic emanations, including those of the keyboard, and transmits them to all other parties involved in the call. If one of these parties is malicious, it can determine what the user typed based on keystroke sounds.

We believe this work is both timely and important, especially due to growing pervasiveness of VoIP software[1]. Thus, keyboard acoustic eavesdropping attacks, if shown to be realistic, should concern every VoIP user. None of the previous studies on keyboard acoustic eavesdropping [3, 10, 11, 18, 5, 14, 30, 35] considered either the setting of our attack, or the features of VoIP software. In particular, VoIP software performs a number of transformations to the sound before transmitting it over the Internet, e.g., downsample, approximation, compression, and disruption of the stereo information by mixing the sound into a single channel. Such transformations have not been considered in the past. In fact, for some prior results, these transformations conflict with the assumptions, e.g., [14, 30, 35] require stereo information for the recorded audio stream. Therefore, conclusions from these results are largely inapplicable to *S&T attack*.

A. Contributions

The contributions of this paper are:

- We demonstrate *S&T attack* based on keyboard acoustic eavesdropping over VoIP software, with the goal of recovering text typed by the user during a VoIP call with the attacker. *S&T attack* can also recover random text, such as randomly generated passwords or PINs. We take advantage of the spectral features of keystroke sounds, and analyze them using supervised machine learning algorithms.
- We evaluate *S&T attack* over a very popular VoIP software: **Skype**. We designed a set of attack scenarios that we consider to be more realistic than those used in prior results on keyboard acoustic eavesdropping. We show that *S&T attack* is highly accurate with minimal profiling of the victim's typing style and keyboard. It remains quite accurate even if neither profiling is available to the adversary. Our results show that *S&T attack* is very feasible, and applicable to real world settings under realistic assumptions. In particular, *S&T attack* allows the adversary to greatly speed up brute-force cracking of random passwords.
- We show, via extensive experiments, that *S&T attack* is robust to VoIP-related issues, such as limited available bandwidth that degrades call quality, as well as to voice conversations on top of keystroke sounds.
- Based on the insights from the design and evaluation phases of this work, we propose some tentative countermeasures to *S&T attack* and similar attacks that exploit spectral properties of the keystroke sounds.

[1] In 2016, Skype reached 300 million active monthly users [19].

B. Organization:

Section II overviews related literature and state-of-the-art on keyboard eavesdropping. Next, Section III describes the system model for our attack and various attack scenarios. Section IV, presents *S&T attack*. Then, Section V evaluates *S&T attack*, discusses our results and impact of VoIP-specific issues. Section VI shows a practical application of *S&T attack* to password cracking. Finally, Section VII proposes some potential countermeasures, and Section VIII summarizes the paper and overviews future work.

II. Related Work

The problem of eavesdropping user input on keyboards is an active and popular area of research. In the following, we first report, in Section II-A, attacks that use acoustic emanations in order to recover the victim's typed text. We then report, in Section II-B, attacks that eavesdrop different emanations of keyboards, such as WiFi signal, wireless connections, vibrations of surfaces, and optical emanations.

A. Attacks Using Sound Emanations

Research on keyboard acoustic eavesdropping started with the seminal paper of Asonov and Agrawal [3]. They showed that, by training a neural network on a specific keyboard, they could later achieve good performance in eavesdropping the input of the same keyboard, or of keyboards of the same model. They also investigated why this is possible, and eventually discovered that the plate beneath the keyboard, where the keys hit the sensors, has a drum-like behavior. This causes the sound of the keys to be slightly different one another. This paper started a new field of research in acoustic eavesdropping of keyboards. We can divide subsequent works based on whether they leverage statistical properties of the sound spectrum, or they use timing information.

Approaches that leverage statistical properties of the spectrum use machine learning techniques, and they then differentiate in whether they use supervised [3, 10, 11, 18] or unsupervised learning [5, 36] paradigms.

Supervised learning approaches require many labeled samples, and are highly dependent on the specific keyboard they were trained with [3], as well as on the typing style [10, 11]. These techniques have been used to recover random text, such as passwords [10, 11], by combining cross-correlation information with a distance measure derived from Fast Fourier Transform (FFT) coefficients. These FFT coefficients alone were shown to carry enough information to recover keystrokes [18] using a neural network, similarly to [3]. Overall, supervised learning approaches have very high accuracy, however at the price of strong assumptions on how the data is collected: obtaining labeled samples of the acoustic emanations of the victim on his keyboard can be difficult.

Unsupervised learning approaches can cluster together keys from the sound, or can generate sets of constraints between

different presses. Zhuang et al. [36] clustered together the sound of the keys, and then assigned labels to the clusters by leveraging the relative frequency of letters of the input language. Their work is, therefore, especially suited to eavesdrop on long texts where the language is known. Berger et al. [5] generated sets of constraints from the recorded sound, and then selected words from a dictionary that match these constraints. Unsupervised learning approaches have the advantage that they do not require ground truth, which can be difficult to obtain. However, they have strong assumptions on the user input, such as obtaining many samples, i.e., the emanations from the input of a long text [36], or requiring it to be a dictionary word [5], and are less effective when the input is random.

Another possible approach is to leverage timing information. One convenient way to exploit timing information is using multiple microphones, such as mobile phone microphones [14, 30, 35], and to analyze the Time Difference of Arrival (TDoA) information to triangulate the position of the pressed key. These proposals differ mostly in whether they required a training phase [30], or used one [14] or more [35] mobile phones. Another direction to exploit timing information is keystroke dynamics. However, such approaches used different side channels, rather than acoustic ones. Zhang et al. [34] obtained timing information from `procfs`, and built sets of constraints between consecutive letters that are useful to recover both random and meaningful text. Song et al. [26] remotely obtained timing information during an interactive SSH session, ultimately recovering the victim's password. However, it would be possible to perform both these attack by extracting timing information from acoustic emanations.

B. Attacks Using Other Emanations

Other approaches to the problem of keyboard eavesdropping used different side channels, rather than acoustic ones. Besides the work on keystroke dynamics that we already described, it is possible to leverage many other emanations. Typing on a keyboard causes its electrical components to emit electromagnetic waves, and it is possible to collect such waves, in order to recover the keystrokes [29]. Furthermore, typing on a keyboard causes the surface where the keyboard is placed to vibrate. These vibrations can be collected by the accelerometer of a smartphone, to understand the pressed keys [17]. Furthermore, typing on a keyboard causes the surface where the keyboard is placed to vibrate. These vibrations can be collected by the accelerometer of a smartphone, to understand the pressed keys [17]. Understanding the movement of the user's hand on a keyboard is another method of recovering his input. It is possible to do this using videos of the user typing [4], or using WiFi signal fluctuation on the laptop used by the user [2]. Finally, it is worth mentioning Wei et al. [32] whose technique allows to reconstruct a target sound by using wireless vibrometry, without the need of a microphone. Their technique could be used to recover the acoustic emanations of a keyboard.

III. System and Threat models

In order to identify precise attack scenarios, we begin by defining a system model that serves as the base for the attacks. Section III-A describes our assumptions about the victim and the adversary, and carefully defines the problem of keyboard acoustic eavesdropping. Section III-B then presents some realistic attack scenarios (within the system model) and discusses them in relation to the state-of-the-art.

A. System Model

Our system model is depicted in Figure 1. We suppose that the victim owns a desktop or a laptop computer with a built-in or attached keyboard, i.e., **not** a smartphone or a tablet-like device. Hereafter it is referred to as target-device. A genuine copy of some VoIP software is assumed to be installed on target-device; this software is not compromised in any way. Also, target-device is connected to the Internet and engaged in a VoIP call with at least one adversary and perhaps other parties.

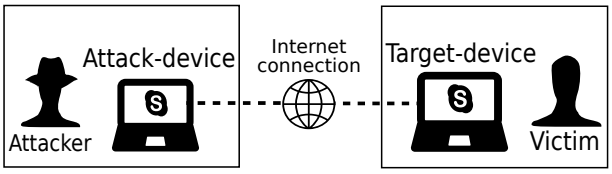


Fig. 1: The system model of our attack.

The attacker[2] is a malicious user who wants to learn some private information about the victim. The attacker owns and fully controls a computer that we refer to as attack-device, which has a genuine unmodified version of the same VoIP software as target-device. The attacker uses attack-device to receive and record the victim's acoustic information using VoIP software. We assume that the attacker relies solely on information provided by VoIP software. In other words, *during the attack*, it receives no additional acoustic information from the victim, besides what VoIP software transmits to attack-device.

B. Threat Model

S&T attack transpires as follows: during a VoIP call between the victim and the attacker, the former types something on target-device, e.g., a password, that we refer to as target-text. Typing target-text causes acoustic emanations from target-device's keyboard, which are then picked up by the target-device's microphone and transmitted to the attacker by VoIP. The goal of the attacker is to learn the target-text by taking advantage of these emanations.

