# YELP: Masking Sound-based Opportunistic Attacks in Zero-Effort Deauthentication

Prakash Shrestha
University of Alabama at Birmingham
prakashs@uab.edu

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

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

## ABSTRACT

Deauthentication is an important component of any computing system that promises to offer legitimate access to restricted services residing on the system. As computing devices are ubiquitous, it has underscored the need to design *zero-effort* deauthentication systems from a usability perspective. While the design of such deauthentication systems is geared towards making them more usable, often the security implication of these deigns overlook the physical security of the system resulting in various side channel vulnerabilities in the system. This issue highlights the need to design a defense mechanism that is capable of minimizing the threat posed by such side channel attacks while having minimal impact on the design of the system.

In this paper, we aim to address the sound-based vulnerability, recently introduced in the literature, against one of the prominent zero-effort deauthentication schemes, called ZEBRA, that transparently and continuously authenticates the user using a wearable device wirelessly connected with the authentication terminal. To this end, we propose *YELP*, a novel and practical defense mechanism based on the principle of sound masking. YELP uses two different types of masking sounds, namely *"white noise"*, and *"music"* for cloaking the acoustic side channel leakage underlying the ZEBRA system. We believe that the use of such masking sounds at a reasonable volume level can hide the acoustic leakage emanating from the physical component of the system, and thereby reduce, if not eliminate, the imposed sound-based vulnerability. Indeed, our results show that *white noise*, as a masking sound, can effectively hide the acoustic leakage from ZEBRA system, thereby significantly reducing the attack success rate of an audio-based side channel attacker while *music* can moderately hide the acoustic leakage from the system. Our work therefore shows that sound masking can be used as an effective tool in improving the security of (de)authentication systems against an audio-based side channel attack without affecting its original design and without requiring additional effort from the user.

**ACM Reference format:**
Prakash Shrestha, S Abhishek Anand, and Nitesh Saxena. 2017. YELP: Masking Sound-based Opportunistic Attacks in Zero-Effort Deauthentication. In *Proceedings of WiSec '17 , Boston, MA, USA, July 18-20, 2017,* 12 pages.
DOI: 10.1145/3098243.3098259


*WiSec '17 , Boston, MA, USA*

DOI: 10.1145/3098243.3098259

## 1 INTRODUCTION

User authentication is an essential security functionality for most computing paradigms. An important component of an authentication system is *deauthentication*, i.e., promptly detecting when to log out a previously authenticated user from an ongoing session. A promising approach to improving the usability of (de)authentication mechanisms is to make them *transparent* to users by reducing, if not eliminating, the cognitive effort required from users. Although such *zero-effort* authentication schemes are compelling, designing them correctly can be a challenge in practice.

A representative scheme in this direction is ZEBRA, a zero-effort bilateral deauthentication method, proposed by Mare et al. [33]. ZEBRA is intended for scenarios where users authenticate to computer terminals (such as desktop computers in a shared setting). In such scenarios, users typically have to either manually deauthenticate themselves by logging out or locking the terminal, or the terminal can deauthenticate a user automatically after a sufficiently long period of inactivity. The former approach requires explicit user effort while the latter approach reduces promptness of log out. The ZEBRA method aims to make the process of deauthentication both prompt and transparent: once a user is authenticated to a terminal (using say a password), it continuously, yet transparently re-authenticates the user so that prompt deauthentication is possible without explicit user action. In ZEBRA, the user is required to wear a bracelet (or a smartwatch) equipped with motion sensors on his mouse-holding hand. The bracelet is wirelessly connected and pre-paired to the terminal, which compares the sequence of events it observes (e.g., keyboard/mouse interactions) with the sequence of events inferred using measurements from the bracelet's motion sensors. The logged-in user is deauthenticated when the two sequences no longer match.

ZEBRA is particularly appealing due to its simplicity of design. However, as shown in a recent work by Huhta et al. [25], this simplicity gives rise to a design assumption that an adversary can exploit to defeat the security of the scheme. In particular, Huhta et al. identified a design flaw in ZEBRA that allows to develop an effective attack strategy, whereby a human attacker observing/listening to a victim at a nearby terminal and *opportunistically* mimicking only a subset of the victim's activities (e.g., only keyboard events) at the authentication terminal. For example, the human attacker can simply listen onto the sounds of the keyboard typing of the victim on another computer terminal and mimic the keystroke events at the authentication terminal. The attack can be used to effectively undermine the security offered by ZEBRA in that the attacker can remain logged in for a relatively long duration of time, during which it can perform malicious activities on the terminal (such as sending emails on behalf of the victim, or changing dosages and writing new prescriptions in a hospital setting.). Depending upon

the application scenarios for ZEBRA, the consequences of such an attack can be devastating.

Given the severity of this threat against the otherwise practical ZEBRA system, we set out to design an effective defensive approach that would work *transparently* with ZEBRA, i.e., without necessitating any design changes to ZEBRA. Since the *visual observation attacks* can be relatively easily addressed by the use of visual barriers around the login terminals that will block the attacker from gaining a clear view of the victim when using another terminal, our focus in this work is on *sound-based observation attacks* which exploit the sounds of the keyboard typing [45] and are challenging to address. Our general idea to resist such sound-based opportunistic attacks against ZEBRA is to utilize the notion of *sound masking* – the login terminals or a device placed in the surrounding environment produces deliberate sounds that will mask the sounds of the keyboards, thereby making it difficult for the attacker to mimic the victim user's activities.

Sound masking itself is already being used in many contexts. It has traditionally been used as a commercial solution ([4, 12, 31, 32, 42]) for providing speech privacy in call centers, offices, medical facilities, law and government facilities, etc. It has been used as a way to reduce distraction, improve focus/productivity and protect sensitive conversations. It is touted as a low cost measure to achieve speech privacy as compared to potentially expensive architectural improvements in the environment. Our approach to bolster the security of ZEBRA against sound-based opportunistic attacks based on sound masking is well-aligned with these traditional solutions and can in fact seamlessly work along with them as a broader solution for authentication/deauthentication security, speech privacy and productivity improvement.

**Our Contributions:** Our primary contributions can be summarized as follows:

(1) ***Securing Deauthentication with Sound Masking***: We propose a novel defense, called YELP[1] (Section 3.1), to known audio-based opportunistic attacks against a prominent, representative deauthentication system, ZEBRA, based on the simple idea of sound masking. YELP works transparently with ZEBRA without requiring any changes to its design and still maintaining its zero-effort property.

(2) ***Designing YELP with Two Types of Masking Sounds***: We design YELP based on two types of masking sounds, white noise produced by the computer terminals themselves and musical sounds produced by a central device present in the environment where the ZEBRA system is being deployed (Section 3.3). While such masking sounds have already been used in real-world scenarios for improving speech privacy, relaxation and productivity, we argue that they may also be effective in improving the security of authentication/deauthentication systems with little to no added cost to the system.

(3) ***Evaluating YELP for Security and Performance***: We evaluate the performance and security of YELP based on the two types of masking sound choices. For the purpose of our evaluation, we first recreate the ZEBRA system (Section 4) as documented by Mare et al. [33]. Based on experiments with human users, we further reproduce the audio-based opportunistic attack on ZEBRA as proposed by Huhta et al. [25], testing its performance (Section 5) against both ZEBRA and YELP. Our results show that YELP can effectively improve the security of ZEBRA without a significant impact on its performance in the benign settings. In particular, we show that by using YELP with *white-noise*, ZEBRA was able to kick out 70% of the attackers as opposed to 22% without YELP within a small number of interactions (as defined by ZEBRA). In addition, YELP with *music* kicked out 60% of the attackers for the same number of interactions.

**Paper Roadmap:** The rest of the paper is structured as follows. In Section 2, we present a review on ZEBRA and opportunistic attacks against ZEBRA. In Section 3, we detail our defense approach followed by Section 4, where we report on our reimplementation of the ZEBRA. In Section 5, we explain our experiment setup, and present the evaluation of the effectiveness of YELP against *audio-only opportunistic keyboard-only* attackers. Finally, we review prior works relevant to our study in Section 6, and conclude and point to future research items in Section 7.

