The Sounds of the Phones: Dangers of Zero-Effort Second Factor Login based on Ambient Audio

Babins Shrestha
University of Alabama at Birmingham
babins@uab.edu

Maliheh Shirvanian
University of Alabama at Birmingham
maliheh@uab.edu

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

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

Abstract

Reducing user burden underlying traditional two-factor authentication constitutes an important research effort. An interesting representative approach, *Sound-Proof*, leverages *ambient sounds* to detect the proximity between the second factor device (phone) and the login terminal (browser). Sound-Proof was shown to be secure against remote attackers and highly usable, and is now under early deployment phases.

In this paper, we identify a weakness of the Sound-Proof system, *namely*, the remote attacker *does not have to predict the ambient sounds* near the phone as assumed in the Sound-Proof paper, but rather can *deliberately make—or wait for—the phone to produce predictable or previously known sounds* (e.g., ringer, notification or alarm sounds). Exploiting this weakness, we build *Sound-Danger*, a full attack system that can successfully compromise the security of Sound-Proof. The attack involves buzzing the victim user's phone, or waiting for the phone to buzz, and feeding the corresponding sounds at the browser to login on behalf of the user. The attack works precisely under Sound-Proof's threat model.

Our contributions are three-fold. *First*, we design and develop the *Sound-Danger* attack system that exploits a wide range of a smartphone's functionality to break Sound-Proof, such as by actively making a phone or VoIP call, sending an SMS and creating an app-based notification, or by passively waiting for the phone to trigger an alarm. *Second*, we re-implement Sound-Proof's audio correlation algorithm and evaluate it against *Sound-Danger* under a large variety of attack settings. Our results show that many of our attacks succeed with a 100% chance such that the Sound-Proof correlation algorithm will accept the attacked audio samples as valid. *Third*, we collect general population statistics via an online survey to determine the phone usage habits relevant to our attacks. We then use these statistics to show how our different correlation-based attacks can be carefully executed to, for instance, compromise about 57% user accounts in just the first attempt and about 83% user accounts in less than a day. Finally, we provide some mitigation strategies and future directions that may help overcome some of our attacks and strengthen Sound-Proof.

CCS'16, October 24-28, 2016, Vienna, Austria

DOI: http://dx.doi.org/10.1145/2976749.2978328

1. INTRODUCTION

Two-factor authentication (2FA), combining the use of a password ("something you know") and a token ("something you have"), is gaining momentum for web authentication. A traditional 2FA scheme requires the user (Alice) to enter her password and copy a random one-time PIN (OTP) from the token over to the authentication terminal. This improves security because the attacker now needs to not only guess the user's password but also the current OTP value to hack into the user's account. The use of a general-purpose smartphone as a token [3, 12, 14], as opposed to a dedicated device [21, 27], helps improve usability and deployability of 2FA, and is currently a commonly used approach on the Internet.

However, the need to look-up and interact with the phone, and copy the OTP value during a 2FA authentication session lowers the system's usability, which may prevent users from adopting this approach for authentication [17]. In this light, researchers and practitioners have recognized the need for reducing, and ideally eliminating, the user burden underlying traditional 2FA, giving rise to an important research direction. The goal of such *zero-effort 2FA* schemes is to allow the user to login using the 2FA approach by ideally only typing in her password.

An interesting representative zero-effort 2FA approach, *Sound-Proof* [17], leverages *ambient sounds* to detect the proximity between the phone and the login terminal (browser). Specifically, during the login session, the browser and the phone each record a short audio clip, and the login is deemed successful only if the two recorded audio samples are highly correlated with each other (and the correct password is supplied). Except of entering the password, Sound-Proof does not require any user action (e.g., transferring PIN codes or even looking-up the phone) – mere proximity of the phone with the terminal is sufficient to login. It may also work even if the phone is inside a purse or pocket. Unlike other zero-effort 2FA approaches [11, 23], which rely upon proximity channels, such as Bluetooth or Wi-Fi, to automatically transfer the PIN codes, a compelling deployability feature of Sound-Proof is that it does not require browser plugins or any changes to the current browsers.

The main security goal of Sound-Proof is to defeat a *remote attacker*, who has learned the user's password (e.g., by hacking into a password database server of the web service in question), and is attempting to login to the user's account, and possibly multiple user accounts. As argued in [17], given the prominence of remote attacks on the web today, this is a very legitimate goal. In order to login to the user's account, the remote attacker against Sound-Proof would have to predict the ambient sounds in the environment of the phone and possibly be in a very similar environment as the user, which may be a difficult endeavor in practice, as shown in the

security analysis reported in [17]. In other words, if the attacker can not predict the user's environment and is in a different environment than the user, the audio samples at the browser's end and the phone's end would not correlate, thereby preventing the attacker from logging in [17]. Indeed, in the comprehensive security evaluation reported in [17], Sound-Proof was shown to be highly secure against such remote attackers. In addition, in the usability evaluation reported in [17], Sound-Proof was shown to be highly user-friendly, when contrasted with a traditional 2FA scheme involving OTPs [14]. Given these very promising security and usability properties, Sound-Proof is apparently now under early deployment phases in the form of a start-up (see: http://sound-proof.ch/).

In this paper, we set out to closely inspect the security of Sound-Proof, motivated by its very appealing usability and practicality features. Unfortunately, we identify a weakness of the Sound-Proof system. Namely, the remote attacker against Sound-Proof *does not have to predict the ambient sounds* near the phone, but rather can *make the phone create predictable or previously known sounds, or wait for the phone to produce such sounds* (e.g., ringer, notification or alarm sounds). Given the close physical proximity of the source of these sounds (the phone's speaker) and the receiver of these sounds (the phone's microphone), the phone's recordings would be dominated by these sounds rather than the ambient noises present in the environment.

Exploiting this weakness, we introduce and build *Sound-Danger*, a full attack system that can successfully compromise the security of Sound-Proof. The attack involves remotely buzzing the victim user's phone, or waiting for the phone to buzz on its own, and feeding the corresponding sounds at the browser to login on behalf of the user. The attack works precisely under the limits of Sound-Proof's threat model, only uses the information available in hacked password databases (e.g., passwords, phone numbers or other account information [1, 2, 4, 6–9]), is fully remote and can be launched against multiple user accounts. We note that phone numbers, in particular, are readily available in password databases as they are commonly used to facilitate account recovery in case of forgotten username/password and are essential for 2FA-supported web services which often need to send OTPs to users' phones via SMS (Sound-Proof also supports fallback to traditional 2FA [17]).

Our Contributions: We believe that our work makes the following scientific contributions:

1. *A Novel Attack against a Notable Zero-Effort 2FA Scheme*: We introduce, design and develop the *Sound-Danger* attack system that exploits a wide variety of a smartphone's functionality to break Sound-Proof, a prominent zero-effort 2FA scheme. Our remote attack involves either making the phone to generate known sounds, such as, by actively making a phone or VoIP call, sending an SMS and triggering an app-based notification, or by passively waiting for the phone to sound an alarm at a predictable moment. Our attack exploits the "sounds of the phones", which is fundamentally different from, and more devastating than, the attacks studied in [17] which exploit the "sounds of the ambience".

2. *Correlation Analysis of the Attack System*: We re-implement the Sound-Proof's audio correlation algorithm and evaluate it against *Sound-Danger* under a large variety of attack settings. Our results show that many of our attacks (e.g., WhatsApp[1] or Facebook calling, Viber[2] notifications and phone alarm) succeed with a 100% chance such that Sound-Proof' correlation detection engine accepts the attacked audio samples as valid, i.e., the attacker (browser) and the phone are deemed to be in proximity (even though they are remote).