We make the following assumptions:

- As mentioned above, the attacker has no real-time audio-related information beyond that provided by VoIP software. Acoustic information can be degraded by VoIP software by downsampling and mixing. In particular, without loss of generality, we assume that audio is converted into a single (mono) signal, as happens with some VoIP software.
- If the victim discloses some keyboard acoustic emanations **together** with the corresponding plaintext – the actual pressed keys (aka *ground truth*) — the volume of this information is small, on the order of a chat message or

[2]We use the terms *adversary* and *attacker* interchangeably.

a short e-mail. We expect it to be no more than a few hundred characters.
- target-text is very short (e.g., $\approx$ 10 characters) and random, as is typically the case for an ideal password.

We now consider some realistic *S&T attack* scenarios. We describe them starting with the more generous setting where the attacker knows the victim's typing style and keyboard model, proceeding to the more challenging one where the attacker has neither type of information.

1) COMPLETE PROFILING: In our initial scenario, the attacker knows some of the victim's keyboard acoustic emanations on target-device, along with the ground truth for these emanations. This might happen if the victim unwittingly provides some text samples to the attacker during the VoIP call, e.g., sends chat messages, edits a shared document, or sends email[3]. We refer to such disclosed emanations as *"labeled data"*. To be realistic, the amount of labeled data should be limited to a few samples for each character.

We refer to this as *Complete Profiling* scenario, since the attacker has maximum information about the victim. This corresponds to attack scenarios used in prior supervised learning approaches [3, 10, 11, 18], with the difference that we collect acoustic emanations using VoIP software, while others collect emanations directly from microphones physically near target-device.

2) USER PROFILING: In this scenario, we assume that the attacker does not have any labeled data from the victim on target-device. However, the attacker can collect training data of the victim while the victim is using the same type of device (including the keyboard) as target-device[4]. This can be done, e.g., with social engineering techniques, or with the help of an accomplice. We refer to this as *User Profiling* scenario, since unable to profile target-device, the attacker profiles the victim's (user's) typing style on the same device type.

3) MODEL PROFILING: This is the most challenging scenario, though the most realistic one. In it, the attacker has absolutely no training data for the victim. In this setting, the attacker and the victim are engaged in a VoIP call and the only thing that the victim types on the keyboard is the secret (password) that the attacker wants to obtain.

The initial goal of the attacker is to determine what laptop the victim is using. To do so, we assume that the attacker maintains a database of sounds from previous attacks on other victims. If the attacker already profiled the model of the current victim's target-device, it can use this information to mount the attack. We refer to this as *Model Profiling* scenario, since although the attacker can not profile the current victim, it can still profile a device of the same model as target-device.

[3]The ground truth could also be collected offline, if the adversary happened to be near the victim, at some point before the actual attack. Note that this attack still does not require physical proximity between the attacker and the victim in *real time*

[4]In case the target-device is a desktop, knowing the model of the desktop does not necessarily mean knowing the type of the keyboard. However, in mixed video/audio call the keyboard model might be visually determined, when the keyboard is placed in the visual range of the camera.

IV. SKYPE & TYPE ATTACK

In this section, we describe *S&T attack* in detail. Recall that all of our scenarios involve the attacker engaged in a VoIP call with the victim. During the call, the victim types something on target-device's keyboard. *S&T attack* proceeds as described below and illustrated in Figure 2.

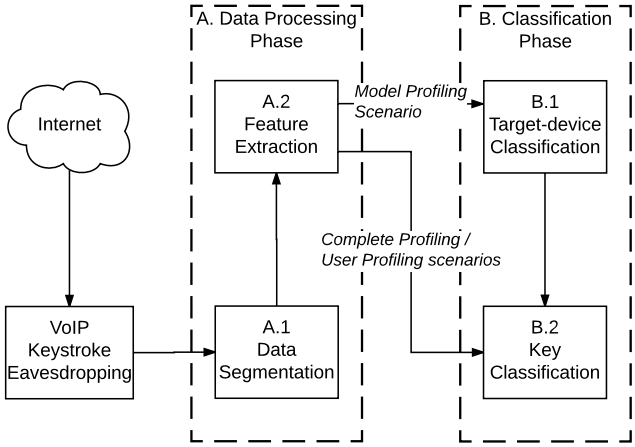


Fig. 2: *S&T attack* steps.

First, the attacker receives acoustic emanations of target-device's keyboard over VoIP, and records them directly. For example, the attacker routes VoIP software output to some local recording software. Then, the attacker conducts two main attack phases: (i) data processing, and (ii) data classification. Each phase involves two steps:

1) The data processing phase includes data segmentation and feature extraction steps. They are performed in each of the three attack scenarios defined in Section III.
2) Data classification phase includes target-device classification and key classification steps. Their execution depends on the specific attack scenario:
   – In *Complete Profiling* and *User Profiling* scenarios the attacker already profiled the victim, either on target-device (*Complete Profiling*) or on a device of the same model (*User Profiling*). The attacker uses such data as a training set, and proceeds to classify target-text. This case is indicated in Figure 2 by the path where key classification follows feature extraction.
   – In *Model Profiling* scenario the attacker has no knowledge of the victim's typing style or target-device. Thus, the attacker begins by trying to identify target-device by classifying its keyboard sounds. The attacker then proceeds to classify target-text by using correct training data. This case is indicated in Figure 2 by the path where target-device classification is the next step after feature extraction.

We now describe data processing, and data classification phases in more detail.

A. Data Processing Phase

The main goal of this phase is to extract meaningful features from acoustic information. The first step is *data segmentation* needed to isolate the keystroke sounds within the

recording. Subsequently, using these sound samples, we build derived values (called features) that represent properties of acoustic information. This step is commonly known as *feature extraction*.

1) Data Segmentation: We perform data segmentation according to the following observation: the waveform of the keystroke sound presents two distinct peaks, shown in Figure 3. These two peaks correspond to the events of: (1) the finger pressing the key – *press* peak, and (2) the finger releasing the key – *release* peak. Similar to [3], we only use the press peak to segment the data and ignore the release peak. This is because the former is generally louder than the latter and is thus easier to isolate, even in very noisy scenarios.

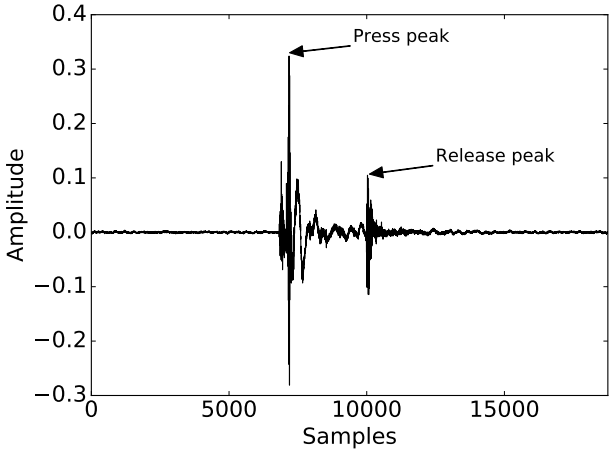


Fig. 3: Waveform of the "*A*" key, recorded on an Apple Macbook Pro 13" laptop.

To perform automatic isolation of keystrokes, we set up a detection mechanism as follows: we first normalize the amplitude of the signal to have root mean square of 1. We then sum up the FFT coefficients over small windows of 10ms, to obtain the energy of each window. We detect a press event when the energy of a window is above a certain threshold. We then extract the subsequent 100ms [5, 36] as the waveform of the given keystroke event. If the sounds of the keys are very closely spaced it is possible to extract a shorter waveform.

2) Feature Extraction: As features, we extract the mel-frequency cepstral coefficients (MFCC) [15]. These features capture statistical properties of the spectrum of the sound, which are the only information that we can use. Indeed, due to the mono acoustic information, it is impossible to setup an attack that requires stereo audio and leverages TDoA, such as [14, 30, 35]. Among possible statistical properties of the sound spectrum – including: MFCC, FFT coefficients, and cepstral coefficients – we chose MFCC which yielded the best results. To select the most suitable property, we ran the follow experiment: using a Logistic Regression classifier we classified a dataset with 10 samples for each of the 26 keys corresponding to the letters of the English alphabet, in a 10-fold cross-validation scheme. We then evaluated the accuracy of the classifier with the various spectral features: FFT coefficients, cepstral coefficients, and MFCC. Results of this experiment are shown in Figure 4, where it is easy to observe that MFCC offers the best features. For the MFCC we use parameters similar to those in [36] (i.e., we use a sliding window of 10ms, with a step size of 2.5ms, 32 filters in the mel scale filterbank, and we use the first 32 MFCC).

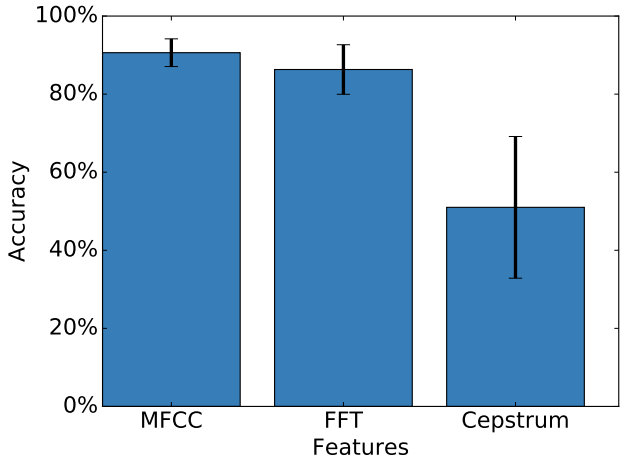


Fig. 4: Average accuracy of single key classification, with various features.