## 2 BACKGROUND

### 2.1 ZEBRA Review

ZEBRA is intended for the scenarios that have multiple terminals. In these scenarios, users often move between terminals. In Mare et al. [33], hospital scenario is presented as their motivating scenario. Hospital environment often has shared terminals that are used by hospital staff. Regardless of shared terminals, a user/staff should not, intentionally or unintentionally, access the terminals where other user has logged in. Users may leave terminals without logging out, but may still remain in close locality. In such scenarios, proximity-based zero-effort deauthentication schemes like ZIA [15] or BlueProximity cannot be used because these methods are not accurate enough for short distances. Although ZEBRA is intended for the scenarios with shared terminals, transparent deauthentication schemes like ZEBRA are broadly applicable to any scenario where users may leave their terminals unattended.

ZEBRA [33] is representative of continuous authentication schemes designed for the scenarios where users authenticate to the terminals (desktop PCs). Users, once authenticated, are required to either manually deauthenticate and lock the terminal, or be automatically deauthenticated by the system after a sufficiently long period of inactivity. The former approach requires user interaction which reduces the usability while latter approach lacks the promptness thereby sacrificing security. ZEBRA intends to make the deauthentication process both prompt and transparent. Once user is authenticated, it will continuously, yet transparently, reauthenticate the user making the user deauthentication process prompt without requiring any explicit user interaction.

*2.1.1 Threat Model:* ZEBRA is designed with the intent to prevent the threat of unauthorized access to a terminal when a user steps away from the terminal without logging out and remains in

[1]Yelp denotes the noises produced by zebras. Our solution gives rise to an improved ZEBRA system that produces masking sounds to bolster its security.

the vicinity of the terminal. Zebra considers two types of adversaries: (1)"innocent", and (2) "malicious". Innocent adversary is an authorized user who uses an unattended terminal for her own purposes either without realizing that another user ("victim") is already logged-in or because she does not want to go through the login step. On the other hand, malicious adversary uses the unattended terminal with the intention of executing some actions on behalf of the victim. Malicious adversary may observe the actions and behavior of the victim. He can fool the authenticating system by mimicking the hand movement of the victim and may authenticate himself as the originally logged-in user.

*2.1.2 System Architecture:* ZEBRA correlates the user activities observed at the terminal with the motion sensor measurements of wrist activities of the user captured by a wrist-worn device. For simplicity, we call the wrist-worn device, a *"bracelet"*, but it can be a general purpose smartwatch. We use a normal smartwatch (LG G watch R) in our implementation and analysis similar to the implementation of [25]. The goal of ZEBRA is to continuously and transparently verify whether the user accessing the terminal is indeed the one who is originally logged in and instantly deauthenticate the unintended user. ZEBRA assumes a computer/terminal with keyboard and mouse, and a bracelet, personal to each individual using system. The bracelet is equipped with motion sensors (e.g., accelerometer, gyroscope) that record the wrist movement of the wearer. The bracelet and the terminal are connected through a wireless channel such as Bluetooth that is used for securely interacting with each other. The terminal keeps record of the bracelet associated with each authorized users. When a user authenticates to the terminal, it connects to the bracelet and starts receiving the sensor data from it. ZEBRA compares the sequence of user interactions observed on the terminal with the sequence of user interaction inferred from motion sensor data of bracelet. If these two sequences do not match, ZEBRA deauthenticates the user.

The main components of ZEBRA are *Interaction Extractor, Segmenter, Feature Extractor, Interaction Classifier, and Authenticator.* ZEBRA considers three types of interaction: typing, scrolling, and hand movement from keyboard to mouse and vice-versa (termed as "MKKM"). An *Interaction Extractor* identifies the actual sequence of interaction based on the keyboard/mouse events observed on the terminal. It also logs the timestamps of each type of interaction in actual interaction sequence. *Segmenter* receives timestamps of each interaction from *Interaction Extractor*, in addition to sensor data from bracelet. It segments the sensor data into blocks based on the timestamps from *Interaction Extractor*. Note that the *Segmenter* considers only the sensor data that falls inside these time slots. From each block received from *Segmenter*, *Feature Extractor* extracts salient features, and supplies them to *Interaction Classifier* that has been trained to infer the type of interaction based on the sensor data. The *Interaction Classifier* infers the type of interaction based on the features from each block and outputs the predicted interaction sequence. Finally, an *Authenticator* compares the actual interaction sequence from *Interaction Extractor* and the predicted interaction sequence from *Interaction Classifier* and determines if the current user is same as, or different from, logged in user.

Authenticator can be tuned through window size($w$), threshold($m$) and grace period($g$). Authenticator compares a window, formed by a sequence of $w$ interactions, at a time. In a window, if the percentage of matching interactions exceeds threshold $m$, the window is marked 1, otherwise it is marked 0. If *Authenticator* marks 0 for $g$ consecutive windows, it outputs "differentfifi and instantly deauthenticate the user.

## 2.2 Opportunistic Attacks on ZEBRA

The primary goal of ZEBRA is to deauthenticate the unintended user or malicious entity promptly and transparently from accessing the terminal. In order to compromise the security of the ZEBRA system, an attacker would need to observe and imitate the behavior and actions of the victim. To evaluate ZEBRA against such an attacker, Mare et al. [33] have considered a "malicious" adversary who accesses the original terminal of the victim while victim is using another nearby terminal. The objective of the malicious adversary is to mimic all of the user's hand movements as close as possible. In ZEBRA [33] experiment, ordinary non-expert users play the role of a malicious adversary. In order to make the scenarios advantageous to attacker, they were given visual as well as verbal cues to indicate what victim was doing. They demonstrated that their system was able to deauthenticate such attackers in reasonable time, while maintaining low false negative rates.

Since underlying techniques rely on observable hand movements, ZEBRA is vulnerable to more effective impersonation attacks. In [25], an opportunistic strategy based on impersonation attack was proposed that could compromise the security of ZEBRA scheme. In the proposed attack strategy, a human attacker observes victim's activities at the nearby terminal and opportunistically mimics only subset of victim's activities (e.g., keyboard activities) at victim's original terminal. With this strategy, authors have shown that the opportunistic attacker have a high probability to break the scheme.

Based on the observation channel and attack strategy, Huhta et al. [25] have considered four different types of attacks, namely *naive all-activity attack*, *opportunistic keyboard-only attack*, *opportunistic all-activity attack*, and *audio-only opportunistic keyboard-only attack*. In first three attacks, attacker can see as well as hear the user interacting with the terminal while in audio-only attack, attacker can only hear but not see the user interactions. In *naive all-activity attack* and *opportunistic all-activity attack*, attacker tries to mimic all activities of the victim while in *opportunistic keyboard-only* and *audio-only opportunistic keyboard-only attack*, attacker tries to mimic only a subset of typing interactions of the victim.

The attack scenarios where attacker can see the victim interaction can be easily defeated by creating a visual barrier around the login terminals. However, the system is still vulnerable to audio-only attack because audio emanation due to the user interaction with terminal can still pass through the visual barrier. So, to defeat audio-based attack, audio signals emanated from keyboard-mouse interaction should be muddled. In this paper, we propose a scheme that can prevent audio-only attack (depicted in Figure 1) by masking the sounds generated during the user-terminal interactions.