3. *Real-World Attack Strategies based on Population Statistics*: As a representative example of how to deploy our attacks in practice, we collect general population statistics via an online survey to determine the phone usage habits and patterns relevant to our attacks. We then use these common statistics from our population sample to show how our different correlation-based attacks against Sound-Proof can be carefully executed to compromise, for example, about 57% user accounts in the first attempt and about 83% user accounts in less than a day. Our attack strategy is extensible to other population samples beyond the one we studied, and may actually be used to eventually compromise almost all user accounts.

Potential Defenses and Broader Implications: While our work mainly serves to raise an alarm against the pitfalls and challenges of designing zero-effort two-factor authentication mechanisms, we also provide some strategies for Sound-Proof designers that may help mitigate some of our attacks and could strengthen the security of Sound-Proof, hopefully without undermining its usability. Further work will be needed in this direction though. We believe that our work is timely, since the Sound-Proof system seems to be nearing deployment at this point. Addressing the vulnerability reported in our paper, prior to fully launching Sound-Proof in the wild, will help protect future Sound-Proof users.

Our attack is not just limited to the specific audio correlation algorithm implemented by Sound-Proof. Since we essentially create the very similar sounds at the attacker's (browser's) end as the victim's (phone's) end, it seems that any audio correlation engine may be defeated. Another algorithm for audio correlation was proposed in [16], which also seems vulnerable to our attacks. On the other hand, if the detection approach employs stricter parameters, such as very high correlation thresholds and narrower synchronization lags between audio samples, to lower the impact of the attack, it would considerably lower the usability of the system, since even in the benign settings, many matching samples will be rejected, preventing the legitimate user from logging in.

Also, at the conceptual level, our attack is not just limited to the domain of two-factor authentication. There exists other zero-interaction proximity detection and device pairing schemes based on ambient audio, which also seem vulnerable to our attack. For example, one security scheme is geared for preventing relay attacks in the context of "local" terminal authentication (e.g., mobile payments) [16]. And, another scheme is aimed for pairing of devices based on common ambient audio signals [22]. Here, if the attacker knows the victim's phone number or app account usernames, it could also succeed in defeating such schemes. However, in the threat model of such schemes, such personalized information about the victim's phone may not be available to the attacker, unlike Sound-Proof where this information is readily available from the same databases which leak the victim's passwords.

Overall, the main broader lesson learned from our study is that, while reducing the user effort and removing the user from the authentication loop is a compelling proposition, it requires utmost care in the design of such schemes.

2. BACKGROUND

Sound-Proof [17] is claimed to be a usable and deployable zero-effort 2FA mechanism, which does not require interaction between a user and the 2FA application on the device during the authentication process. In Sound-Proof, the second authentication factor is the proximity of the user's phone and the client terminal (browser),

[1]http://www.whatsapp.com/
[2]http://www.viber.com/

which is verified by the application on the phone by comparing the ambient noise recorded by the phone and the browser.

2.1 Threat Model

The primary goal of Sound-Proof is to defeat a remote attacker, who may be attempting to login into a victim user's account from a remote machine, which is in full control of the attacker. Sound-Proof's threat model assumes that this remote attacker has the knowledge of the victim user's username and password. This information can be learned, for example, via leaked password databases of the web service that may be using Sound-Proof or other web services for the purpose of authenticating its users. The attacker's goal is to authenticate to the web service on behalf of the user and possibly compromise multiple user accounts. Sound-Proof assumes that the attacker has not compromised the user's phone and/or the user's terminal. If the attacker gains control of one of the victim's devices, the security of any 2FA scheme reduces to the security of password-only authentication. Also, Sound-Proof does not consider targeted attacks such as those involving co-located malicious entities that are in close physical proximity of the victim.

This threat model may be weaker than that considered by traditional 2FA schemes involving OTPs. However, as argued in [17], given the prominence of remote attacks, this is still a very legitimate model. If more and more web services and users adopt Sound-Proof given its unique zero-effort feature, and remote attackers could still be thwarted, this will be a major improvement to the state of web authentication in practice.

As such, our proposed *Sound-Danger* system follows a threat model very similar to that of Sound-Proof. We consider that the attacker gets other user information from the leaked password database besides user credentials. That is, we assume that the password databases store phone numbers for password-only or 2FA implementations in order to send account recovery information or verification codes [1, 7], IP address information from which the users log in [1, 2, 9], or even users' physical address information [9]. The *Sound-Danger* attacker uses the phone numbers to perform active attacks while it utilizes IP addresses or physical address information to locate the users and their timezones. By identifying the timezone of the users, the attacker can estimate when a particular noise may occur at the users' side, such as morning alarms. Since many users often use the same usernames across multiple web applications, the attacker can utilize the username information to perform attacks based on notifications triggered by apps that use the same username. For example, if a user has the same username in the leaked database server and Skype, the attacker can send a notification (e.g., a friend's request) to user's Skype account. Like in Sound-Proof's threat model, the *Sound-Danger* system does not attempt targeted attacks. Rather, it assumes that the attacker can collect general population statistics through online user surveys in order to devise specific attack strategies against a population of users for compromising multiple user accounts.

2.2 Implementing Sound-Proof Framework

As a prerequisite to evaluating the *Sound-Danger* attack system, we first re-implemented Sound-Proof, as described in [17]. We implemented phone-side, server-side and browser-side applications as described below:

- *Phone Application:* We created an Android app that stays idle in the background and is automatically activated when a push message arrives. Google Cloud Messaging (GCM) is used to send a push message from the browser to the Android phone. When GCM push message arrives from the browser for recording, the Android app automatically gets activated and starts recording the ambient noise. The app stops recording as soon as another GCM push message arrives.
- *Web Server and Browser Application:* The server component is implemented using PHP while the browser component is implemented in HTML and JavaScript. Browser application has a simple button to control the recordings on the browser and on the phone. When the button is pressed to "start recording", the browser application sends GCM push message to the Android phone. If the button is pressed to stop recording, a "stop recording" GCM push message is sent to the Android phone. In the meantime, the browser application also starts recording ambient noise. Thus, the browser application has two main functions: (1) sending start/stop recording commands, i.e., GCM push messages, to the Android phone, and (2) recording ambient noise. In order to record ambient noise through the browser, we use HTML5 WebRTC API [15]. In particular, we use navigator.getUserMedia() API to access the local microphone from within the browser.

Time Synchronization: As the two devices (phone and terminal running browser application) may have two different local time clocks, our implementation, like Sound-Proof, requires the recordings from these devices to be synchronized. For this reason, both the phone and the browser applications run a simple time synchronization protocol with the web-server. Similar to Sound-Proof, the protocol is implemented over HTTP that allows each device to compute the time difference between the local time and the server time. Each device runs the time synchronization protocol while it is recording the ambient audio. Both devices compute their round-trip time delay ($\theta = t_2 - t_0$) and then clock difference ($\delta = t_2 - t_1 - \theta/2$) with the web-server. Here, t_0, t_1, and t_2 are the timestamps of the device's request transmission, the server's request reception/response transmission, and the device's response reception, respectively. During our offline analysis of audio samples, the recordings from each of the devices are adjusted taking into account the clock difference (δ) with the web-server.

2.3 Implementing and Testing Sound-Proof's Correlation Engine

Correlation Analysis: Correlation analysis between an audio pair is implemented in a similar fashion as Sound-Proof. That is, we used one-third octave band filtering and cross-correlation to get a similarity score of an audio pair, as described below:

- *One-third Octave Bands:* We divide the audio samples into different bands based on frequency. Each band covers a specific range of frequencies. A frequency is said to be an octave in width when the upper band frequency is twice the lower band frequency. A one-third octave band is defined as a frequency band whose upper band-edge frequency is equal to the lower band frequency multiplied by the cube root of two [20]. The audio spectrum from 20Hz to 20kHz can be divided into 32 one-third octave bands with the center frequency of 19th one-third octave band set to 1000Hz. The center frequency of the lowest band is 16Hz covering from 14.1Hz to 17.8Hz, while the center frequency of the highest band is 20kHz covering from 17.78kHz to 22.39kHz [25].