B. Classification Phase

In this phase, we apply a machine learning algorithm to the features extracted in the Data Processing phase, in order to perform:

- target-device classification using all keystroke sound emanations that the attacker received.
- Key classification of each single keyboard key of target-device by using sound emanations of the keystrokes.

Each classification task is performed depending on the scenario. In *Complete Profiling* and *User Profiling* scenarios, the attacker already profiled the victim on target-device, and or on a device of the same model, respectively. Then, the attacker loads correct training data and performs the key classification task, to understand target-text.

In contrast, in *Model Profiling* scenario, the attacker first performs target-device classification task, in order to identify the model. Next, the attacker loads correct training data, and proceeds to the key classification task.

The only viable machine learning approach for both the key and target-device classification tasks is a supervised learning technique. As discussed in Section III-B, approaches that require lots of data to cluster, such as [5] are incompatible with our assumptions, because we have a small amount of both training and testing data. Moreover, randomness of target-text makes it impossible to implement constraint-based approaches, which would require target-text to be a meaningful word, as in [36].

1) Target-device Classification: We consider the task of target-device classification as a multiclass classification problem, where different classes correspond to different target-device models known to the attacker. More formally, we can define the problem as follows: we have a number of samples

$s \in S$, each represented by its feature vector $\vec{s}$, and generated by the same target-device l of model $\tilde{l}$, among a set $\mathcal{L}$ of known target-device models. We want to know which target-device model generated the samples in S, by classifying every sample s, and then taking the mode of these predictions. To perform this classification task, we use a k-nearest neighbors (k-NN) classifier with $k = 10$ neighbors. We empirically determined these parameters as best-suited for the problem at hand.

2) Key Classification: We consider key classification to be a multiclass classification problem, where different classes correspond to different keyboard keys. More formally, we can define the problem in terms of expected output, as follows: given the sample s generated by the victim pressing key k among the set $\mathcal{K}$ keyboard keys, we want to determine the probability of one of the keys $k' \in \mathcal{K}$ being equal to k.

To evaluate the quality of the classifier on multiple predictions we use the *accuracy* and the *top-n accuracy* measures. Given the true values of k, accuracy is defined in the multiclass classification case as the fraction of correctly classified samples over all samples. Formally, if y_i is the true value of the i-th sample of our test set, and $\hat{y}_i$ is its predicted value, the accuracy measure is:

$$acc(y, \hat{y}) = \frac{\sum_{i=0}^{|y|} (y_i = \hat{y}_i)}{|y|}.$$

Given the definition of accuracy, the top-n accuracy is defined as:

$$acc_n(y, \hat{y}) \frac{\sum_{i=0}^{|y|} (y_i \in \hat{y}_i)}{|y|},$$

where the classifier is allowed to make n most probable guesses, and $\hat{y}_i$ is a set of n predictions $\hat{y}_i^0 \ldots \hat{y}_i^{n-1}$ for every i-th sample of our test set.

To perform key classification, we use a Logistic Regression (LR) classifier, since it outperformed all other classifiers, including: Linear Discriminant Analysis (LDA), Support Vector Machines (SVM), Random Forest (RF), and k-nearest neighbors. We show this in an experiment which uses each candidate classifier to classify a dataset of 10 samples, for each of the 26 keys corresponding to the letters of the English alphabet, in a 10-fold cross-validation scenario. We use MFCC as features, and, for each classifier, we optimize the hyper-parameters with an extensive grid search.

Results of this experiment are shown in Figure 5. It demonstrates that the best performing classifiers are LR and SVM, especially if the classifier is allowed to make a small number of predictions (between 1 and 5), which is more realistic in an eavesdropping setting. In particular, both LR and SVM exhibit around 90% accuracy for the first guess (top-1 accuracy), and over 98.9% accuracy for five guesses (op-5 accuracy). However, LR slightly outperforms SVM until top-4.

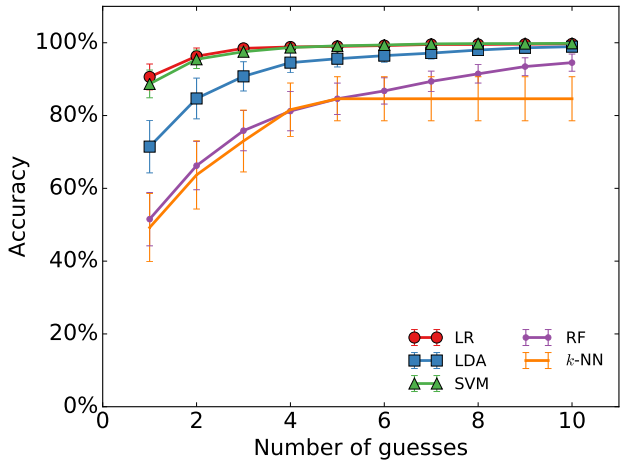


Fig. 5: Average top-n accuracy of single key classification, as a function of the number of guesses, with different classifiers.

V. Evaluation

We now evaluate the performance of *S&T attack* on a thorough set of experiments, that cover all scenarios we described. We choose **Skype** as underlying VoIP software. This is based on three reasons: (i) Skype is one of the most popular VoIP software choices [19, 1, 21]; (ii) its codecs are used in the Opus, an IETF standard [27], employed in many other VoIP applications, such as Google Hangouts and Teamspeak [20]; (iii) it reflects our general assumption about mono audio. Therefore, we believe Skype is fairly representative of a wide range of VoIP software packages and its world-wide popularity makes it appealing for attackers.

In this section, we first describe how we collected experimental data. Then, we report the results of our experiments. Finally, we analyze several issues arising from using VoIP and Skype to perform *S&T attack*, such impact of bandwidth reduction on the quality of the audio, and the likeliness for the keystroke sound to overlap with the voice of the victim, that might degrade the efficacy of *S&T attack*.

A. Data Collection

We collected data from five distinct users. For each user, the task was to press the keys corresponding to the English alphabet, sequentially from "A" to "Z", and to repeat the sequence ten times, first by only using the right index finger (activity known as *Hunt and Peck* typing, or just *HP* typing), and then by using all the fingers of both hands (*Touch* typing) [11]. We believe that typing the letters in the order of the English alphabet, rather than, for example, typing English words did not introduce bias. Typing the English alphabet in order is similar to typing random text, that *S&T attack* targets.

Note that collecting only the sounds corresponding to letter keys, instead of those for the entire keyboard, does not affect our experiment. The "acoustic fingerprint" of every key is related to its position on the keyboard plate [3]. Therefore, all keys behave, and are detectable, in the same way [3]. Thanks to this property, we believe that evaluating only the letters is sufficient to prove our point. Moreover, because of this property, it would be trivial to extend our approach to consider different keyboard layouts, by associating the keystroke sound with the position of the key, rather than the symbol of the key, and then mapping the positions to different keyboard layouts.

Every user ran the experiment on six laptops: two Apple Macbooks Pro 13" mid 2014, two Lenovo Thinkpads E540, and two Toshiba Tecras M2. We selected these laptops to be representative of many common modern laptop models:

Macbook Pro is a very popular aluminium made high-end laptop, Lenovo Thinkpad E540 is a 15" mid-priced laptop, and Toshiba Tecra M2 is an older laptop model, manufactured in 2004. All acoustic emanations of the laptop keyboards were recorded by the microphone of the laptop in use, with Audacity software v2.0.0. We recorded all data with a sampling frequency of 44.1kHz, and then saved it in WAV format, 32-bit PCM signed.

We then filtered the data obtained via Skype software. To do so, we used two machines running Linux, with Skype version 4.3.0.37, connected to a high-speed network. To simulate a microphone input from the calling computer, we routed recorded data to Skype software, which "believed" that the input came from a microphone. During the calls, there was no sensible data loss. We analyze bandwidth requirements needed for data loss to occur, and the impact of bandwidth reduction, in Section V-C1.

At the end of the data collection and processing phase, we obtained datasets for all the five users on all six laptops, with both the HP and Touch-typing styles. All datasets are both unfiltered, i.e., raw recordings from the laptop's microphone, and filtered through Skype. Each dataset consists of 260 samples, 10 for each of the 26 letters of the English alphabet. The number of users and of laptops we considered is in line with related work on the topic [3, 10, 11, 18], where only $1-3$ keyboards were involved and a single test user.

B. S&T Attack Evaluation

We evaluated *S&T attack* with all scenarios described in Section III-B. We evaluated *Complete Profiling* scenario in great detail, by analyzing performance of *S&T attack* separately for all three laptop models, two different typing styles, and VoIP filtered and unfiltered data. We consider this to be a favorable scenario for showing the accuracy of *S&T attack*. In particular, we evaluate performance by considering VoIP transformation, and various combinations of laptops and typing styles. We then further analyzed only the realistic combination of *Touch* typing data, filtered with Skype.