# 3 OUR DEFENSE: A DESIGN-INDEPENDENT APPROACH BASED ON SOUND MASKING

As detailed in Section 2.2, in an *audio-only opportunistic keyboard-only* attack, the attacker is only able to hear victim's interactions but is barred from receiving any visual cues related to victim's

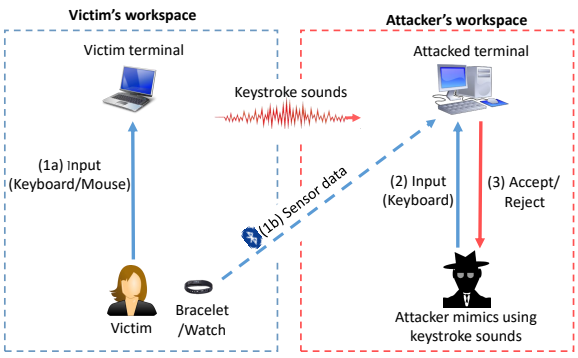



**Figure 1: Audio-only opportunistic keyboard-only attack.**

interactions. Specifically, attacker is tuned to listen only for keyboard interactions meaning attacker starts typing when he hears keystrokes emanating from victim's keyboard and stops typing as soon as the keystroke emanations from victim's keyboard stop. We classify this type of attack as an audio based side-channel attack on ZEBRA scheme.

In [25] and ZEBRA [33], the countermeasures against an adversary revolved around creating a visual barrier that could potentially block the adversary from mimicking the victim's interaction with the aid of visual cues. However, in an *audio-only opportunistic keyboard-only* attack, the attacker can mimic the keyboard interactions of the victim by just listening and does not require any visual cues about victim's interactions. A visual barrier does not prevent such an attacker from eavesdropping on keystroke emanations generated from victim's keyboard interactions.

For defending ZEBRA authentication scheme against an *audio-only opportunistic keyboard-only* attack, we need to design a defense mechanism that can potentially thwart this class of attack while having a minimal impact on the design of the original ZEBRA scheme. Thus we put forth the design principles that should be followed by such a defense mechanism to be effective while retaining usability:

(1) **Effectiveness:** The proposed design system should be able to reduce the success rate of an *audio-only opportunistic keyboard-only* attacker to an extent that the attack success rate is close enough to that of a random attack, whereby the attacker does not get any audio (and visual) cues to listen onto the keystroke sounds and attempts to mimic the victim's interactions arbitrarily.
(2) **Design Independence:** It should have minimal to no effect on the original design of ZEBRA system working transparently with the existing scheme.
(3) **Zero-effort:** There should be no user interaction required at any time for the defense mechanism to be operational.

In this work, we come up with YELP, a defense mechanism based on sound masking that tries to remain true to the design principles that we laid down for an effective and practical defense against *audio-only opportunistic keyboard-only* attacker.

## 3.1 Choice for Defense Mechanism

As stated earlier, we need countermeasure against an audio-based side-channel attack on ZEBRA scheme. Since the principle source of side channel emanation relevant to an *audio-only opportunistic keyboard-only* attacker is the keyboard, the problem of designing a competent defense mechanism involve eliminating or hiding the keystroke emanations generated from the keyboard. Several countermeasures have been proposed for handling unwanted audio leakage from systems that constitute a security concern especially in the light of audio based side-channel attacks. We divide these countermeasures into two classes based on their way of handling acoustic side channel leakage: *acoustic leakage elimination* (active noise control) and *acoustic leakage masking* (passive noise control).

**Acoustic Leakage Elimination:** Elimination of unwanted acoustic emanations can be done either physically or programatically. Softer keyboards that produce almost no sound when the keys are pressed have been proposed to eliminate acoustic leakage. However, such keyboards are costly to produce and their behavior after prolonged usage is unknown. Since not all keys on a keyboard are used frequently, over time more frequently used keys may start demonstrating different behavior from the less frequently used keys. Touchpads do not have mechanical components as keys and present as another option that could replace the traditional keyboards. Another method involves creating an audio barrier similar to visual barrier as proposed in ZEBRA scheme as a countermeasure to shield acoustic leakage. The acoustic leaking system could be contained inside a specially designed construct that absorbs audio leakage. However, this method would require placing every terminal under such a construct and this exercise could be costly and impractical.

Acoustic leakage can also be eliminated by noise cancellation techniques implemented in the software. Software implementation has the advantage of being less costly and easier to implement on multiple devices. In noise cancellation, an *anti-noise* signal is generated, that is used to cancel out the generated noise. An *anti-noise* signal is calculated by estimating the inverse of the noise that is to be canceled. Roy et al. [40] proposed a mechanism for noise cancellation in real time that involved calculating fast fourier transformation (FFT) of the noise signal and picking the k-strongest frequencies based on their FFT values. These values are then combined and reversed in sign to build an estimated inverse of the noise signal. This estimated inverse signal is then phase aligned with the actual noise signal to produce the *anti-noise* signal.

Similar technology is used in noise cancellation headphones that utilize an extra microphone to measure the noise and generate the *anti-noise* waveform that is mixed using analog technology with the original sound to deliver the *noise-free* sound to the user. This technique however requires extra hardware in the form of a supplementary microphone and analog technology for mixing the waveforms hard-wired in the circuit. For implementation in ZEBRA scheme, it would require additional cost to supplement all the terminals with the necessary hardware and circuit modifications.

**Acoustic Leakage Masking:** This approach is aimed at cloaking the emanation by producing a signal that effectively masks the intended acoustic emanation making it difficult to identify it. For effective masking, the masking signal should cover the frequency range of the signal to be masked and should be at least as loud as the original signal. Thus the properties of the masking signal depend upon the frequency and amplitude characteristics of the signal that needs to be masked.

The use of sound masking is commercially wide spread in offices, medical facilities, meeting rooms and military facilities. Most of these applications of sound masking aim to achieve speech privacy and lower distraction level in a specified environment. Abosrb, Block, and Cover ("ABC") is a common principle used in sound masking an environment. For example, in order to achieve speech privacy, sound masking is used in conjunction with sound proofing the walls and the ceiling (*absorb*), blocking the sounds from spreading by using partitions (*block*) and finally putting in sound masking speakers in proper locations (*cover*).

Pink noise (equal energy per octave) is the most common masking noise used by majority of sound masking solutions. Another alternative is white noise (equal energy per hertz) that may be used in place of pink noise. The noise that is to be used as the masking signal should also be random in nature. This feature helps in thwarting a profiling attack where an attacker can choose a small sample of the noise and try to reduce it from the overall audio.

Our aim while designing YELP is to fulfill previously mentioned design principles namely: effectiveness, design independence and zero-effort. In addition, we have to consider the cost of defense mechanism as ZEBRA scheme is geared towards authentication on multiple terminals in medical facilities and installation of additional hardware on each terminal would increase the cost linearly. Keeping in mind above factors, we decided to go with sound masking as our choice for YELP.

## 3.2 Design and Implementation

Our YELP design based on sound masking involves producing appropriate masking sound at each authentication terminal that is able to hide keystroke emanations generated during victim's interactions. They also serve to obfuscate the auditory channel of audio-based attacker that result in cloaking of any keystroke sounds reaching the attacker with the help of masking sound. Masking sound can be generated using speakers installed at every authentication terminal that uses ZEBRA scheme. Most computer terminals already come equipped with speakers and in case of terminals devoid of inbuilt speaker, portable speakers can be used that are conveniently cheap. We also include speakers that could be wall mounted, thereby giving a more dispersed coverage for the masking sound. These wall mounted speakers should be installed in a way that maximizes the effect of masking sound and does not have any blind spots near authentication terminals where the masking sound is less audible than any other authentication terminal.

## 3.3 Design Principles and Rationale

**Choice of Masking Sounds**: The choice of proper masking sound is important for an effective defense against *audio-only opportunistic keyboard-only* adversary. A masking sound should be able to make sure that the attacker is unable to make out the keystrokes being typed by the victim and the masking sound should not be disruptive to the natural state of the surrounding environment. Commonly used masking sounds include pink noise, white noise, music, commonly occurring background sounds like raindrops, ocean waves, wind blowing, coffee shop chatter, etc. Some of these noises are also used as relaxation aids and reduce distraction from unwanted sounds.