  Since we are targeting Sound-Proof, we divide the audio into the bands ranging from 50Hz to 4kHz. Sound-Proof utilizes only these set of bands, as these bands provided the best Equal Error Rate (EER) in the analysis reported in [17]. Hence, we only use the sixth band with the center frequency 50Hz to the twenty sixth band with the center frequency 4kHz, i.e. we consider only

twenty bands out of the thirty two available bands. We use twentieth order Butterworth bandpass filter [19] in MATLAB to split the audio samples into these bands.

- *Cross Correlation:* We use the same system that was implemented in [16] to correlate ambient noise. Sound-Proof also closely follows this system for calculating cross-correlation. We use standard cross-correlation function to measure the similarity between the time-based signals X_i and X_j. To calculate the similarity, we first normalize the signals according to their energy. Then, we calculate the correlation between each signal at different lags and use maximum correlation value. The correlation between two time-based signals X_i and X_j is measured as:

$$Corr(i,j) = max(CrossCorr(X_i, X_j)) \quad (1)$$

Sound-Proof also considers the lag to get the cross-correlation. This plays a major role to prevent attacks on the system when an attacker submits a similar audio sample as that in victim's environment, which may be separated by a certain lag. Sound-Proof has bound the lag l between 0 and l_{max}, where it sets l_{max} to 150ms. Hence, in our attack analysis, we also check the maximum cross-correlation of audio pairs with the time lag bound to 150ms. This lag value yielded a low EER in Sound-Proof's analysis reported in [17].

Data Collection and Experiments: We collected audio samples using the framework described in Section 2.2 at different locations such as lab/office, home, cafe, and library. We used Google Chrome on MacBook Air and Samsung Galaxy S V to record the audio samples using our implementation of Sound-Proof. We collected total of 525 audio pair samples and mix-matched them to get the correlation between each audio pair. The data collection experiment was approved by our University's IRB. The audio recordings were around 8 seconds long which were trimmed to 3 seconds after time synchronization for the correlation analysis (similar to [17]).

Similar to Sound-Proof, our implementation uses one-third octave band filtering and cross-correlation to calculate the similarity score between an audio pair, as described above. We use octave band filtering to split 3 second long audio recordings from both devices into 20 one-third octave bands. Maximum correlation is computed with the time-lag bound to 150ms between these audio pairs in their respective bands. The average correlation value obtained from these bands is the correlation between the audio pair.

The audio pairs which are co-recorded (recorded at the same location and at almost the same time) are labeled as True Positive (TP) and the rest are labeled as True Negatives (TN). Once the correlation values for each of the co-recorded audio pairs as well as non co-recorded audio pairs are calculated, we compute the FPR (False Positive Rate) and FNR (False Negative Rate) as a function of the correlation threshold. FPR for a given threshold defines the fraction of audio pairs (out of all audio pairs) which are not co-recorded but are classified as valid (TP) at that threshold, while FNR for a given threshold defines the fraction of co-recorded audio pairs (out of all audio pairs) that are classified as invalid (TN) at that threshold. Using FPR and FNR at different threshold values, we calculate EER (Equal Error Rate) and determine the optimal value of correlation threshold (T_c) at which FNR and FPR are equal.

From the data collected in our experiments, we obtained the optimal threshold T_c of 0.1524 yielding an EER of 0.1607. The correlation threshold set in our experiment is in line with the correlation threshold of Sound-Proof (0.13) [17]. Moreover, our other parameter settings are exactly the same as in Sound-Proof's implementation [17], i.e., using the audio samples each of length 3 seconds, filtering the audio samples into 20 different one-third octave bands, and cross-correlating the audio samples with time lag bound to 150ms. Since our correlation threshold is higher than that used in Sound-Proof's implementation, it means that attacking our implementation will be harder than attacking Sound-Proof's implementation reported in [17]. In other words, the *Sound-Danger* attack against our implementation (threshold 0.1524) would imply an attack against Sound-Proof's implementation (threshold 0.13) [17]. Nevertheless, we analyze the performance of *Sound-Danger* for different (higher) correlation threshold values in the attack analysis in Section 4, and show that the attacks still work well even at such higher thresholds.

3. ATTACKS

As the threat model of our *Sound-Danger* attack system suggests, we assume that the attacker is already in possession of the victim's username and password but is not co-located with the victim (i.e. victim's phone). The attacker's goal is to satisfy the second factor requirement, which is to fool the system into accepting the co-location of the *attacker's terminal* and the *victim's phone*.

We consider two types of attacks against Sound-Proof, both of which exploit the sounds generated by the phone itself. The first type of *Sound-Danger* attack is the active attack, where the attacker performs an activity by which a sound (a phone ringing tone or an app-based notification) would dominate the ambient audio around the victim's device. Since the attacker is aware of the audio produced at the phone's end, it can generate the same sound at its own surroundings (or feed the same sound programmatically to the browser) and succeed in proving the co-location with the phone. The second type of *Sound-Danger* attack is the passive attack, in which the attacker waits for the phone to create a previously known sound, specifically a morning alarm, at an opportune moment and then tries to generate the same noise at its local terminal. The steps taken by the attacker to target Sound-Proof are shown in Figure 1.

3.1 Active Attacks

In the *Sound-Danger* active attack scenarios, we assume that the attacker has already compromised the web service that uses Sound-Proof. Since account databases on such servers typically store other forms of user's data (e.g., phone number for the password recovery purposes), hacking into the server reveals other information that can be a used in the active attacks. Such data if not stored on the main server is assumed to be obtained with data aggregation attack through other services that user has an account with (which probably does not use 2FA).

Based on the type of information that the attacker possesses, we define (and later implement and evaluate) the following attacks:

Ringtone Attack: In this attack, the attacker predicts the ringtone (or vibration sound) that the victim has set for the received phone calls. The attacker calls the victim using the knowledge it has obtained from compromised account database. At the same time, the attacker attempts to login by entering the previously leaked victim's credentials to the login page via its own login terminal. To mimic the victim's ambient noise (now the sound of the ringtone), attacker plays the same ringtone audio at its location next to the login terminal.

Information known to the attacker: Victim's username and password, victim's phone number and victim's ringtone.

Task of the attacker: Ring the victim's phone and create same sound around the local login terminal.

App Notification Attack: In this attack, the attacker predicts the voice/messaging application running on the victim's phone and

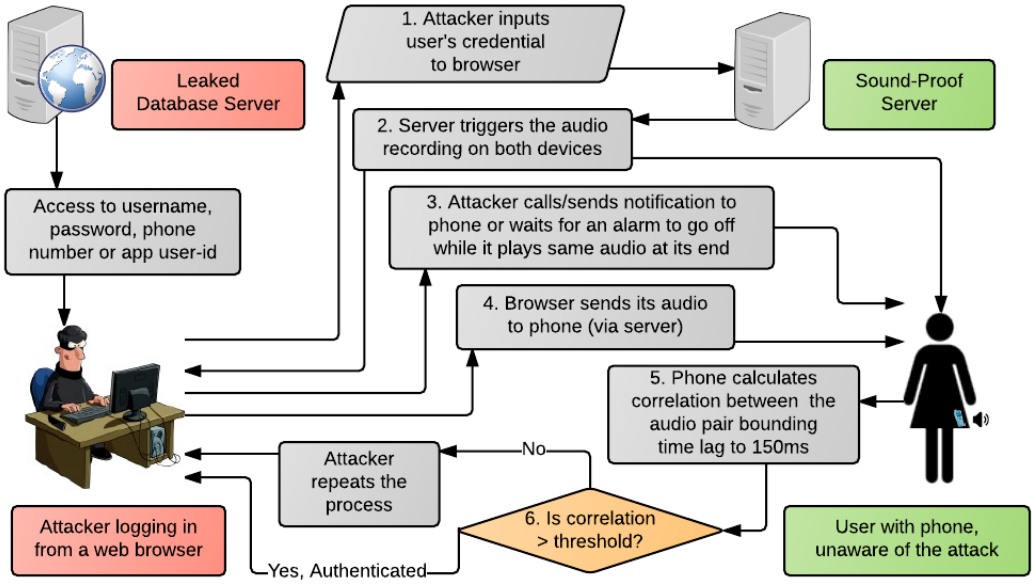


Figure 1: ***Sound-Danger* Attack Flowchart:** The attacker (human or bot) enforces the ambient audio to be highly similar at both (attacker's and victim's) ends by making calls or sending notifications to the victim's phone, or waiting for an alarm to go off at the victim's phone, and by simultaneously feeding the same sounds at its own end. The attacker would succeed in logging into the webservice, while the user may remain unaware of the attack or even when the attack is detected, the account may already have been compromised.

tries to activate the notification tone or ring tone of the application by communicating to the victim through the application. Since the user typically registers to many of the web services using phone number or user id, the attacker can contact the user on these applications either by their phone number (obtained by hacking the primary account database) or user id (possibly similar to the one registered with the primary service.