We evaluated *S&T attack* accuracy in recognizing single characters, according to the top-n accuracy, defined in [6], as mentioned in Section IV-B2. As a baseline, we considered a random guess with accuracy x/l, where x is the number of guesses, and l is the size of the alphabet. On our experimental setup, therefore, accuracy of the random guess is $x/26$, since we considered the 26 letters of the English alphabet. Because we want to eavesdrop on possibly random text, we can not use "smarter" random guesses that, for example, take into account relative frequencies of letters in a given language.

1) Complete Profiling Scenario: To evaluate the scenario where the victim disclosed some labeled data to the attacker, we proceeded as follows. We considered all the datasets one at a time, where each dataset we recall it consists of 260 samples (10 for every letter of the alphabet), in a stratified 10-fold cross-validation scheme[5]. For every fold, we performed feature selection on the train data using a Recursive Feature Elimination algorithm [9]. We calculated the accuracy of the classifier over each fold, and finally we calculated the mean and standard deviation of the accuracy values we obtained.

Figure 6 depicts the results. In particular, we show the results of the different combinations of typing style (*HP* or *Touch*), and of the type of data (filtered through Skype or unfiltered). In Figure 6a, we show the accuracy on the HP typing and unfiltered data combination, that we consider as the most favorable. In this case, *S&T attack* achieves the lowest performance on the Lenovo laptops with top-1 accuracy of 52.4%, and a top-5 accuracy 84.5%. On the Macbook Pro and Toshiba laptops, we obtain instead a very high top-1 accuracy, 90.1% and 74.5% respectively, and a top-5 accuracy of 98.9% and 94.2%, respectively. We believe that the different performance between the three laptops is due to different build qualities, where the keyboard of the particular Lenovo laptop model that we considered is made of cheap plastic materials.

We report the other results for the HP typing and Skype filtered data combination, for Touch typing and unfiltered data combination and for Touch typing and Skype filtered data combination in figures 6b, 6c, and 6d, respectively. We can see that there is a very small difference between the four combinations. For example, on the Macbook Pro laptop we have a top-1 accuracy of 90.1% on the favorable HP typing and unfiltered data combination, but we still have a top-1 accuracy of 83.23% on the realistic Touch typing and Skype filtered data combination. The same tendency across the different combinations holds for every laptop. In particular, the comparison of the results of unfiltered data with Skype filtered data, reported on average in Figure 7, shows that Skype does not reduce the accuracy of *S&T attack*. This means that a keyboard acoustic eavesdropping attack over VoIP is feasible and can be considered a real-world threat.

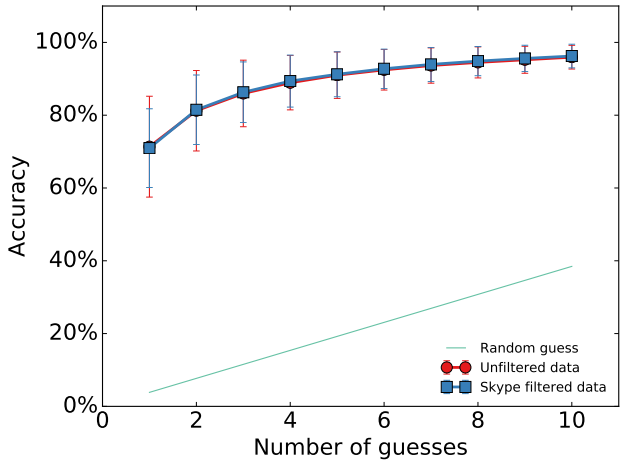
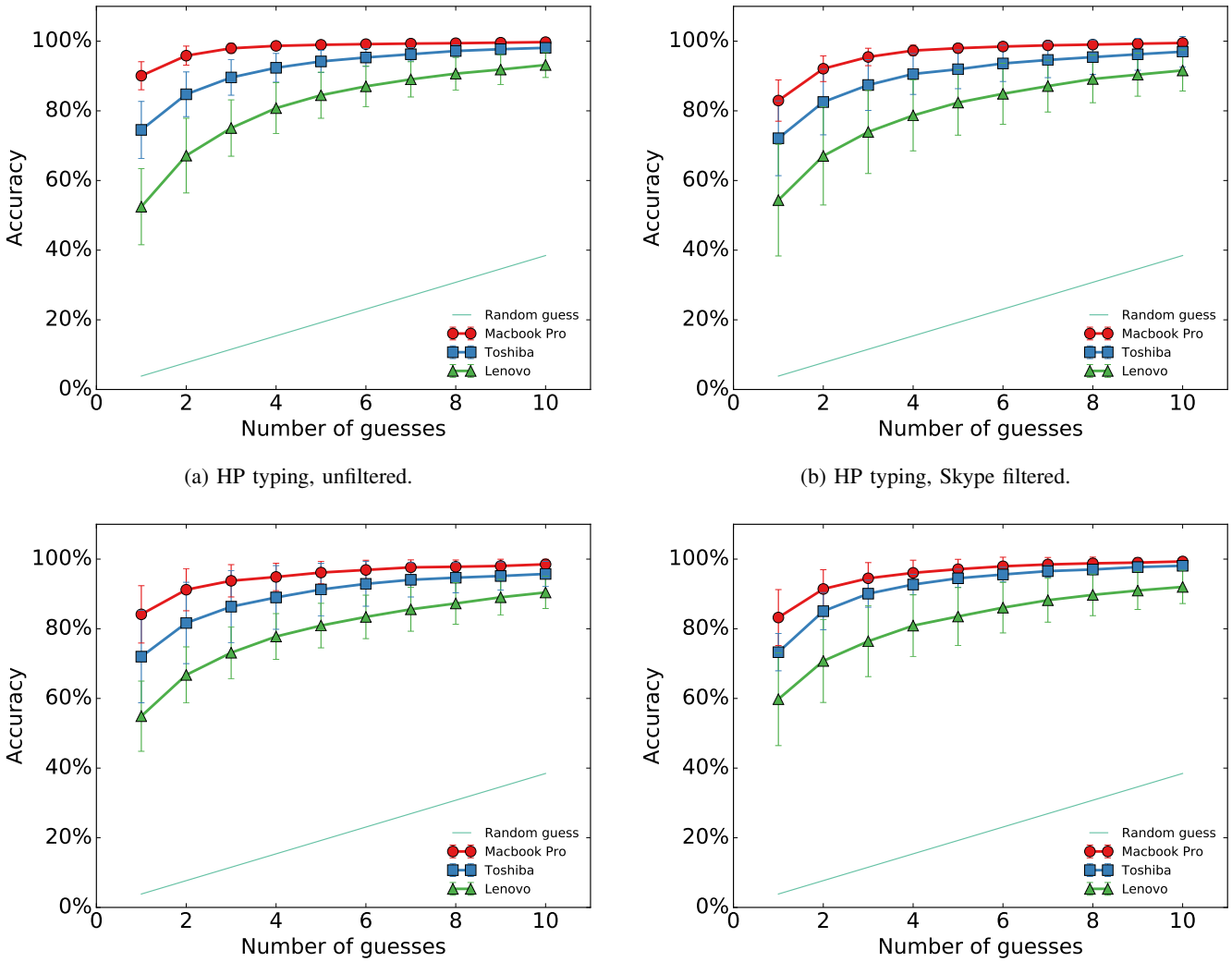


Fig. 7: *S&T attack* performance: *Complete Profiling* scenario, average accuracy of unfiltered and Skype filtered data.

From now onwards, we focus our analysis only on the most realistic combination, i.e., the Touch typing and Skype filtered data. We consider this combination to be the most realistic, because *S&T attack* will be usually carried out over Skype, and it is more common for users to type with the Touch

[5]In a stratified k-fold cross-validation scheme, the dataset is split in k subsamples of equal size, and each subsample has the same percentage of samples for every class as the complete dataset. One subsample is used as testing data, and the other $k-1$ subsamples are used as training data. The process is repeated k times, using each of the subsamples as testing data.



(a) HP typing, unfiltered.

(b) HP typing, Skype filtered.

(c) Touch typing, unfiltered.

(d) Touch typing, Skype filtered.

Fig. 6: *S&T attack* performance: *Complete Profiling* scenario, average accuracy.

typing style, rather than the HP typing style. We limit ourself to this combination to further understand only the real-world performance of *S&T attack*.

2) A More Realistic Small Training Set: As discussed in Section III-B, a possible way to set up *S&T attack* on the *Complete Profiling* scenario would exploit data accidentally disclosed by the victim, e.g., instant-messaging with the attacker during the VoIP call. However, each of the datasets we collected comprises 10 repetitions of every letter from "A" to "Z", totally 260 letters, which is a reasonably low amount, but has unrealistic letter frequencies. We therefore trained the classifier with a small subset of the training data that respects the letter frequency of the English language. To do this, we retained 10 samples of the most frequent letters according to the Oxford Dictionary [22]. Then, we randomly excluded samples of the less frequent letters until only one sample for the least frequent letters was available. Ultimately, the subset contained 105 samples, that can be a realistic short message, such as a chat message, or an email. We then evaluated the performance of the classifier trained with this subset, on a 10-fold cross-validation scheme. This random exclusion scheme was repeated 20 times for every fold. We report the results on the Touch typing Skype filtered data in Figure 8.