We tested white noise, music and naturally occurring sounds as possible candidates for masking sound based on precedent of their usage. White noise was generated using the noise generator AM 1200 [5], music was produced by playing "top 40 hits this week" and naturally occurring sounds were chosen from Noisli [37]. White noise has been used in commercial sound masking applications for suppressing unwanted sounds in offices, airplanes cockpits, military buildings etc. Our choice of music was inspired by coffee shops and cafes that play popular music on loudspeakers to provide a relaxing atmosphere as well as tone down surrounding conversation. Commonly occurring background sounds, in particular coffee shop chatter and white noise from [37], were used for similar purpose as previously described.

We conducted experiments in ZEBRA setup with masking sound generated at both attacker and victim's terminal. This setup seems most suitable as it hides the victim's interaction with the terminal as well as make it difficult for the attacker to eavesdrop on victim's keystrokes. In case of musical sounds, we used a centrally located loudspeaker equidistant from both the attacker and the victim. We measured the sound pressure level of masking sound at victim's and attacker's terminal setting it to be no more than 65db. From all the masking sounds described earlier, white noise generated at both ends and music from an equidistant loudspeaker performed the best. Combinations of coffee shop chatter along with white noise and other sounds such as ocean waves, wind, train track noises when played along with white noise failed to mask the keystrokes. Thus we proceeded with white noise generated at both the ends and music playing from an equidistant loudspeaker as our choices for defense mechanism design. A high level design of our defense setup is depicted in Figure 2.

# 4 REIMPLEMENTING ZEBRA

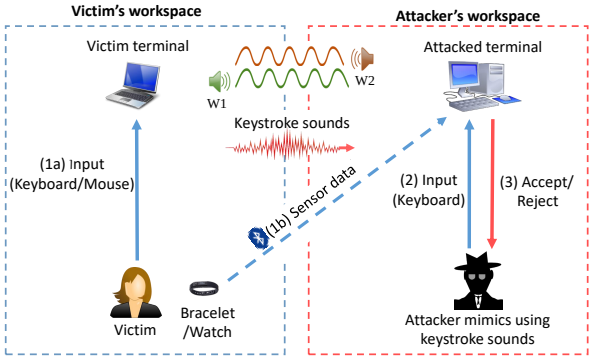
In this section, we report on our reimplementation of the ZEBRA system as documented by Mare et al. [33] and Huhta et al. [25], which is essential to evaluating our YELP defense system.

## 4.1 Preliminaries and Design

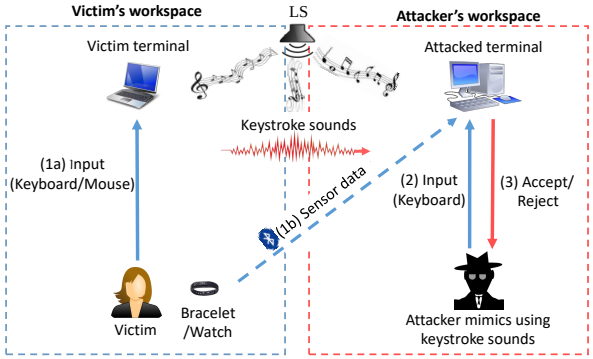
**Hardware:** In our implementation, we considered a standard PC as a terminal and LG G Watch R smartwatch as a bracelet. LG watch is a widely available smartwatch with 1.2 GHz CPU and 512MB RAM that comes with accelerometer and gyroscope. It has sampling rate of 200Hz similar to the smartwatch used in [25].

**Software:** Our implementation of ZEBRA consists of two applications: *Android Wear Application* that runs on the LG watch, and *Java application* that runs on the terminal. Android Wear application captures motion measurements of the user's wrist while terminal application captures the actual keyboard/mouse interaction that it observes at its terminal. Android Wear application and Java application communicate through Bluetooth to synchronize their clocks and to send motion measurements from the watch to the terminal. Rest of the components of ZEBRA, specifically *Interaction Extractor, Segmenter, Feature Extractor, Interaction Classifier, and Authent*icator that comprises "ZEBRA Engine", are implemented in MATLAB with the functionalities as described in Section 2.1.

**Feature Set and Classifier** We used same set of 12 features extracted from each of the accelerometer and gyroscope signals of




(a) Defense setup with white noise. 'W1' and 'W2' indicate the speakers at victim's and attacker's space, respectively that generate white-noise.




(b) Defense setup with music. 'LS' represent an equidistant loudspeaker that plays the music.

**Figure 2: Defense Design Model**

segmented blocks as considered in [33] and [25]. They are *mean, median, variance, standard deviation, median absolute deviation (MAD), inter-quartile range (IQR), power, energy, peak-to-peak amplitude, auto-correlation, kurtosis, and skew*. Similar to the implementation in [25], we used Random Forest classifier [10]. It consists of 100 weak-learners and each of the learners considers *sqrt(n)* features, where 'n' is total number of features, 24 in our case. Moreover, the classes were weighted to account for any imbalances in the training dataset. We used the exact parameters as provided in [25]. A full list of parameters are shown in Appendix Table 2.

## 4.2 Performance Evaluation

**Benign Setting:** We followed the same approach as in [25, 33] to evaluate the usability of the system i.e., we computed the false negative rate (FNR) as the fraction of interaction windows from a user that the *Authenticator* marked as from "different user". Similarly, we used leave-one-user-out cross validation approach over the 39 samples collected from 13 user sessions, each session consisting of three different scenarios, as described in Section 5. We trained the classifier using 36 samples of bracelet data from all the other 12 sessions. We then tested the classifier using 3 samples from the current session. Thus, we built 12 different classifiers, and reported the aggregate classification results of 39 samples.

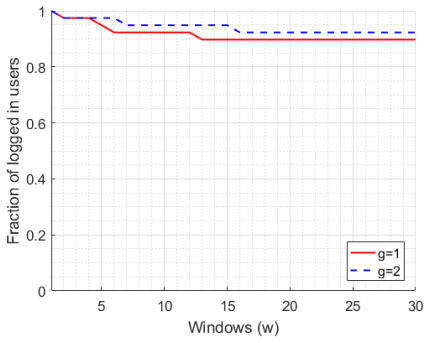



**Figure 3: Fraction of users remaining logged in to ZEBRA after '*n*' authentication windows (with $w = 20$, $m = 60\%$)**

**Table 1: Confusion matrix for 39 legitimate user samples.**

| | | Predicted | | |
|---|---|---|---|---|
| | | **Typing** | **Scrolling** | **MKKM** |
| **Actual** | **Typing** | 8139 | 72 | 331 |
| | **Scrolling** | 100 | 1068 | 5 |
| | **MKKM** | 483 | 39 | 5157 |

FNRs of our implementation of ZEBRA system (shown in Appendix Figure 9) are in the range between 0-10% which is inline with [33] (0-18% ). FNRs are less than 6% for window sizes above 15. We also computed the estimated time (in terms of number of authentication windows) for which a legitimate user remained logged in. Similar to [25, 33], we fixed w = 20 and m = 60% for estimating this time. Figure 3 shows fraction of users still logged in after given *'n'* authentication windows for a grace period (g) of 1 and 2. With a strict grace period of ($g$) = 1, 90% of the users were recognized as a correct user through out the session while with lenient grace period of $g$ = 2, this fraction slightly increases to nearly 92%. These fractions seems in line with the results reported in [33].

Table 1 shows the classification performance of *Interaction Classier* combining all 39 (13 x 3) classifications in the form of confusion matrix. This confusion matrix shows that our classifier is good at correctly inferring the events similar to the implementation of [25]. For an instance, we obtain a precision of 93.33% (8139/8722) and a recall of 95.28%(8139/8542) for recognizing the typing events. These values are in line with those reported in [25].

**Security against Innocent Adversaries:** Innocent adversaries are the general users who inadvertently start using a terminal where another user has already logged in. Similar to [25], we evaluate the security of our implementation of ZEBRA against such innocent adversaries by computing True Negative Rate (TNR) for *mismatching* sequences where actual interaction sequence of one sample is compared to predicted interaction sequence of a different sample. While mismatching the sequences, traces were synchronized by aligning the starting point of the sequences being compared. TNR for such mismatching sequences is the fraction of windows that are correctly classified as a *wrong* user.