Examples of such applications are Google Voice, FaceTime, Skype, Facebook, WhatsApp and Viber. Calling or texting the user on these applications generates a default ringtone or notification tone that is known to the attacker and is usually not changed by the users. Hence, the attacker starts a login attempt to the primary service using the known credentials, and then contacts the user on any of the mentioned applications. At the same time, it plays the same ringtone or notification tone locally near the login terminal. The attacker would succeed since it regenerates the same ambient noise as the victim's phone locally (around the attacker's login terminal).

Information known to the attacker: Victim's username and password, victim's phone number, victim's installed application on the phone, victim's id with the application (same as phone number or primary username) and application ringtone.

Task of the attacker: Ring the victim's messaging application and create the same sound around the local login terminal.

Feasibility of Attacks: In all of our attacks above, the attacker would have to predict some information necessary to execute the attacks (e.g., the type of ringtone used by the victim). However, given predictable patterns and phone usage habits across users, this is not much of a problem for the attacker. In Section 5, we support these assumptions and claims made here, based on the data gathered from the participants in an online survey.

3.2 Passive Attacks

In the passive attack, the attacker predicts, or knows users' activity and launches the attack based on the knowledge it has from the users' profile gathered from the leaked database. Unlike the active attack, the attacker does not attempt to generate a sound at the user's side but only regenerates the same ambient sound that is supposed to be available at the user's side. Unlike the active attack, the passive attack does not alert the user by creating a sound, and hence can be repeatedly attempted without triggering suspicion.

Although there are different scenarios where an attacker can create a similar ambient noise to that at victim's side such as similar media attack (both attacker and victim are watching same media/TV channel, as also briefly considered in [17]), same event (both attacker and victim are attending a popular event), or similar vehicles sound (attacker knows when victim commutes and uses similar in-vehicle sound), we chose to exploit the ambient noise that is created by victim's phone itself such as alarms or morning reports. We believe that this attack has a higher chance of succeeding compared to other ambience-based passive attacks since the sound of the alarm of the phone will dominate the ambient sounds.

Alarm Attack: In this attack, the attacker knows the specific app which generates an audio at particular time of day such as the morning alarm. Attacker attempts to log in at a specific moment when such alarm is supposed to go off using the victim's credential. At the same moment, the attacker plays the alarm tone at its local login terminal to mimic the ambient noise around the victim's phone. In this attack apart from the victim's username and password, the victim's alarm time, alarm tone is also known to the attacker.

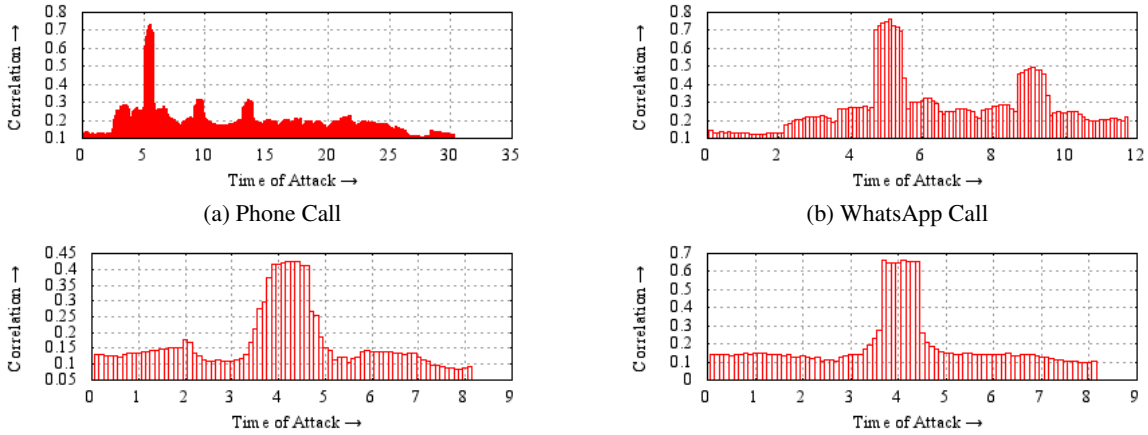
Information known to the attacker: Victim's username and password, victim's alarm ringtone and victim's timezone.

Task of the attacker: Create the same alarm sound around its local login terminal.

Feasibility of the Attack: Similar to the active attacks, the attacker would have to predict some information necessary to execute the attacks (e.g., the type of ringtone used by the victim and victim's timezone). However, given predictable patterns and phone usage habits across users, this is not challenging, as we demonstrate in Section 5 based on the results of an online survey.

3.3 Active vs. Passive Attacks

We introduced passive and active attacks, each of which has their



(a) Phone Call

(b) WhatsApp Call

(c) SMS Notification

(d) Facebook Notification

Figure 2: Change in correlations when an attacker makes call or sends notification via different apps at different point of time. In Figures a and b, the ringtone at the victim's side starts playing at the 5th second while in Figures c and d the notification audio goes off at 4th second as depicted by highest correlation. There is no audio at the victim's side from the ringtone when the attacker plays the respective audio at 0th second. The correlation values are higher when the ringer is ringing compared to that when there is no ringer i.e. before 2 sec in call.

own merits. With the passive alarm attack, it is very likely that the victim user would not notice the ongoing malicious login attempt. Therefore, the attacker might be able to repeat the attack repeatedly until it is successful. In case of active attacks, the sounds generated by the attacker on the user's phone (e.g., a phone ringing tone) could notify the user and seek her attention. However, only a few seconds of audio is enough for the Sound-Proof system to verify the co-presence of the phone and the terminal. Therefore, by the time the user attends to the phone (e.g., to pick up the call) and even when the user notices the malicious login attempt, the attack would have already succeeded and the user's account might have already been compromised.

Although Sound-Proof logs the login attempts on the device (as suggested in [17]), the users may leave their phones unattended, in purses or bags, might not be concerned about security or be diligent enough to the extent that they review the logs carefully and frequently. Extensive research literature in user-centered security shows that users may not pay attention to security notifications or heed security warnings and messages (e.g., [13, 24]). Moreover, relying upon the users to detect such attacks will break the "zero-effort" property of Sound-Proof. Furthermore, even if the logs were read and understood by the users, the attack may have already succeeded by the time suspicious activity is noticed.

4. ATTACK CORRELATION ANALYSIS

In this section, we show the correlation analysis of different attacks using *Sound-Danger* introduced in Section 3. In other words, we test the rate at which the attack samples (corresponding to attacker's browser and victim's phone) will be accepted as valid login attempts by our implementation of the Sound-Proof app. In our analysis, we used Samsung Galaxy S5 from Verizon as the victim's smartphone along with Google Chrome browser in MacBook Air (mid 2012) as attacker's terminal to perform the attack. The attacker makes calls or sends notification from LG G3 from Verizon or a computer to create an audio it desires at the victim's side.