We observe that we suffer a loss of around 30% accuracy on every laptop, mainly because the (less frequent) letters for which we have only a few examples in the training set are harder to classify. However, the performance of the classifier are still good enough, even with such a very small training set, made of 105 samples with realistic letter frequency. This further motivates the *Complete Profiling* scenario: the attacker can exploit even a few emanations that the victim discloses with a short chat message during a Skype call.

3) User Profiling Scenario: In this case, we recall that the attacker profiled the victim on a laptop of the same model of the target laptop. Therefore, to evaluate this scenario we proceeded as follows. We selected the dataset of a particular user on one of the six laptop, and we used such dataset as

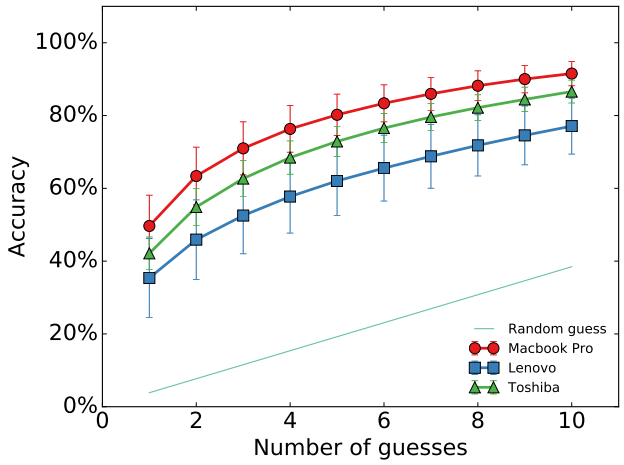


Fig. 8: *S&T attack* performance: *Complete Profiling* scenario, average accuracy, on a small subset of 105 samples that respects the letter frequency of the English language.

our training set. We recall that such dataset is formed by 260 samples, 10 for every letter of the alphabet. This training set modeled the data that the attacker acquired, e.g., with social engineering techniques. We used the dataset of the same user on the other laptop of the same model as a test set, to model the target laptop. We performed this selection for all the six laptops, and we report the results of this attack in Figure 9 for the realistic combination of Touch typing and Skype filtered data.

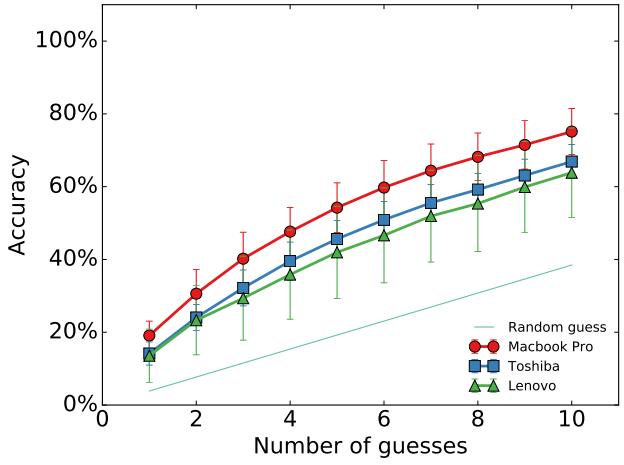


Fig. 9: *S&T attack* performance: *User Profiling* scenario, average accuracy.

We can see that the top-1 accuracy decreases to as low as 14% on the Toshiba and Lenovo laptops, and to 19% on the Macbook Pro. However, the top-5 accuracy grows up to 41.9%, 54%, and 45.6% on the Lenovo, respectively, Macbook Pro, and Toshiba laptops, restoring a good accuracy of the attack. This result shows that it is still useful to use techniques such as social engineering to obtain labeled data of the victim on a different laptop.

4) Model Profiling Scenario: We now evaluate the unfavorable, however most realistic, scenario where the attacker does not know anything about the victim. We recall that performing *S&T attack* in this scenario requires two different steps: (i) target-device classification, and (ii) key classification.

Target-device classification. The first step for the attacker is to understand whether the target-device is a known one. We suppose that the attacker collected a database of sounds of many different keyboards, used by himself or by some accomplices. When the attacker receives acoustic emanations of the keyboard of the target-device, either (i) the laptop model is present on the database, or (ii) the laptop model is not present.

If the model of the target-device is present in the database, the attacker can then use this data to train the classifier. To evaluate this scenario, we proceeded as follows. We completely excluded all the records of one user and of one specific laptop of the original dataset. We did this to create a training set where both the victim's typing style and the victim's target-device are unknown to the attacker. We also added a number of keyboards and laptops on the training set, namely an external Apple keyboard (unknown model), a Logitech Internet keyboard, a Logitech Y keyboard, an Acer E15 laptop, and a Sony Vaio Pro 2013 laptop. We added these models to show that one laptop is recognizable from its keyboard acoustic emanations among many different models. We evaluated the accuracy of the k-NN classifier in classifying the correct laptop model, on the Touch typing and Skype filtered data combination. In our results, we guess the correct laptop model 93% of the times. This experiment confirms that an attacker is indeed able to understand what laptop the victim is using, by using acoustic emanations of the keyboard disclosed through Skype.

If the model of the target-device is not present in the database, the attacker needs to understand it, as he can not conclude *S&T attack* until he obtains training data for that model. One way, to understand that the target-device is not known, is to use the confidence of the classifier. In particular, we observed that, if the target-device is present in the database, most of the samples are classified correctly (i.e., most of the samples "vote" correctly). When the target-device is not present on the database, the predicted labels for the samples are more spread among known models. A simple way to assess if the votes are spread out among the possible labels is to calculate the difference between the mean and the most voted label. We observed that trying to classify an unknown laptop leads to a lower distance of this metric (0.21 versus 0.45). The attacker can use these observations, and try to use any further information he can gather, for example social engineering techniques, laptop [12], microphone [7], or webcam [16] fingerprinting.

Key classification. Once the attacker understood what target-device the victim is using, he can move to understand the keys that the victim typed. However, the attacker does not have any data about the victim, besides the test data, to train the classifier with. He can therefore use, as a training set, the data of another user on a laptop of the same model of the target-device. We report the results of *S&T attack* in this scenario in Figure 10, for the Touch typing and Skype filtered data combination.

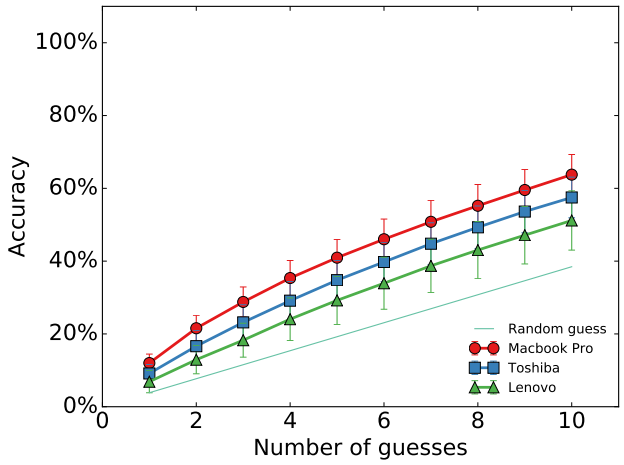


Fig. 10: *S&T attack* performance: *Model Profiling* scenario, average accuracy.

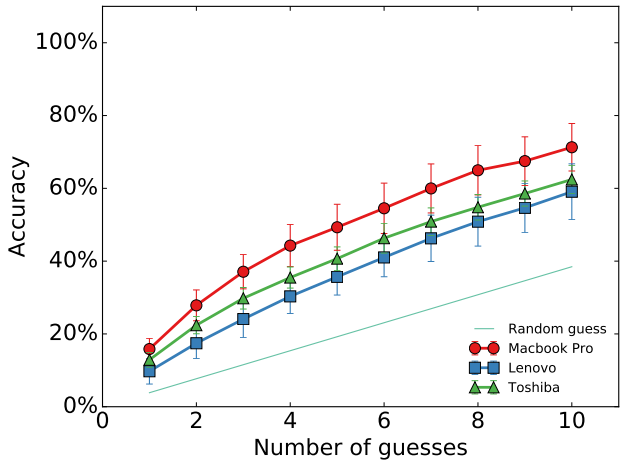


Fig. 11: *S&T attack* performance: *Model Profiling* scenario with "crowd" training data, average accuracy.

We can see that, as expected, the accuracy decreased from the previous scenarios. However, especially with the Macbook Pro datasets and the Toshiba datasets, we still have a notable advantage from a random guess baseline. Randomly guessing the labels has an top-1 accuracy of 3.84%, and a top-5 accuracy of 19.23%. Indeed, our classifier outperforms this baseline, as high as twice as better if we consider top-5 accuracy, as we report in Table I.

TABLE I: *S&T attack* performance: *Model Profiling* scenario, accuracy and improvement from the random guess baseline.