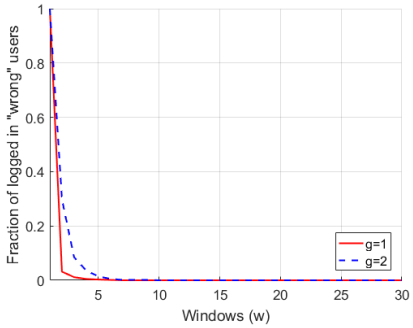



**Figure 4: Results for simulated *innocent* adversaries against ZEBRA. Fraction of "wrong" users remaining logged in after '*n*' authentication windows for different grace period (g).**

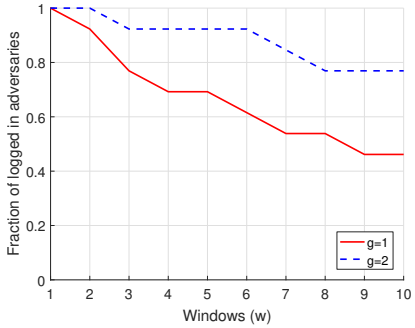



**Figure 5: Results for *audio-only opportunistic keyboard-only* attackers against ZEBRA (without YELP). Fraction of attackers remaining logged in after '*n*' authentication windows for different grace periods (g).**

For the threshold of 60-70% for TNR (Appendix Figure 10), most (more than 85%) of the authentication windows are correctly classified as non-matching windows for the window size above 20. Figure 4 shows the fraction of innocent adversaries remaining logged in while interacting with the terminal for a given number of authenticating windows when using $w = 20$ and $m = 60\%$. It can be seen that all the wrong users were kicked out within 5 authentication windows which indicates that our system is robust against such innocent adversaries.

# 5 YELP EXPERIMENT AND EVALUATION

## 5.1 Data Collection

In our study, we recruited 13 users by word of mouth. Participants were mostly students with age ranging from 18 to 35. 9 of the participants were males and 4 were females. All the participants were required to play the role of users (victim) while two of the trained researchers played the role of expert attackers following the methodologies used in [25]. Participants were told that the purpose of the experiment was to study the effect of ambient sound on user-behavior towards the terminal. We purposely did not tell the participants about the actual motive of the experiment before the experiment because it might have impacted the user-behavior. They were also told that either *white-noise* or *music* will be played during the experiment. Before starting the experiment, they were asked about their general demographic information. During the course of experiment, the participants performed three 10-minute tasks of filling a web form similar to [25, 33] in three different settings: (a) in devoid of YELP (ZEBRA only), (b) in presence of YELP with *music*, and (c) in presence of YELP with *white noise*. Experiments were conducted in a lab setting (quiet environment) to make the scenario advantageous to attacker. Our study and the data collection followed the IRB procedures at our institutions.

From each 10 minute task, two sets of user data were collected. First set of user data consisted of accelerometer and gyroscope readings from the bracelet worn by the user and the second set contained the actual sequence of user-terminal interaction extracted by Interaction Extractor on the terminal. An attacker assigned to a user performed *audio-only opportunistic keyboard-only* attack in each of the three settings mentioned above. In each of the settings, attacker and victim were positioned approximately at a distance of 2m, both facing away from each other so that attacker could not see the victim but could hear victim interacting with the terminal. The 13 user sessions resulted in a total of 39 samples, each sample consisting of three different measurements: the motion sensor data from bracelet of the user, the actual interaction sequence of the user, and the interaction sequence of the attacker. All the measurements within a sample were synchronized.

## 5.2 Results

*5.2.1 Keyboard-only Opportunistic Attack against ZEBRA (without YELP).* To validate *audio-only opportunistic keyboard-only* attack against ZEBRA, we used only the samples collected in the settings without implementation of YELP. For each of the user, we built a separate classification model by employing leave one user out approach. In particular, we used 36 samples from 12 user sessions to build a classifier for a given user's sample set. Finally, result was computed by aggregating the results from 13 classifiers for each of the traces. In the benign setting, 12 out of 13 users were recognized correctly as a legitimate user by ZEBRA and they were able to remain logged for entire duration of experiment at both $g = 1$ and $g = 2$.

Similar to [25], we computed the average False Positive Rate (FPR) for different thresholds($m$) between 50% and 70%, and for different window sizes($w$) from 5-30 (presented in Appendix Figure 11). The FPR represents the fraction of attackers' authentication windows that were incorrectly marked as from the original user. With a lenient threshold of 50%, the FPR values range from 75% to 98% while with a strict threshold of 70%, the FPR values range from 55% to 70%. General perception is that the high FPR will indicate the high probability of an attacker remaining logged in to the system for longer period of time. However, as most of the activities of victim and attacker constitute typing (as can be seen from confusion matrix in Table 1), keyboard interaction from an attacker may match accidentally with motion sensor from a victim. Therefore, in our case, FPR is not an effective measure to visualize the impact of masking sound on the attacker's performance. Instead, a measure of how quickly ZEBRA recognize the attacker successfully would show the actual impact of masking sound on the attackers' performance. So, in the rest of this section we use the measure of how long attacker can remain logged in to the system

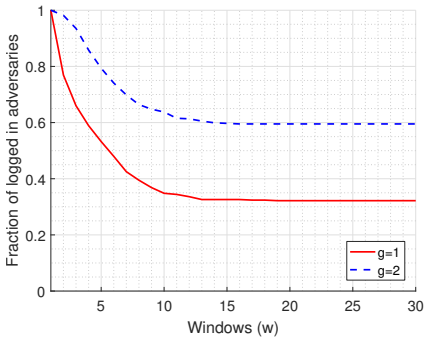



**Figure 6: Results for simulated *keyboard-only random* attackers against ZEBRA. Fraction of attackers remaining logged in after '*n*' authentication windows for different grace periods (g).**

to evaluate the effectiveness of our defense YELP using $m = 60\%$ and $w = 20$.

We plot the fraction of logged-in adversaries as a function of number of authentication windows by setting $m = 60\%$ and $w = 20$ to see how long an opportunistic attacker can remain logged in to the system. This represents the period of time in terms of authentication windows during which the attackers remain logged-in to the system. Figure 5 depicts this fraction for $g = 1, 2$. Note that an opportunistic attacker against ZEBRA creates very less number of interactions, and hence less number of windows, as he is selectively mimicking only the subset of victim keyboard activities. In our experiment, we get the number of interactions equivalent to 30 windows from normal users while with opportunistic attackers we get the number of interactions equivalent to only 10 windows on average. So, for the benign case, we present result for 30 authentication windows while for attack case we present result for 10 authentication windows.

The high FPR of approximately 90% (for $w = 20$ and $m = 60\%$) results in more than 45% of the attackers remaining logged in at $g = 1$ and more than 75% of attackers remaining logged in for $g = 2$ for the whole duration of the experiment. With the same parameter settings, [25] reported that approximately 15% of the opportunistic attackers remained logged in for whole 10 minutes duration at $g = 1$. The higher fraction of successful attackers in our case as compared to that in [25] may be because of the higher FPRs of our implementation of ZEBRA. More than 75% attackers remain logged in successfully up to third authenticating windows at $g = 1$ and more than 90% at $g = 2$. Nearly 70% of the attackers remained logged in until fifth authenticating windows at $g = 1$. This result shows that ZEBRA system is vulnerable highly against *audio-only opportunistic keyboard-only* attackers. Therefore, it highlights the need to either minimize, if not eliminate, the acoustic emanation through keyboard or use the masking sounds to hide the acoustic emanation to disable the attacker from hearing the keystroke sounds.

*5.2.2 Keyboard-only Random Attack.* In *keyboard-only random* attack, an attacker tries to access the victim's terminal using only keyboard. Unlike *audio-only opportunistic keyboard-only* attack where attacker tries to selectively mimic the victim keyboard activities based on audio cues from the victims' keystrokes, *keyboard-only random* attacker performs only keyboard activities in normal fashion without any audio or visual cues from victim. As we are interested in evaluating the effectiveness of masking sound against *audio-only opportunistic keyboard-only* attackers, we use *keyboard-only random* attack scenario as a baseline for our evaluation.