To perform the attacks described in Section 3, the attacker follows the steps as illustrated in Figure 1. The attacker who performs such attacks tries to generate or predict a similar audio at the victim's side while it logs into the victim's account with the victim's credentials. The attacker has full control over the computer/browser that it is using. However, the attacker does not have any direct control over the victim's smartphone/app.

4.1 Ringtone and App Notification Attacks

To test our ringtone and app notification based active attacks, as described in Section 3, we use phone ringtone (call/SMS) as well as various other ringtones from some of the popular apps in Google Play Store, such as Facebook, WhatsApp Messenger, Viber, and Skype. We use default ringer of the app/phone call. Although some of the victims may have customized the ringtone for phone call or any other app, some of these apps do not allow users to customize the ringtone for calls or notifications. The primary difference between call and notification attack is that the ringtone audio is played longer for the call than it is for the notification.

Since the attacker does not have any direct control over the victim's phone, the attacker faces challenges due to two types of delays: (1) Sound-Proof recording delay, and (2) call/notification delay. Due to Sound-Proof recording delay, the attacker cannot perfectly guess at what time instance Sound-Proof starts recording audio from the victim's phone for the purpose of login. And, due to the call/notification delay, when the attacker makes call or sends notification to the victim's phone, the attacker also cannot make a perfect estimate to when the ringer sound will be played at the victim's side. To estimate these two delays, the attacker can run an experiment by making calls or sending notifications to itself and monitor the delays. Based on this, the attacker tries to synchronize the ringer being played at both sides as much as possible. We run and analyze the attacks assuming that the attacker knows when Sound-Proof starts recording at its end. This is a valid assumption since the attacker fully controls its terminal. Because of the delays mentioned above, during the actual attack, Sound-Proof may start recording either before or after the ringer goes off at the victim's side. We set forth to analyze how the correlation values change when the victim's ringer goes off at different points of time.

Attack Analysis: Sound-Proof compares 3-second long audio samples, as discussed in Section 2.3. Let us say the victim receives call/notification by an attacker at the n^{th} second. The attacker starts the authentication at the t^{th} second. This is when Sound-Proof starts recording at both sides. If Sound-Proof starts recording at t such that $t < n - 3$, Sound-Proof will not record any audio com-

ponent due to the ringer, and hence, there will be low correlation between the audio pair. When $t = n$, the correlation will be the highest as the attacker has fully synchronized the audio at its side with that at the victim's side. This pattern exhibited by the correlation values can be visualized in Figure 2. Here, the ringer for call goes off at the 5th second (Figures 2a and 2b) while the sound of the notification goes off at the 4th second (Figures 2c and 2d). Hence, the correlation is very low prior to the first 2 seconds. Now, when $t > n$, the correlation should drop as the two audio samples are not synchronized. However, the correlation after the ringer has started ringing ($t > n$) is higher than that when there was no audio at all ($t < (n - 3)$). Therefore, we know that the correlation increases and is reasonably high as long as there is some matching audio even if the two audio samples are not at perfect match or synchronized. The increase in the correlation values after $t > (n - 3)$ proves this pattern as depicted in Figures 2a and 2b.

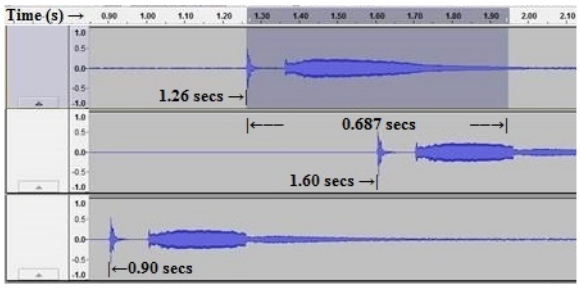


Figure 3: Analyzing Facebook notification audio sample (∼687ms long). The first audio sample (top) represents the audio played by the attacker. The second audio sample (middle) and the third audio sample (bottom) represent the audio recorded at victim's side when the notification audio rings after or before the attacker's side, respectively.

To further analyze this property, we choose Facebook notification attack instead of call attack for simplicity. We use Audacity[3] to analyze why the correlation in Figure 2d increases at $t = 3.4$ second and drops only after $t = 4.6$ second while the exact match occurs at 4th second as illustrated in Figure 3. The first audio signal in Figure 3 represents the audio signal that the attacker plays at its terminal which is 3 second long. The audio due to Facebook notification has audible signal of length 687ms which starts at 1.26th second. The second audio signal represents the audio recorded by the app at victim's terminal for $t = 4.6$ second where the notification ringtone goes off at 1.60th second. The third audio signal represents the audio recorded by the app at victim's terminal for $t = 3.4$ second where the notification ringtone goes off at 0.90th second in the 3 second long audio. From these three audio samples, we can see that whenever there is an overlap between the audible sounds, the correlation rises despite the time lag bound to 150ms. This is because when there is some audio, the correlation values from some of the 1/3-octave bands out of 20 bands increase, increasing the overall average correlation value.

Real Attacks and Success Rates: After analyzing the attack methodologies mentioned above, we set forth to perform the real attacks. To perform such attacks, the attacker needs to: (1) make a call/send a message via different apps to a victim's smartphone, (2) log in from a browser on a terminal which it fully controls, and (3) play the ringtone or a notification sound that the victim device may generate due to attacker's call/message. At the attacker's end, we used an LG G3 phone and a MacBook Air and, at the victim's end, we used a Samsung Galaxy S5 phone. The attacker first observed how long it takes for another device to ring in each different app when it makes a call to the corresponding apps. Then, the attacker made calls to the victim's device from those apps. The attacker tried to synchronize the ringtone played when it logs in from the Google Chrome browser on MacBook Air.

[3]http://www.audacityteam.org/

Table 1: Success rate of different types of attacks with respect to different correlation thresholds. Highlighted cells represent attack with success rate at least 90%.

	Attack Type	Tc= 0.1524	Tc = 0.18	Tc = 0.2
Active Call	Phone Call	81.82%	72.73%	63.64%
	Viber	100.00%	100.00%	90.00%
	WhatsApp	100.00%	100.00%	100.00%
	Facebook	100.00%	100.00%	72.73%
	Skype	41.67%	25.00%	16.67%
	Facetime	92.86%	57.14%	42.86%
	Vibration	85.42%	81.25%	72.92%
Passive Notification	SMS	64.71%	35.29%	17.65%
	Skype	85.71%	52.38%	19.05%
	WhatsApp	66.67%	33.33%	25.00%
	Viber	100.00%	92.86%	85.71%
Passive	Alarm	100.00%	90.00%	80.00%

We tested different attacks (active calls and notifications) against our implementation of the Sound-Proof system, and collected the audio samples stored in the victim's smartphone and the audio uploaded to the server from the attacker's browser. The success rates for our attacks with the correlation threshold $T_c = 0.1524$ for different types of attacks are shown in Table 1. We can see that many of our attacks were highly successful, including WhatsApp, Facebook and Viber calling, Viber notification and alarm. We also used vibration based attack where the noise is produced by a vibration of the phone instead of the phone playing a ringtone during a call. We placed the phone in different location such as on desk, inside a pocket, inside a bag, and in hand to see if the audio detected by the phone for such placements of the phone affects the attack success rate. Table 1 also shows the attack success rate when the threshold was increased to $T_c = 0.18$ and $T_c = 0.2$. When the correlation threshold is increased, the attack success rate decreases slightly as expected (although many attacks are still highly successful). We note that increasing the threshold would make the attacks little harder but at the expense of usability since even legitimate user may be prevented from logging in more frequently. Further experiments revealed that the attack success rate did not change even when the victim device was placed in front of a television with high volume. This confirmed our hypothesis that the sounds of the phone will dominate the sounds of the ambient surroundings.