Dataset	Top-1	Improv. from random guess	Top-5	Improv. from random guess
Macbook Pro	11.99%	+312%	40.95%	+213%
Lenovo	6.87%	+178%	29.22%	+152%
Toshiba	9.14%	+237%	34.77%	+180%

To further improve these results, the attacker can use a different strategy to build his training set. We suppose that the attacker recorded multiple users on a laptop of the same model of the target-device. This can be obtained by multiple attackers, or by an attacker and some accomplices. The attacker then combines all these samples of different users to form a "crowd" training set. We evaluated this scenario as follows: we selected the dataset of one user on a given laptop, as a test set. We then created the training set by combining the data of the other users on the laptop of the same model. We repeated this experiment selecting every combination of user and laptop as test set, and the corresponding other users and laptop as a training set. Results for the the Touch typing and Skype filtered data combination are reported in Figure 11 and in Table II. We observe that the overall accuracy grows by 6-10%, and therefore this technique can be used by the attacker to further improve the detection rate of the classifier.

These results show that it is indeed still possible, in a realistic VoIP scenario, and against a target text which is both short and random, to perform *S&T attack*. Moreover, it is possible to do this with little to no specific training data of the victim, meaning that the attacker can effectively have no knowledge of the victim.

TABLE II: *S&T attack* performance: *Model Profiling* scenario with "crowd" training data, accuracy and improvement from the random guess baseline.

Dataset	Top-1	Improv. from random guess	Top-5	Improv. from random guess
Macbook Pro	15.85%	+412%	49.30%	+256%
Lenovo	9.74%	+253%	35.68%	+185%
Toshiba	12.90%	+335%	40.68%	+211%

C. VoIP-specific Issues

To conclude our experimental evaluation, we further analyze the impact of some issues that arise by using VoIP to perform *S&T attack*, and that we did not consider before. Indeed, using VoIP as the attack medium poses additional challenges to the attacker, such as the probable presence of speech on top of the sound of keystrokes, that need to be evaluated as well. Moreover, by using Skype as the VoIP software, we need to investigate whether technicalities of the SILK codec [27] degrade the performance of *S&T attack*, and to what extent. For example, the codec reduces the audible bandwidth, when the available Internet bandwidth is low, and this operation degrades the spectrum of the sound. We now analyze the impact of different Internet bandwidths on the performance of our system in Section V-C1, and the impact of the victim talking while pressing the keys of the target text (e.g., talking while typing a password) in Section V-C2.

1) The Impact Of Bandwidth: In our experimental setup, to filter the recorded data through Skype, both the receiving and sending machines were connected to a high-speed network. However, a realistic call can happen through slower network

connections. We therefore did a number of sample Skype calls between the two computers, monitoring the network load of the transmitting one. To assess the impact of bandwidth reduction on the accuracy of our classifier, we designed an experiment as follows: we filtered all the data recorded on one of the Macbook Pro laptops by all the users with the HP typing style using Skype, together with a five minutes sample of the *Harvard Sentences*, commonly used to evaluate the quality of VoIP applications [23]. We initially let the Skype software use the full bandwidth available, and we measured that the software used an average of 70 Kbit/s without any noticeable packet loss. We subsequently limited the bandwidth of the transmitting machine at 60 Kbit/s, 50 Kbit/s, 40 Kbit/s, 30 Kbit/s, respectively, 20 Kbit/s. We observed that, with values below 20 Kbit/s, the quality of the call is compromised, because of frequent disconnections. *S&T attack* with such a small bandwidth is therefore not possible, and we argue that real users suffering this degradation of service would anyway not be willing neither able to continue the Skype call. Therefore, we believe the bandwidths we selected are representative of all the conditions on which we find the Skype software is able to operate. We then evaluated both the accuracy of *S&T attack*, and the quality of the call by using the voice recognition software CMU Sphinx v5 [13] on the Harvard Sentences. We show the results in Figure 12.

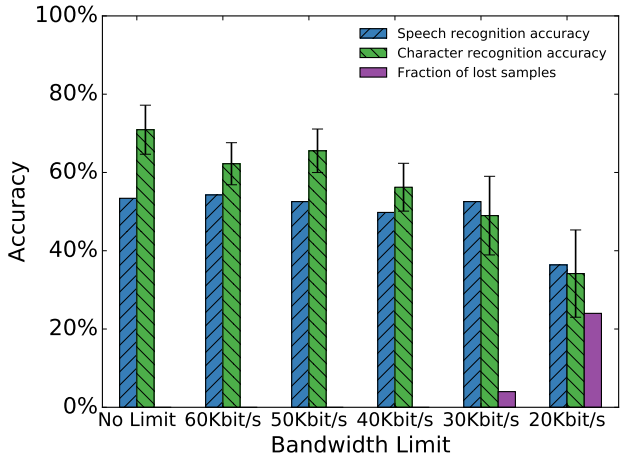


Fig. 12: Voice recognition and *S&T attack* accuracy, on data acquired through Skype with different connection bandwidths.

We can see that, while there is no change to the accuracy of the voice recognition software until the 20 Kbit/s threshold, the classifier suffers a noticeable loss at and under 40 Kbit/s. This analysis shows that aggressive downsampling, and communication errors, can greatly hinder the accuracy of the attacker on the eavesdropping task, and that a loss of the order of 20% is to be expected if the connection speed is very low. We also observe that, at 20 Kbit/s, even if the Skype call is working, many samples of both the speech and keyboard sounds are lost or irreparably damaged due to the small bandwidth, and the final quality of the call might be undesirable for the user. However, it is realistic to assume Skype to be always working at the best possible quality or almost at the best possible quality, since 70-50Kbit/s are bandwidths that are small enough to be almost guaranteed.

2) The Impact Of Voice: In the experiments we described so far, we did not consider that the victim can possibly be talking while he types the target text. However, in a VoIP call, this can happen frequently, as it is probable that the victim is talking while he types something on the keyboard of his target-device. We evaluated the impact of this scenario as follows: we considered all the data of one user on the Macbook Pro laptop, consisting of 260 samples, 10 for every class, in a 10-fold cross-validation scheme. For every fold, we performed feature selection on the train data with a Recursive Feature Elimination algorithm, and we then overlapped the test data with a random part of a recording of some Harvard Sentences with the pauses stripped out (so that the recording always has some voice in it). To account for the random overlap, we repeated the process 10 times, to have the keystroke sound overlap different random phonemes. We then evaluated the mean and standard deviation of the accuracy of the classifier.

We repeated this experiment with different relative intensities of the voice against the intensity of the sound of the keystrokes. We started at -20dB, meaning that the keystrokes are 20dB louder than the voice of the speaker, and evaluated progressive steps of 5dB, until we had the voice of the speaker 20dB louder than the keystrokes. We performed this scheme on the data for all users on the Macbook Pro laptop, with Touch typing and data filtered with Skype. We show the results in Figure 13.

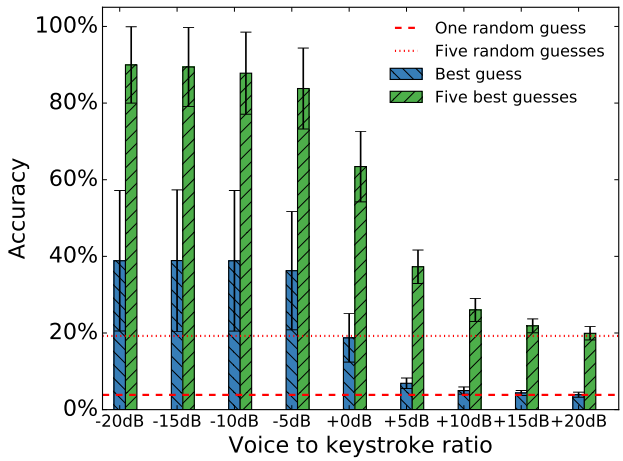


Fig. 13: *S&T attack* performance: average accuracy, overlap of keystroke sounds and voice, at different relative intensity.

We observe that, from -20dB until 0dB, the system does not suffer almost any performance loss, and then the accuracy rapidly decreases, until it reaches the random guess baseline at $+20$dB. We explain both the positive and the negative results with the phenomenon of auditory masking [31], where only the most powerful tone among all the tones at a given frequency is audible. In our case, the greater the difference between the intensity of the sound of the keystroke and of the voice, the more only the frequencies of the louder sound will be audible. However, it is realistic to assume that the speaker will talk at a reasonable volume during the Skype call. Given that the keystrokes are very loud when recorded from a laptop microphone (sometimes almost peaking the headroom of the microphone), it is unlikely that the victim will talk more than

5dB louder than a keystroke sound. These results therefore show that the victim speaking does not prevent the attacker to perform *S&T attack*.

VI. An S&T attack application: Password Cracking

We now consider a practical application of eavesdropping, which is the problem of password cracking. Password cracking is an important problem, as passwords are one of the main targets of attackers. Well known attacks that aims at guessing a password use dictionaries, lists of known passwords, or social engineering techniques.

Secure passwords that prevent dictionary attacks are random combinations of alphanumeric characters. It is possible to generate such random passwords with computers, or with initialisms (using the initial letter of many words) [33]. This last technique leads to random-looking passwords that are almost as secure as truly random ones [28, 33], but easier to memorize. To crack these secure random passwords, attackers need to use brute-force techniques that, however, require lots of time. For example, there are $(26{+}26{+}10)^{10} \approx 839$ quadrillion possible passwords composed of ten lowercase and uppercase letters, and numbers. A brute-force attack would require, on average, half of this number of tries.