We simulate this scenario by *mix-matching* the samples, where interaction sequence of one sample is applied against motion sensor readings from a different sample. While mix-matching the samples, they were synchronized by aligning the starting point of interaction sequence of a sample being used with starting point of interaction sequence of a sample whose motions sensor readings are being used. All the mouse related events (i.e., scrolling and MKKM) were also removed from the interaction sequence as we are interested in keyboard-only random attack. For each mix-matched sample, we used the classification model (built earlier in *audio-only opportunistic keyboard-only* attack analysis) corresponding to the user whose motion sensor data is being used.

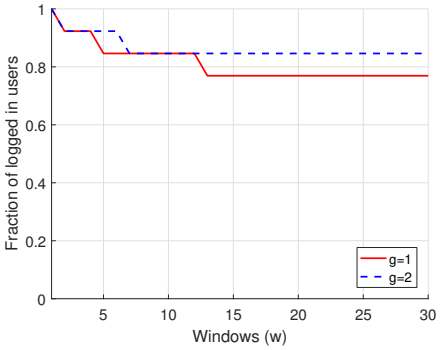
Figure 6 shows the performance of ZEBRA against *keyboard-only random* attackers. It shows the fraction of random attackers remaining logged in successfully for given number of authentication windows for g = 1, 2. At $g = 1$, more than 50% of the attackers were kicked out after 5 authentication window and nearly 65% of the attackers were kicked out after 10 authentication window. When using $g = 2$, only 20% of the attackers were logged out after 5 authentication window while more than 35% of the attackers were logged out after 10 authentication window. At $g = 1$, more than 30% (and 60% at $g = 2$) of the random attackers were able to withstand the ZEBRA system for the whole 10 minute session. The success of random attackers to remain logged in during the entire session may be because most of the interactions constitute keyboard interactions and interaction from one sample may have match accidentally with the motion sensor readings from a different sample while mix-matching.

*5.2.3 Keyboard-only Opportunistic Attack against YELP.* We now present the evaluation of the effectiveness of YELP, in particular the effect of *white-noise* and *music* of YELP as a masking sound, against *audio-only opportunistic keyboard-only* attack while considering the attacks without YELP scenario as a base scenario.

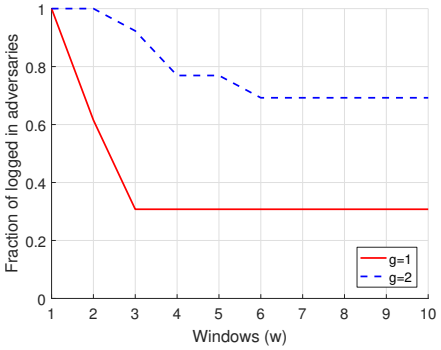
**In presence of White-Noise:** To see the effectiveness of YELP with *white-noise* as a defense measure against *audio-only opportunistic keyboard-only* attackers, we use all the user samples collected in presence of white-noise. The training and classification results were computed in a similar manner as in Section 5.2.1.

Figure 7 shows the performance of the users with ZEBRA (Figure 7a) and the performance of *audio-only opportunistic keyboard-only* attackers against ZEBRA in presence of YELP with *white-noise* (Figure 7b). Figure 7a shows the fraction of normal users remaining logged in for given number of authentication windows. At $g = 1$, more than 75% of the users were correctly recognized as a correct users through out the duration of the experiment. When using $g = 2$, nearly 85% of the users were correctly identified as a legitimate user during the whole 10 minute session.

Figure 7b shows the fraction of *audio-only opportunistic keyboard-only* attackers remaining logged in successfully for given number of authentication windows in presence of YELP with *white-noise*. The fraction of attackers kicked out in presence of YELP with *white-noise* (Figure 7b) is statistically significant (Wilcoxon signed-rank test,

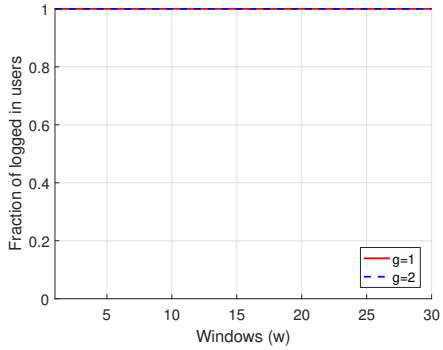



**(a) Fraction of users remaining logged in after *'n'* authentication windows for different grace period (g).**

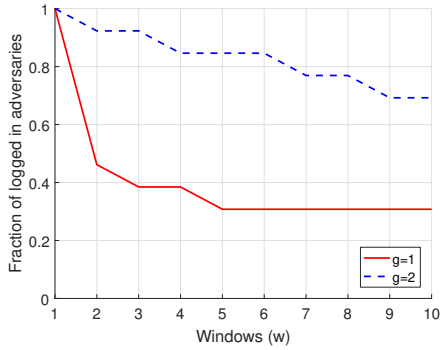



**(b) Fraction of attackers remaining logged in after *'n'* authentication windows for different grace period (g).**

**Figure 7: Results for the users and *audio-only opportunistic keyboard-only* attackers in presence YELP with *white-noise* as a masking sound.**




**(a) Fraction of users remaining logged in after *'n'* authentication windows for different grace period (g).**




**(b) Fraction of attackers remaining logged in after *'n'* authentication windows for different grace period (g).**

**Figure 8: Results for the users and *audio-only opportunistic keyboard-only* attackers in presence of YELP with *music* as a masking sound.**

$z = -2.803$ and p = 0.003 < 0.05) compared to the setting without YELP (Figure 5) across all the windows. For example, when using $g = 1$, almost 70% of the attackers were kicked out in presence of YELP with *white-noise* while in the settings without YELP, less than 25% of attackers were kicked out at third authentication window. In case of *keyboard-only random* attack, 40% of the attackers were kicked out at the same authentication window.

For YELP with *white-noise*, only 30% of the attackers were able to remain logged-in during the entire experiment which is nearly the same fraction as in *keyboard-only random* attack (30%) while more than 45% of *audio-only opportunistic keyboard-only* attackers were able to remain logged-in for entire duration of experiment in the setting without YELP. This shows that the *white-noise* can potentially mask the keystrokes sounds, thereby making it difficult for the *audio-only opportunistic keyboard-only* attacker to hear any keystroke sounds. Therefore, the system with $g = 1$, can quickly and effectively detect larger fraction of attackers successfully in presence of *white noise*. At $g = 2$, nearly 70% of the attackers were able to remain logged in during the entire experiment in presence of *white-noise*. This fraction lies between the fraction (nearly 75%) of *audio-only opportunistic keyboard-only* attackers in the setting without the implementation of YELP and the fraction (60%) of *keyboard-only random* attackers who remain logged in through out the session. This shows that *white-noise* of YELP works fairly as a defense measure against the opportunistic attackers at $g = 2$.

**In presence of Music:** We use the similar approach as in YELP with *white-noise* scenario to see the effectiveness of the use of YELP with *music* as a masking sound against *audio-only opportunistic keyboard-only* attack. We use all the 13 user sample sets collected in presence of *music* as described above.

The performance of the users and the opportunistic attackers against ZEBRA in presence of YELP with *music* as a masking sound is presented in Figure 8. Figure 8a shows that all the users were recognized correctly as a legitimate user during the entire 10 minute session for both $g = 1$, and 2. Figure 8b shows the fraction of attackers remaining logged-in as a function of number of authentication windows for $g = 1, 2$. The fraction of attackers kicked out in presence of YELP with *music* (Figure 8b) is statistically significant (with $z = -2.934$ and p = 0.002 < 0.05) compared to the setting without YELP (Figure 5). For an instance, using $g = 1$ almost 60% of the attackers were logged out after third authentication window in presence of YELP with music while in the setting without YELP less than 25% of the attackers were kicked out at the same authentication window.