4.2 Passive Alarm Attack

As described in Section 3.2, the attacker could execute the alarm attack at a specific time of the day (assuming the attacker knows when the alarm will go off at the victim's phone). Here, the attacker may know the timezone of the victim (through leaked password databases). Since the attacker has control over the browser and the device it is using at its end, the attacker can change its own timezone to be synchronized with that of the victims. To simulate this setting, we played LG G5's default alarm in front of the browser while the phone was set to create the alarm at a fixed time instance. Since both phones played the same alarm simultaneously at different ends, we achieved high correlation for the alarm attack reflecting to 100.00% success rate with $T_c = 0.1524$. The success rate decreased when we increase the correlation threshold, but we could still achieve 80% success rate. The result for this attack for different threshold is summarized in Table 1 (last row).

5. LEARNING POPULATION STATISTICS

To support the claims and assumptions made in Section 3, we conducted a survey by recruiting Amazon Mechanical Turk workers. The study was approved by our University's IRB. The participation in the study was strictly voluntary and participants could opt out of the study at any time. The survey took only about 10 minutes for each participant, for which they were compensated $0.7. In this section, we discuss the design and results from this survey.

5.1 Study Design

To better inform the design and execution of our attacks in the real-world, we asked the participants to answer several questions about their smartphone usage, including their habits of using the smartphones, the smartphone applications they use, and the ringtone and the notification sound they set or prefer to use. Below we summarize the set of questions we posed during the survey:

Demographic Information: We asked the participants about their gender, age, education, industry or field they belong to, country of residence, and their general computer knowledge.

Applications: We asked the participants about the applications installed and used on the phone, particularly those that generate a ringtone or a notification sound (e.g., Google Voice, FaceTime, Skype, Viber, Tango, ooVoo, LINE, WhatsApp, Telegram Messenger, Facebook, Phone, Text Message, Alarm Clock, and Calendar). Such applications are the primary target of the attacker in our *Sound-Danger* system.

Notifications and Sounds: We queried about the type of ringtone (e.g., default, vibrate, silent), and notification tones that the participants set for their applications in different situations and time of the day (e.g., while at work or asleep). If a particular popular ringtone is set often, the attacker can possibly attack many participants with our ringtone attack.

5.2 Study Results

General and Technical Background: We recruited 113 Amazon Mechanical Turk workers. Almost equal number of male (50.82%) and female (49.18%) users participated in the study. Although we did not set any geographical restriction, majority of the participants were from U.S.A (73%) and India (21%). The participants were falling in the age group 18 to 65, precisely 18-24 (12.30%), 25-34 (54.92%), 35-44 (22.95%), 45-54 (8.20%), and 55-64 (1.64%). The participants had high school (6.56%), college degree (25.41%), Associate degree (7.38%), Bachelor's degree (40%), Master's degree (1.72%), and Doctorate degree (2.46%). The participants were from different industrial background including: education, technical services, marketing, information technology, health care, and financial services. The demographic information shows that the survey covers a representative sample of real-world users.

The participants seem to have a reasonable general computer background as they ranked their general computer skill mostly as good (40%) and excellent (45%). We asked the users about their choice of username and password. The result shows that many users reuse the same username and/or password over multiple services. We will discuss in Section 7 that reusing the username may help an attacker who has compromised the web-service and knows the username of the victim to more successfully perform the ringing attack on an application (e.g., Skype) with the same username.

Habits of Using Smartphone and Apps: All of our participants said they have a smartphone. 80% of the participants said they carry their phone all the time and they have their phone connected to Internet always or most of the time. Most of the participants had voice, text and data services activated in their plan. Apple iPhone with over 39% and Samsung with 27% were the two most popular phone brands (other popular brands were: LG, Motorola, and HTC). The information obtained from this part of the survey shows that launching the introduced attack would be feasible, since many of the participants have a smartphone with voice/data/text plans that can be used by the attacker in an active attack.

Table 2: Popularity of instant messaging applications and the default ringtones for each app among participants.

Application	Popularity	Default Ringtone
Facebook	87%	67%
Skype	55%	63%
Google Voice	41%	66%
FaceTime	41%	83%
WhatsApp	36%	66%
Viber	22%	68%

Table 3: Popular ringtone setting for Samsung and iPhone.

Phone Brand	Location	Ringing Setting			
		Silent	Vibrate	Default	Custom
Apple	At home	5%	43%	45%	30%
	At work	20%	64%	16%	11%
	Asleep	25%	41%	30%	18%
Samsung	At home	13%	13%	40%	37%
	At work	20%	50%	13%	20%
	Asleep	27%	20%	37%	20%

All the participants in the survey said they frequently use phone calling and text messaging applications. The most popular instant messaging applications installed on participants' phones were: Facebook, Skype, Google Voice, FaceTime, WhatsApp and Viber. Application with higher popularity (such as Skype) especially those for which people use default ringing tone are the most attractive target applications for *Sound-Danger*. Table 2 summarizes the fraction of participants who use a given app and the default ringtone popularity among the participants who use the app.

We asked the participants about the kind of sounds they use for each application on their smartphone. The more predictable the ringtone is, the more successful the active ringtone attack would be in *Sound-Danger*. For the phone calls and text messaging, vibration and default ringtone are the most popular settings, silent and custom ringtone being less popular. It seems people tend to set the vibration at work, and set the default ringtone while at home. While still some participants tend to set custom ringtone for their phone calls and text messaging, custom ringtone is not that popular for instant messaging applications. More than half of the participants set the default ringtone for the instant messaging applications. Vibration is the second most popular setting for the instant messaging applications. The participants said that during a day they keep their phone on ringing mode or on vibrate mode about half the time, while they set it on silent mode only once in a while. These measures show that the *Sound-Danger* active attacks that target users by calling them on some popular instant messaging applications have a higher chance to succeed. Apart from the instant messaging application, launching a passive attack by playing the default alarm tone seems quite feasible. About half of the participants set the default alarm tone that the attacker can play locally at certain time of the day to mimic the sounds of the victim's phone. Table 3 summarizes the popular ringing setting of the two most popular phone brands, namely iPhone and Samsung among the participants, in three common situations: at home, at work and while asleep.

6. REAL-WORLD ATTACK STRATEGIES

The survey results in Section 5 help us devise real-world attack strategies and estimate the corresponding attack success rates. The

Table 4: ***Sound-Danger* Attack Strategy** ($T_c = 0.1524$)**:** The percentage of compromised users at the beginning of each attack round i is denoted as CN_i. Effective attack success rate (Eff_i) of the attack at each round i depends upon the particular type of device victim is using ($device$), the particular state of the device the attack is targeting ($state$), the iterative success rate for a number of login attempts (k) the attack will be repeated for (Itt), and the percentage of currently uncompromised users the attack is targeted towards (UN_i), as shown in Equation 3. In our calculations, $k = 3$ all throughout. The last column ($CN_i = 1 - UN_i$) shows the percentage of compromised users (CN) after each attack round. Before the start of the attack, i.e., at round 0, $CN_0 = 0\%$ ($UN_0 = 100\%$). When the attack applies to all devices (e.g., vibrational attacks), the device probability ($device$) is 100%, and when it applies to specific device types, iPhone and Samsung, the device probabilities are 39% and 27%, respectively. The highlighted cell represents the percentage of user accounts successfully compromised in eight rounds, which may finish in less than a day.