In the following, we analyze how our attack can help greatly decrease the average required trials to successfully crack a password. In particular, in Section VI-3, we calculate the entropy reduction of passwords thanks to our attack, and in Section VI-4 we calculate the reduction of the average number of trials of an improved brute-force scheme that leverages our attack.

3) Entropy Reduction: It is common to assess the security of random passwords by calculating the Shannon entropy $H(X)$ [24] of the password generation process, as a measure of its unpredictability. The generation process is represented by the random variable X with n possible values $\{x_1, ..., x_n\}$ that have probability $P(x_i)$. Shannon entropy is then defined as follows:

$$H(X) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log_2 \mathrm{P}(x_i).$$

We can rewrite the formula, since we assume random passwords, composed of n characters selected with uniform probability among an alphabet A composed of $l = |A|$ symbols, as follows:

$$H(X) = log_2(l^n),$$

that give us the *bits* of entropy of the password generation process, and therefore of the password. For example, the password we considered previously, composed of ten lowercase and uppercase letters, and numbers, selected uniformly at random, has an entropy of $H(X) = log_2((26 + 26 + 10)^{10}) = 59.54$ bits.

It is possible to understand the speedup of brute-force techniques by assessing the entropy reduction of passwords thanks to our attack. We consider a password composed of 10 lowercase characters of the English alphabet, selected with uniform probability. Such a password has $log_2(26^{10}) = 47.0$ bits of entropy. On the *Complete Profiling* scenario, our attack has more than 90% accuracy within five guesses, on the Touch typing, Skype filtered dataset. This means that the entropy of the password is now $log_2(5^{10}) = 23.22$ bits with very high probability, with an entropy reduction of more than 50%. We can not calculate the entropy reduction of the password on scenarios with lower accuracy, such as the *Model Profiling* scenario, because it is highly probable that at least one set of predictions is not correct. Instead, to evaluate these scenarios, we design an improved brute-force scheme, that allows us to greatly reduce the average number of trials required to successfully crack the password.

4) Average Trials Reduction: As introduced in our motivating example, a brute-force attack on a password in which the characters are selected uniformly at random requires, on average, half of the number of trials before succeeding. However, as we know the accuracy of our attack, we can try the most probable characters before the others. For example, if our attack has a top-5 accuracy of 50%, it is reasonable to try these five guesses first, for every character. We can therefore define a brute-force strategy as follows: given the x guesses of our attack for each of the n characters, we first consider all the x^n combinations of such characters. We then assume that the set of x guesses of the first character was wrong, and subsequently consider all the other characters. When we finish considering that one set of guesses was wrong, we consider all the combinations of two wrong guesses (i.e., first and second sets of guesses were wrong, first and third sets were wrong, up to the seventh and eighth sets). We repeat this scheme until we finally try the combinations where the classifier was always wrong. This brute-force scheme leverages the probability of success of our attack to minimize, on average, the required time to crack a password.

Since this scheme takes into account the fact that we only have a probability for the set of guesses to be correct, we can use it to assess the speedup that our attack gives to brute-force. In particular, we can calculate how many tries we require to have 50% probability of guessing the password, which is the average number of guesses the brute-force needs to succeed. We consider again a password composed of ten lowercase and uppercase letters, and numbers.

The baseline brute-force attack that we previously described requires $\frac{(26)^{10}}{2} = 8.39 \cdot 10^{13}$ guesses to have 50% probability. On the *Complete Profiling* scenario, that we recall has an average top-5 accuracy of more than 90%, we only need $9.76 \cdot 10^6$ tries to have 50% probability. This corresponds to a very high average speedup of 10^7. On the *Model Profiling* scenario, where we have a top-5 accuracy around 40%, we need $7.79 \cdot 10^{12}$ tries to reach 50% probability of cracking the password, which is still one order of magnitude better than plain brute-force attacks, on average. There is similar tendency if the attack guesses ten characters for every character of the password. We show the cumulative distribution function of the probability of success for the brute-force, and for our enhanced scheme in the *Model Profiling* scenario, in Figure 14. These results show that our attack is, indeed, able to greatly speed up brute-force attacks, either thanks to its high accuracy, or thanks to the probability strategy we introduced.

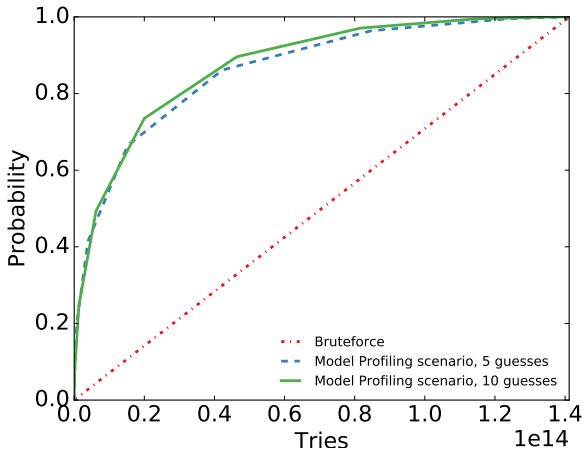


Fig. 14: Cumulative distribution function of the probability of cracking the password, for regular brute-force, and for our scheme.

VII. Possible Countermeasures

The evaluation of our attack on real data showed that keyboard acoustic eavesdropping through VoIP applications is a real threat. In this section, we briefly discuss some possible countermeasures, and analyze their effectiveness in preventing our attack, and other attacks that leverage statistical properties of the spectrum of the sound.

A simple countermeasure to our attack could be a short "ducking" effect, a technique where we greatly reduce the volume of the microphone and overlap it with a different sound, when a keystroke is detected. However, this could ruin the quality of the voice call, as the voice is removed in its entirety as well. An effective countermeasure should be less intrusive as possible, and disrupt only the sound of the keystrokes, avoiding to ruin the call of the user.

To build a less intrusive countermeasure, which could potentially prevent all of the techniques that leverage spectrum information, we can apply short random transformations to the sound when we detect a keystroke. A convenient way to do it is to apply a random multi-band equalizer over a number of small frequency bands of the spectrum. This technique allows us to modify the intensity of specific frequency ranges, called "bands". Each band should be selected at random, and its intensity should be modified by a random small amount, thus effectively modifying the spectrum of the sound. Moreover, with this approach, the voice of the speaker should be still intelligible.

To show the efficacy of this countermeasure, we designed an experiment as follows: we considered all the data recorded on the Macbook Pro laptop, one user at a time, in a 10-fold cross-validation scheme. For every fold, we applied a multiband equalizer with 100 bands to the test data only, where each band has a random center between 100 Hz and 3000 Hz, a very high resonance Q of 50, and a random gain between -5dB and +5dB. We then tried to classify these samples. We did this using both MFCC and FFT features, to see if such countermeasure could prove effective even against different spectral features. We report the results of this experiment in Figure 15, where we show the accuracy of the attack both without and with the countermeasure, for MFCC and FFT features.

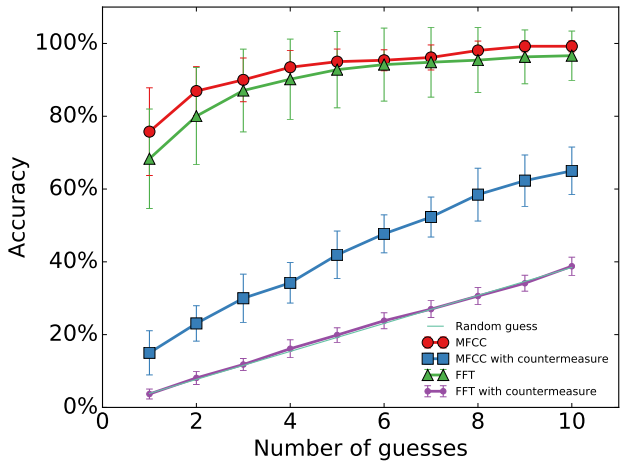


Fig. 15: Average accuracy of single key classification against a random equalization countermeasure.

We observe that our countermeasure successfully disrupts FFT coefficients, such as the ones used in [3, 10, 11, 18], by reducing the accuracy of the attack to the baseline random guess. For the MFCC features that we use in our attack, however, the countermeasure still manages to reduce the accuracy by 50% on average, but the features prove to be partially robust to this tampering.

Ultimately, the most effective and unobtrusive solution to the attack is prevention. It is therefore better not to *Skype and Type* or, at least, not to type sensitive information, such as passwords or confidential emails.

VIII. Conclusions and Future Work

In this paper, we designed a highly accurate VoIP keyboard acoustic eavesdropping attack. We first described a number of realistic attack scenarios, using VoIP as a novel means to acquire acoustic information, and with realistic assumptions, such as random target text and very small training sets in Section III. We then designed an attack that considered all of these real-world limitations that an attacker could have, and carefully selected the tools of the trade to maximize the accuracy of the attack in Section IV. We thoroughly evaluated our attack, using the VoIP software Skype, on the different scenarios in terms of accuracy of character recognition in Section V. We studied a practical application of *S&T attack*, the problem of password cracking from an attacker, in Section VI. We finally discussed some possible countermeasures to our attack, and to other attacks that leverage spectral features of the sound of keyboards, in Section VII.