As compared to the setting without YELP, a smaller fraction (30% over 45%) of *audio-only opportunistic keyboard-only* attackers were able to remain logged in during the entire session in presence of

*music* as a masking sound. When comparing with *keyboard-only random* attack, nearly same fraction of attackers were able to logged in during the entire session. When $g$ = 2, though there seems to be less impact on the fraction of attackers remaining logged in through out the entire experiment duration, YELP with *music* works fairly well similar to *white-noise*. It is indicated by the fraction (more than 85%) of attackers being logged out (at fifth authentication window) in presence of YELP with *music* that lies in between the fraction (more than 90%) of *audio-only opportunistic keyboard-only* attackers in the setting without YELP and the fraction (80%) of *keyboard-only random* attackers. This shows that YELP with *music* can reduce the performance of *audio-only opportunistic keyboard-only* attackers at the level of the performance of the random attacker at $g$ = 1 while when $g$ = 2, *music* tends to reduce the performance of *audio-only opportunistic keyboard-only* attackers towards the performance of *keyboard-only random* attackers. Therefore, YELP with *music* as a masking sound works almost at the same level as the *white-noise* for $g$ =1, and a bit less for $g$ = 2.

## 6 RELATED WORK

The most common form of user authentication on computing devices has been passwords. As time has progressed, passwords have become longer and more complex requiring the users to extra effort to remember or securely secure their only means of authentication. However, humans always have a tendency to choose a simple, easy to recall password [9, 11, 17, 19, 27, 36, 38, 39]. This fact leads to an inherent vulnerability due to human factor in user authentication.

In order to improve upon user authentication mechanism, several other schemes have been proposed that do not involve passwords. Secure physical tokens [41] have been proposed in conjunction with passwords to bolster user authentication. Biometric authentication takes it one step further by utilizing various metrics like fingerprint pattern [14], hand veins[29], iris pattern [13], facial structure [8] or blood vessel [43] information to establish the identity of a human user. They offer a better solution to user authentication than traditional authentication mechanism as it is easy to test their accuracy based on prediction of false positive and false negative rates for each type of metrics used [2, 16, 22, 26]. Behavioral biometrics like gait of a user [20], keystroke typing pattern [28, 34, 35], mouse dynamics [1] have all been proposed as authentication schemes that are less intrusive from a user's perspective. Continuous user authentication (e.g. [30]) elevates the security offered by previously mentioned authentication schemes by continuously determining the identity of the user as a trusted entity in the background.

Audio channels have previously been exploited for launching side channel attacks against keystrokes ([3, 7, 18, 23, 44, 45]), mechanical printers [6], RFIDs/wireless devices [24] and CPU emissions [21]. The attacks on keystroke emanations have been successful at extracting individual keystrokes with a reasonable degree of accuracy. Random passwords as well as typed text have been decoded using the techniques described in such attacks. Halevi et al.[24] were able to extend acoustic eavesdropping from keystrokes to vibrations and Genkin et al. [21] showed that it is feasible to extract the full RSA key from CPU acoustic emanations.

Sound masking has traditionally been used as a commercial solution ([4, 12, 31, 32, 42]) for providing speech privacy in call centers, offices, libraries, medical facilities, law and government facilities etc. It has been proposed as a way to reduce distraction, improve focus and protect sensitive conversations. It is touted as a low cost measure to achieve privacy as compared to architectural improvements in the environment. Zhuang et al.[45] also proposed addition of masking sound while typing as a way to reduce the quality of the acoustic emanations that may impede keystroke classification.

## 7 CONCLUSION AND FUTURE WORK

ZEBRA is a representative zero-effort deauthentication system. However, it has been shown susceptible to a practical audio-visual based vulnerability where a clever opportunistic human attacker, by observing/listening victim's activities with the authentication terminal, can compromise the security of ZEBRA. Since the visual observation attacks can be relatively easily addressed by the use of visual barriers around the login terminals, our work focused on defending against audio-based observation attacks. Given the severity of this threat against ZEBRA system, we designed an effective defensive approach – YELP– that works transparently with ZEBRA. In order to cloak the typing sounds, YELP utilizes the notion of sound masking that has already been in use as a commercial solution for providing speech privacy, and also as a way to reduce distraction, improve focus and productivity. In particular, YELP uses two types of masking sounds, white noise produced by the computer terminals themselves and musical sounds produced by a centrally located loudspeaker system present in the environment.

To evaluate the effectiveness of YELP, we recreated the audio-based opportunistic attacks on ZEBRA as proposed in [25] and tested it against ZEBRA and YELP. Our results showed that YELP can improve the security of ZEBRA without significantly impacting its performance in the benign setting. We demonstrated that YELP with both *white-noise* and *music* as a masking sound can effectively hide the acoustic leakage from ZEBRA system, and can reduce the attack success rate of an audio-based opportunistic attacker.

In the future work, YELP may need to be evaluated against potentially more sophisticated attacks, such as those employing noise filtering techniques, for example, using noise cancellation headphones. Noise cancellation occurs by measuring ambient noise and producing an *anti-noise* signal to cancel the ambient noise. Since ambient noise is generally considered to be any sound except speech, it remains to be seen if noise cancellation can be geared towards filtering masking sound from keystroke sounds without affecting the quality of the keystroke sounds themselves.

On a broader note, we believe that the sound masking based defensive approach like YELP can be broadly applied against other audio-based side channel attacks (e.g., those involving information leakage through CPU and printer sounds). This approach would constitute an independent as well as low cost system when compared to architectural improvement techniques that are usually deployed against audio-based attacks and may be expensive. While designing defensive tool based on sound masking, proper selection of masking sounds entails choosing sounds that are less distracting and at the same time effective at masking sensitive audio leakage of the vulnerable system. Further study is required to create a refined design methodology for practical and secure sound masking based defensive tools against audio-based attacks.

# REFERENCES


[1] Ahmed Awad E Ahmed and Issa Traore. 2007. A new biometric technology based on mouse dynamics. *IEEE Transactions on Dependable and Secure Computing* 4, 3 (2007), 165–179.

[2] Petar S. Aleksic and Aggelos K. Katsaggelos. 2006. Audio-Visual Biometrics. *Proc. IEEE* 94, 11 (2006), 2025–2044.

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

[4] Atlas IED. 2017. Speech Privacy Solutions for Commercial Environments. https://www.atlasied.com/speech-privacy-solution. (3 2017).

[5] AtlasIED. 2017. Self Contained Sound Masking System UL2043 with built in Loudspeakers. https://www.atlasied.com/low-profile-sound-masking-system-ul2043. (3 2017).

[6] Michael Backes, Markus Durmuth, 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] Charles Beumier and Marc Acheroy. 2000. Automatic 3D face authentication. *Image and Vision Computing* 18, 4 (2000), 315–321.

[9] Matt Bishop and Daniel V. Klein. 1995. Improving system security via proactive password checking. *Computers and Security* 14, 3 (1995), 233–249.

[10] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[11] Julie Bunnell, John Podd, Ron Henderson, Renee Napier, and James Kennerdy-Moffat. 1997. Cognitive, associative and conventional passwords: Recall and guessing rates. *Computers and Security* 16, 7 (1997), 645–657.

[12] Cambridge Sound Management, Inc. 2017. Cambridge Sound Management: The Leader in Sound Masking. http://cambridgesound.com/. (3 2017).

[13] Siew Chin Chong, Andrew Beng Jin Teoh, and David Chek Ling Ngo. 2005. Iris authentication using privatized advanced correlation filter. In *Advances in Biometrics*. Springer, 382–388.

[14] T Charles Clancy, Negar Kiyavash, and Dennis J Lin. 2003. Secure smartcard-based fingerprint authentication. In *Proceedings of the 2003 ACM SIGMM workshop on Biometrics methods and applications*. ACM, 45–52.