Attack Round (i)	**Attack Description**	**Probabilities**				$Eff_i(k=3)$	**Compromised User Accounts** ($CN_i = 1 - UN_i$)
		$device$	$state$	x	$Itt(k=3)$		
1	Vibration at work	100.00%	57.00%	85.40%	99.69%	56.82%	56.82%
2	iPhone call at work	39.00%	16.00%	81.80%	99.40%	2.68%	59.50%
3	Samsung call at work	27.00%	13.00%	81.80%	99.40%	1.41%	60.19%
4	Vibrate at night	100.00%	30.00%	85.40%	99.69%	11.69%	72.60%
5	iPhone call at night	39.00%	30.00%	81.80%	99.40%	3.19%	75.79%
6	Samsung call at night	27.00%	37.00%	81.80%	99.40%	2.40%	78.19%
7	iPhone alarm	39.00%	50.00%	100.00%	100.00%	3.19%	81.38%
8	Samsung alarm	27.00%	50.00%	100.00%	100.00%	1.88%	83.27%

effective success rate of the attack, for an adversary who does not know (but can guess) the user's behavior and their habits of using the phone, can be calculated by multiplying the usage probabilities we obtained from the survey by the success rates reported in our correlation analysis of the attacks (Section 4).

Preliminaries: Let us call the success rate of the correlation-based attack, as we presented in Table 1, as x (e.g., success rate of a ringing attack for an attacker who *knows* the victim's ringtone). Then, such attack would succeed with the probability $p = device \times state \times x$, where $device$ denotes the probability of owning a specific type of device (e.g., Apple's iPhone) and $state$ denotes the probability of the phone being in a particular state (e.g., default ringtone at home). Note that the attacker can make multiple login attempts at a given point of time to increase the chances of success. In this case, for k iterations of login, the "Iterative success rate" (termed $Itt(k)$), for a particular attack (with success rate x) can be calculated as:

$$Itt(k) = 1 - (1 - x)^k \tag{2}$$

The attacker repeats the above attack, with different attack variations, in multiple rounds. During each attack round i, the attacker performs the attack with k iterations targeting the remaining uncompromised users from round $i - 1$ (i.e., UN_{i-1}). Initially, no users have been compromised (i.e., $UN_0 = 100\%$). Thus, the "Effective attack success rate" $Eff_i(k)$, in round i with k iterations, which represents the fraction of users the attacker has compromised in round i, is given by:

$$Eff_i(k) = device \times state \times Itt(k) \times UN_{i-1} \tag{3}$$

Note that $device$, $state$ and $Itt(k)$ do not depend on the attack round per se, but rather they depend on the type of attack performed, i.e., these values remain the same even when the order in which the attack rounds are executed is changed. In contrast, Eff and UN depend upon the attack round.

A Concrete Sample Strategy: The attacker can devise a strategy to compromise the maximum number of victims by choosing any subset of the attack variations discussed in Section 3 and launching them in a particular order. In the rest of this section, we show a sample attack strategy based on our online survey results (Section 5) and a subset of our attacks (Table 1) in a specific order to compromise about 83% of user accounts in a total period of less than a day. Our attack strategy is summarized in Table 4. This is only a sample strategy for demonstrating the overall effectiveness of our attack. In practice, a real-world attacker can devise other strategies to maximize the impact of the attack based on the target user population under question.

As in our *Sound-Danger* attack model, we start with the assumption that the attacker has already obtained username, password, phone number and timezone of each of the target users by compromising a server and is trying to login to the victim user's account by: (1) entering the first authentication factor (username and the password), and (2) attacking the Sound-Proof application to prove the possession of the second factor (the phone). We also assume that the server throttles login attempt after three login trials to prevent a login brute force attack (a common practice employed by many web services). Therefore, we limit the attacker's login attempts to three, setting $k = 3$ in our attack strategy throughout. We test our attack strategy against our implementation of Sound-Proof with the threshold $T_c = 0.1524$.

We start with UN_0 equal to 100.00% of the user accounts (no account has yet been compromised). Since many of the users in our survey indicated that they keep their phones in vibration mode at work and this attack works irrespective of the device type, we attack such users in the first round. We know that 57.00% of the users have their device in vibration mode at work (Table 3), and the attack success rate x for the vibration attack is 85.42% (Table 1). This yields the iterative attack success rate (Itt) in this round to be 99.69%. Hence, the effective attack success rate (Eff_1) for this attack is $device \times state \times Itt \times UN_0$ = $100.00\% \times 57.00\% \times 99.69\% \times 100.00\%$ = 56.82%. This means that we can compromise 56.82% of the users with the vibration attack by just making three phone calls during work hours. The calculations and success rates for this round of the attack are summarized in Table 4, row 1. Compromising about 57% accounts in just one round, for example, right after the password database was leaked, would be a significant threat. The attacker may stop here, or continue to the next round to compromise more user accounts.

To attack the rest of the uncompromised users, $UN_1 = 43.18\%$, after the first round, we choose the next popular $device$ and $state$ combination based on Table 3. Through our survey, we observed that 39.00% of the users have iPhone. From Table 3, we know that 16.00% of the users keep their device under default ringtone at work. Therefore, in our strategy, the second attack round would involve the default ringtone call for the "iPhone at work" users, since we can have the maximum impact with this approach. From Table 1, the attack success rate for default ringtone x is 81.80%, which amounts to Itt equal to 99.40%. The effective attack success rate (Eff_2) for this round of the attack is $device \times state \times$

$Itt \times UN_1$ = 39.00% × 50.00% × 16.00% × 43.18% = 2.68%. Hence, after the second round of the attack, we have successfully compromised 59.50% of the user population (56.82% in the first round plus 2.68% in the second round). This attack round is summarized in Table 4, row 2. The attacker may continue for the next few rounds in a similar fashion, as shown in Table 4. In case of alarm attack (round 7 and 8), the attacker needs to guess when the alarm normally goes off at the victim's end. Our user survey reports that all of the users use alarm; however only 50% of them use default alarm ringtone. Now, for a morning alarm, we assume that users set their alarms at 5am, 6am, 7am, and 8am. For three attempts, the probability that the attacker plays the alarm simultaneously with the victim's alarm is 75% (3 out of 4). Hence, Eff_7 is calculated as $device \times state \times x \times UN_6$ = 39.00%×50.00%×100.00%×21.81%×**75%** = 3.19%. Similarly, we calculate Eff_8 using the same alarm guessing probability. As we can see, after the eighth round, our *Sound-Danger* system will have compromised a total of over 83% of the user accounts.

Since we started the attack "at work", and end it in the morning time (with the alarm attack), it is fair to say that all the rounds of the attacks will have finished over a period of less than a day. A persistent attacker may continue further the next day, perhaps trying other attacks at different points of time, and may gradually compromise almost all user accounts in few days.

Finally, we re-calculated the success rate for the above attack strategy against our implementation of Sound-Proof with threshold values higher than $T_c = 0.1524$. We found that even when we increase T_c, our attack strategy is still successful, by compromising 82.60% of the users with $T_c = 0.18$, and 81.52% with $T_c = 0.2$. Hence, our attack strategy remains robust to increased thresholdization, highlighting the overall vulnerability of Sound-Proof.

7. POTENTIAL MITIGATION

A natural defense against our attacks would be to disable the 2FA system in the scenario when a call or a notification is received (and the corresponding sounds are played by the phone), or when an alarm is triggered. Alternatively, the calls, notifications or alarms could be disabled when the 2FA login takes place. This approach is in line with the configurable feature of the iOS system to mute the device when an app is recording, or block recording when the device is playing sounds. However, such mitigation will prevent the user from receiving calls/notifications or setting alarms *while* logging into Sound-Proof enabled accounts, and could possibly degrade the usability of the phone system.

Another possible defense is to reduce the probability of guessing the phone sounds. This defense relies on the user to prevent the attack by picking ringtones that are difficult for the attacker to predict and possibly changing them frequently in order to stop the attacker from attempting an exhaustive search. The analysis of the user survey in Section 5 shows that many users set the default ringtone for the instant messaging applications (e.g., Skype or Facebook) that makes it easier for an active attacker to predict the sound.