We believe that our work is an important contribution to keyboard acoustic eavesdropping, because of the real-world applicability of our attack. Our attack proves to be feasible and accurate over the VoIP software Skype, on all the attack scenarios we considered, with minimal or no profiling of the

victim's typing style and keyboard. In particular, it is accurate on our *Model Profiling* scenario, where the attacker profiles a laptop of the same model of the laptop of the victim, but has no information or data about the victim. This allows an attacker to effectively steal information from an unknown victim. If the goal for the attacker is to eavesdrop a random password, we showed how our enhanced bruteforce scheme, where we first try the letters guessed by the attack, reduces the average number of tries by 12 orders of magnitude on the most favorable attack scenario, and by one order of magnitude on the most difficult attack scenario. Moreover, we always considered the most realistic scenarios and assumptions, that are unfavorable for the attacker. Therefore, all of our results would increase with more favorable conditions, for example if the target text is in a known language, or if it is predictably random (such as common user passwords). We also considered VoIP-specific issues of our attack, such as the impact of the audible bandwidth reduction operated by VoIP software in presence of low Internet bandwidth, and the problem of the victim speaking on top of the sound of the keystrokes. We showed how our attack is robust to bandwidth reduction, and to the presence of other sounds, such as voice. We finally discussed some countermeasures, and observed that our attack is indeed hard to counter, and is more easily prevented by avoiding to type sensitive data during VoIP calls.

A. Future Work

We believe that our choice of laptops and test users is a representative sample. The number of tested laptops was in line with related work, and the number of users was greater, as related work always collected data only on one user [3, 10, 11, 18]. However, it would be useful to run the experiments on more laptop models, and with more users, to further confirm that our attack works regardless of the different construction materials, and of different typing styles. Moreover, we consider Skype to be representative of most VoIP software, but we aim to evaluate the attack over different software, for example Google Hangouts. We also plan to test *S&T attack* against meaningful target text, such as English text, and to use dictionaries and crowdsourced approaches (e.g., Google Instant), to correct detection errors and improve the accuracy of *S&T attack*.

We finally aim to take a further look at the possible countermeasures, to analyze the actual real-time feasibility of random equalization in the presence of the sound of keystrokes, evaluate its impact on the perceived quality of the call by the user, and to improve its performance.

References

[1] *2015: Skype's year in review*. URL: http://blogs.skype.com/2015/12/17/2015-skypes-year-in-review/ (visited on 06/29/2016).

[2] Kamran Ali et al. "Keystroke recognition using WiFi signals". In: *Annual International Conference on Mobile Computing and Networking*. ACM. 2015, pp. 90–102.

[3] Dmitri Asonov and Rakesh Agrawal. "Keyboard acoustic emanations". In: *Symposium on Security and Privacy*. IEEE. 2004, pp. 3–11.

[4] Davide Balzarotti, Marco Cova, and Giovanni Vigna. "Clearshot: Eavesdropping on keyboard input from video". In: *Symposium on Security and Privacy*. IEEE. 2008, pp. 170–183.

[5] Yigael Berger, Avishai Wool, and Arie Yeredor. "Dictionary attacks using keyboard acoustic emanations". In: *Conference on Computer and Communications Security*. ACM. 2006, pp. 245–254.

[6] Stephen Boyd et al. "Accuracy at the top". In: *Advances in Neural Information Processing Systems*. NIPS. 2012, pp. 953–961.

[7] Anupam Das, Nikita Borisov, and Matthew Caesar. "Do you hear what I hear?: fingerprinting smart devices through embedded acoustic components". In: *SIGSAC Conference on Computer and Communications Security*. ACM. 2014, pp. 441–452.

[8] Jeffrey Friedman. "Tempest: A signal problem". In: *NSA Cryptologic Spectrum* (1972).

[9] Isabelle Guyon et al. "Gene selection for cancer classification using support vector machines". In: *Machine learning* 1-3 (2002), pp. 389–422.

[10] Tzipora Halevi and Nitesh Saxena. "A closer look at keyboard acoustic emanations: random passwords, typing styles and decoding techniques". In: *Symposium on Information, Computer and Communications Security*. ACM. 2012, pp. 89–90.

[11] Tzipora Halevi and Nitesh Saxena. "Keyboard acoustic side channel attacks: exploring realistic and security-sensitive scenarios". In: *International Journal of Information Security* 5 (2015), pp. 443–456.

[12] Tadayoshi Kohno, Andre Broido, and Kimberly C Claffy. "Remote physical device fingerprinting". In: *TDSC* 2 (2005), pp. 93–108.

[13] Paul Lamere et al. "The CMU SPHINX-4 speech recognition system". In: *International Conference on Acoustics, Speech and Signal Processing*. IEEE. 2003, pp. 2–5.

[14] Jian Liu et al. "Snooping keystrokes with mm-level audio ranging on a single phone". In: *Annual International Conference on Mobile Computing and Networking*. ACM. 2015, pp. 142–154.

[15] Beth Logan et al. "Mel Frequency Cepstral Coefficients for Music Modeling." In: *International Symposium on Music Information Retrieval*. ISMIR. 2000.

[16] Jan Lukas, Jessica Fridrich, and Miroslav Goljan. "Digital camera identification from sensor pattern noise". In: *Transactions on Information Forensics and Security* 2 (2006), pp. 205–214.

[17] Philip Marquardt et al. "(sp) iPhone: decoding vibrations from nearby keyboards using mobile phone accelerometers". In: *Conference on Computer and Communications Security*. ACM. 2011, pp. 551–562.

[18] Zdenek Martinasek, Vlastimil Clupek, and Krisztina Trasy. "Acoustic attack on keyboard using spectrogram and neural network". In: *Conference on Telecommunications and Signal Processing*. IEEE. 2015, pp. 637–641.

[19] *Microsoft BUILD 2016 Keynote*. URL: https://channel9.msdn.com/Events/Build/2016/KEY01 (visited on 06/29/2016).

[20] *Opus Codec Support*. URL: https://wiki.xiph.org/OpusSupport (visited on 07/19/2016).

[21] *Over 1 billion Skype mobile downloads*. URL: http://blogs.skype.com/2016/04/28/over-1-billion-skype-mobile-downloads-thank-you/ (visited on 06/29/2016).

[22] *Oxford Dictionary - Which letters in the alphabet are used most often*. URL: http://www.oxforddictionaries.com/words/which-letters-are-used-most (visited on 06/29/2016).

[23] EH Rothauser et al. "IEEE recommended practice for speech quality measurements". In: *Transactions on Audio and Electroacoustics* 3 (1969), pp. 225–246.

[24] Claude Elwood Shannon. "A mathematical theory of communication". In: *SIGMOBILE Mobile Computing and Communications Review* 1 (2001), pp. 3–55.

[25] Diksha Shukla et al. "Beware, your hands reveal your secrets!" In: *SIGSAC Conference on Computer and Communications Security*. ACM. 2014, pp. 904–917.

[26] Dawn Xiaodong Song, David Wagner, and Xuqing Tian. "Timing Analysis of Keystrokes and Timing Attacks on SSH." In: *Security Symposium*. USENIX. 2001.

[27] Jean-Marc Valin, Koen Vos, and T Terriberry. "Definition of the Opus audio codec". In: *IETF, September* (2012).

[28] Kim-Phuong L Vu et al. "Improving password security and memorability to protect personal and organizational information". In: *International Journal of Human-Computer Studies* 8 (2007), pp. 744–757.

[29] Martin Vuagnoux and Sylvain Pasini. "Compromising Electromagnetic Emanations of Wired and Wireless Keyboards." In: *Security Symposium*. USENIX. 2009, pp. 1–16.

[30] Junjue Wang et al. "Ubiquitous keyboard for small mobile devices: harnessing multipath fading for fine-grained keystroke localization". In: *Annual International Conference on Mobile systems, applications, and services*. ACM. 2014, pp. 14–27.

[31] RLf Wegel and CE Lane. "The auditory masking of one pure tone by another and its probable relation to the dynamics of the inner ear". In: *Physical review* 2 (1924), p. 266.

[32] Teng Wei et al. "Acoustic eavesdropping through wireless vibrometry". In: *Annual International Conference on Mobile Computing and Networking*. ACM. 2015, pp. 130–141.

[33] Jeff Jianxin Yan et al. "Password Memorability and Security: Empirical Results." In: *Symposium on Security and Privacy* 5 (2004), pp. 25–31.

[34] Kehuan Zhang and XiaoFeng Wang. "Peeping tom in the neighborhood: Keystroke eavesdropping on multi-user systems". In: *Security Symposium*. USENIX. 2009.

[35] Tong Zhu et al. "Context-free attacks using keyboard acoustic emanations". In: *SIGSAC Conference on Computer and Communications Security*. ACM. 2014, pp. 453–464.

[36] Li Zhuang, Feng Zhou, and J Doug Tygar. "Keyboard acoustic emanations revisited". In: *Transactions on Information and System Security* 1 (2009), p. 3.