[15] Mark D Corner and Brian D Noble. 2002. Zero-interaction authentication. In *Proceedings of the 8th annual international conference on Mobile computing and networking*. ACM, 1–11.

[16] John Daugman. 2004. How Iris Recognition Works. *IEEE Transactions on Circuits and Systems for Video Technology* 14, 1 (2004), 21–30.

[17] David C. Feldmeier and Philip R. Karn. 1990. UNIX password security-ten years later. In *Advances in Cryptology (CRYPTO'89)*. 44–63.

[18] 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.

[19] F.M. Furnell, P.S. Dowland, H.M. Illingworth, and P.L. Reynolds. 2000. Authentication and Supervision: A Survey of User Attitudes. *Computers and Security* 19, 6 (2000), 529–539.

[20] Davrondzhon Gafurov, Kirsi Helkala, and Torkjel Søndrol. 2006. Biometric gait authentication using accelerometer sensor. *Journal of computers* 1, 7 (2006), 51–59.

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

[22] Lawrence O' Gorman. 2003. Comparing Passwords, Tokens, and Biometrics for User Authentication. *Proceedings of the IEEE,* 91, 12 (2003), 2021–2040.

[23] 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*.

[24] Tzipora Halevi and Nitesh Saxena. 2013. Acoustic Eavesdropping Attacks on Constrained Wireless Device Pairing. In *IEEE Transactions on Information Forensics and Security (TIFS)*.

[25] O Huhta, P Shrestha, S Udar, M Juuti, N Saxena, and N Asokan. 2016. Pitfalls in Designing Zero-Effort Deauthentication: Opportunistic Human Observation Attacks. In *Network and Distributed System Security Symposium (NDSS)*.

[26] Anil K. Jain, Ross Arun, and Sharath Pankanti. 2006. Biometrics: A tool for information security. *IEEE Transactions on Information Forensics and Security* 1, 2 (2006), 125–143.

[27] David L. Jobusch and Arthur E. Oldehoeft. 1989. A survey of password mechanisms: Weaknesses and potential improvements. *Computers and Security* 8, 8 (1989), 675–689.

[28] Rick Joyce and Gopal Gupta. 1990. Identity authentication based on keystroke latencies. *Commun. ACM* 33, 2 (1990), 168–176.

[29] Ajay Kumar and K Venkata Prathyusha. 2009. Personal authentication using hand vein triangulation and knuckle shape. *Image Processing, IEEE Transactions on* 18, 9 (2009), 2127–2136.

[30] John Leggett, Glen Williams, Usnick Mark, and Mike Longnecker. 2007. Dynamic identity verification via keystroke characteristics. *International Journal of Man-Machine Studies* 35, 6 (2007), 859–870.

[31] Lencore. 2017. Lencore: Comfort. Privacy. Intelligibility. http://www.lencore.com/. (3 2017).

[32] Logison Acoustic Network. 2017. Sound Masking. https://www.logison.com/technology/sound-masking. (3 2017).

[33] Shrirang Mare, Andrés Molina Markham, Cory Cornelius, Ronald Peterson, and David Kotz. 2014. ZEBRA: zero-effort bilateral recurring authentication. In *Security and Privacy (SP), 2014 IEEE Symposium on*. IEEE, 705–720.

[34] Fabian Monrose, Michael K Reiter, and Susanne Wetzel. 2002. Password hardening based on keystroke dynamics. *International Journal of Information Security* 1, 2 (2002), 69–83.

[35] Fabian Monrose and Aviel D Rubin. 2000. Keystroke dynamics as a biometric for authentication. *Future Generation computer systems* 16, 4 (2000), 351–359.

[36] Robert Morris and Ken Thompson. 1979. Password security: a case history. *Commun. ACM* 22, 11 (1979), 594–597.

[37] Noisli. 2017. Noisli: Improve focus and boost your productivity. https://www.noisli.com/. (3 2017).

[38] Rachel Pond, John Podd, Julie Bunnell, and Ron Henderson. 2000. Word Association Computer Passwords: The Effect of Formulation Techniques on Recall and Guessing Rates. *Computers and Security* 19, 7 (2000), 645–656.

[39] Bruce L. Riddle, Murray S. Miron, and Judith A. Semo. 1989. Passwords in Use in a University Timesharing Environment. *Computers and Security* 8, 7 (1989), 569–579.

[40] N Roy, M Gowda, and R. R D Choudhury. 2015. Ripple: Communicating through physical vibration. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI15)*.

[41] RSA. 2017. RSA SecurID Hardware Tokens. https://www.rsa.com/en-us/products/rsa-securid-suite/securid-hardware-tokens. (3 2017).

[42] Speech Privacy Systems. 2017. Speech Privacy Systems™. https://www.speechprivacysystems.com/. (3 2017).

[43] Masaki Watanabe, Toshio Endoh, Morito Shiohara, and Shigeru Sasaki. 2005. Palm vein authentication technology and its applications. In *Proceedings of the biometric consortium conference*. 19–21.

[44] 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*.

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

## APPENDIX: ADDITIONAL TABLES AND FIGURES

**Table 2: Parameters and their values used in our implementation of ZEBRA (same as in [25]). For MKKM, idle threshold and maximum duration is 5s.**

| Parameter | Value |
|---|---|
| Mininum duration | 25 ms |
| Maximum duration | 1 s |
| Idle threshold | 1 s |
| Window size ($w$) | 5-30 |
| Match threshold ($m$) | 50-70% |
| Overlap fraction ($f$) | 0 |
| Grace period ($g$) | 1, 2 |

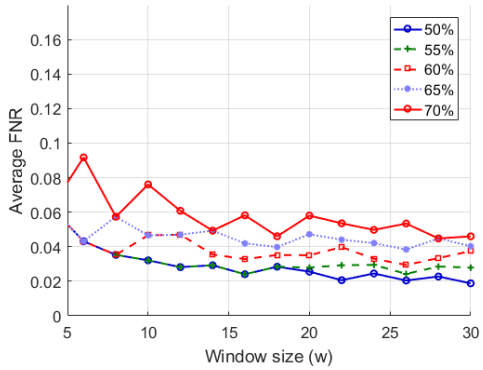



**Figure 9: Performance of legitimate users. Average FNR vs. window size (w) for different threshold (m) values. Fraction of windows that are incorrectly classified as mismatching.**

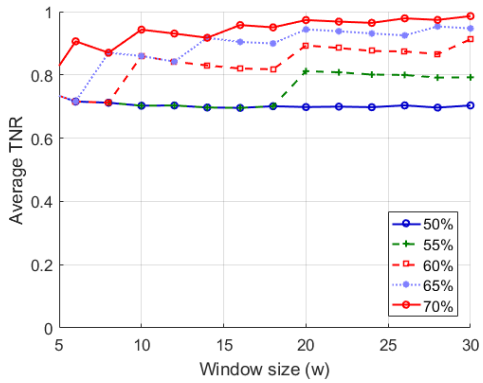



**Figure 10: Average TNR for different threshold ($m$) and different window size ($w$) values. Fractions of windows that correctly identify a wrong user in simulated accidental usage of the terminal.**

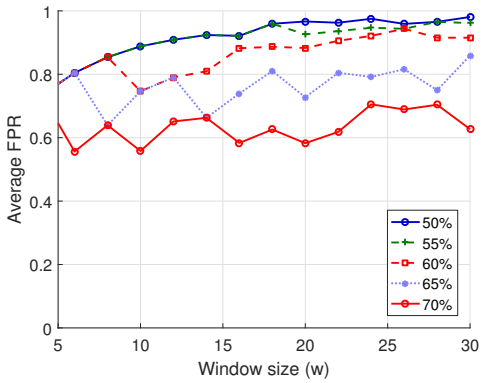



**Figure 11: Results for *audio-only opportunistic keyboard-only* attackers without YELP. Average FPR for different threshold ($m$) values and for different window sizes ($m$). Fraction of attacker windows that are incorrectly classified as from legitimate users.**