Similar to the custom ringtone, combination of sounds and/or vibration is a possible defense mechanism. During our attack analysis, we noticed that the correlation reduces to below the threshold value when the notification sound is mixed with vibration and the attacker plays only the notification ringtone at its side. Simply combining the ringtone and the vibration at the attacker's side to mimic the audio at the victim's side does not work for the attacker as we noticed that the vibration started at different point when the ringtone starts playing at the victim's phone. Hence, for a successful attack, the combination should be precisely synced with the one at the victim's side (with the occurrence of vibration at the exact position in the audio), which seems unlikely.

Our user survey shows that many users reuse their username over multiple accounts. The security issues rising from reusing the password have been demonstrated previously (reuse of the password helps the attacker, who compromises one service, to compromise other accounts with the same password). In light of our attacks, the reuse of the username raises a similar issue as that of the password reuse. For example, an attacker who has compromised the web-service and knows the username of the victim might successfully perform the ringing attack on an application (e.g., Skype) with the same username. Picking a different username for each account prevents the attacker who has already compromised the server from trying to launch our active attacks.

Any of the above defenses possibly introduce certain usability issues. For example, users might prefer default notification tone over custom ringtone. Or reusing the username requires the user to remember multiple usernames associated with each account. Further study is required to understand how these possible usability issues may impact the user experience of the phone system while strengthening the security of Sound-Proof in the face of *Sound-Danger*.

8. DISCUSSION AND FUTURE WORK

Sound-Proof Demo Analysis: Karapanos et al. [17] have deployed Sound-Proof demo app and released apps for Android[4] and Apple[5]. We set forth to analyze how the demo app (version 1.6 on Android) performs against our attacks compared to our implementation of Sound-Proof. We observed that the demo app uses higher value of correlation threshold ($T_c = 0.2$) than the one reported in the paper [17] ($T_c = 0.13$). As we only had access to the app binary (and not source code), we could not directly figure out the values for other parameters deployed in the demo app.

Our evaluation showed that FNR of the demo app (benign setting) when the phone was kept beside the computer was quite high, at 27.91%. When the phone was kept inside a bag/purse, FNR increased to 50%. Compared to the results reported in [17], the higher FNR might have been due to the use of higher value of correlation threshold (and possibly other tighter parameters) than that reported in the paper.

We then attacked the demo app with one active attack and one passive attack. In the active attack trials, we made calls to the victim using WhatsApp. In the passive attack trials, we tested the demo app against alarm audio using two different devices (victim uses Samsung Galaxy S5 while attacker uses LG G3). FPR for the active attacks was found to be 38.46% while that for the passive attacks was 64.71%. The attack success rate on the demo app is less than that on our implementation of Sound-Proof. This may again be due to the fact that the demo app uses higher value of T_c (and possibly other stricter parameters). As shown in Table 1, the success rate for different attacks against our implementation of Sound-Proof decreased when Tc was increased. Moreover, we noticed that the average correlation provided by the demo app was relatively high (e.g., the average correlation for the alarm clock is 0.29 with 0.16 as minimum correlation). Hence, if the demo app had used the T_c reported in the paper [17] ($T_c = 0.13$), the alarm attacks would have been 100.00% successful.

Furthermore, we analyzed for which parameters in our implementation of Sound-Proof will produce similar results to that by Sound-Proof demo app. To this end, we recorded audio from two devices simultaneously using both apps. We collected 30 audio

[4]https://play.google.com/store/apps/details?id=ch.soundproof
[5]https://itunes.apple.com/us/app/sound-proof/id1069858990

instances and logged the correlation score from the demo app. We calculated the correlation results for different length of audio recorded (3s, 4s, 5s, and 6s) and for different threshold values (0.1524, 0.18 and 0.2). We found that when we compared 6 second long audio with $T_c = 0.2$, the scores from our app and the demo app had the maximum correlation (we used the alternative computation formula for Pearson's r [18] to calculate the correlation level). This suggests that the demo app is using 6-second long audio snippets rather than 3-second.

Overall, this analysis suggests that the Sound-Proof demo app uses much stricter parameters, which resulted in very high FNRs. Notably, even with this parametrization resulting in very low usability (high FNR), the *Sound-Danger* attack can still be successful against the demo app, further validating its feasibility as a viable real-work attack against Sound-Proof.

Attacking Sound-Proof Smartwatch Implementation: Sound-Proof is currently implemented with a smartphone as the second factor device. However, if Sound-Proof is implemented using other devices, such as smartwatches, as briefly discussed in [17], our *Sound-Danger* attacks will still work. When an attacker makes a call to the victim's phone, the victim's smartwatch would record the ambient audio containing the call's sound generated by the phone. This audio sample recorded by the smartwatch is similar to the audio samples recorded by the smartphone, hence these two audio (from phone or from watch) should yield similar correlation values as that between the audio samples of phone and terminal. If the smartwatch itself features a speaker and rings when a call or a notification is received, a similar attack will still apply. Android Wear 1.4 provides support for the devices with embedded speakers, such as Huawei watches and ASUS ZenWatch 2 available in Google Store. Moreover, our preliminary study on smartwatches shows that the current smartwatches do not feature high quality microphones as in phones and the correlation using the audio recorded by a smartwatch has higher error rates compared to that using the audio recorded by a phone. In the future, if smartwatches features better microphones along with speaker hardware and become standalone devices (no companion device, like a phone, needed), these smartwatches would still be vulnerable to our attacks.

Attacking Other Security Applications: As we will review in Section 9, there are different systems which use audio as one of the modalities to detect the co-presence of two devices. Co-presence detection of two devices has generally been proposed in the context of mobile payment applications to prevent relay attacks [16, 22, 26]. Further, another security scheme is aimed for pairing of devices based on common ambient audio signals [22]. However, the threat model for co-presence detection and device pairing systems differs from that for 2FA systems. These systems do not consider that the user information is leaked and that the attacker may already know the phone numbers and other specifics associated with the corresponding users. Further, our attacks become difficult against such co-presence detection systems as the attacker has to interact with the point-of-sale (POS) payment terminal in a retail store rather than the web browser under its full control.

If we extend the threat model of the above systems such that the attacker knows the phone number/user id of the user, then our attacks are applicable. A motivated attacker against co-presence detection systems [16, 26] can go near a POS terminal and generate a ringtone audio at its side (let's say in a jewelry store) while a colluding attacker makes call/sends notification to the victim's phone at the same time (let's say in a restaurant). This will break the co-presence detection system as audio at both ends can be dominated by the ringer's audio. In a similar vein, the security of device pairing systems based on ambient audio [22] can be compromised. Here, the attacker can perform an active attack to produce certain sounds which will then be used by the pairing system as a seed to generate a common secret between two devices. If the ringtone dominates the ambient noise then the seed used by victim's devices can match with the seed at the attacker's side.

9. RELATED WORK

Two-Factor Authentication: Most common and traditional form of 2FA employs hardware tokens such as RSA SecurID [21] and Yubico [27]. These hardware tokens are specialized devices used solely for the purpose of authentication. Such schemes require users to carry and interact with the token. These schemes may be expensive to deploy because the service provider must provide one such token per customer.

Many software tokens 2FA schemes are also available, including Google 2-Step Verification [14], Duo Push [12], and Celestix's HOTPin [3]. These schemes are both scalable and flexible as single personal device can be used with multiple services in such schemes. These schemes are also cost effective, since deploying software tokens are logistically much simpler. These schemes prompt the user with a push message on his phone with current login attempt information and the user interacts with his phone to authorize the login.

PhoneAuth [11] is a software token 2FA scheme that leverages Bluetooth communication between the browser and the phone, to eliminate user-phone interaction. The Bluetooth channel enables the server (through the browser) and the phone to run a challenge-response protocol which provides second authentication factor. This scheme requires browser to have Bluetooth communication capability which is currently not available on many browsers. Authy [10] is another approach that allows seamless 2FA using Bluetooth communication between the computer and the phone. However, Authy requires extra software to be installed on the computer.

Traditionally, these 2FA schemes increase resistance to online dictionary attacks. The work of [23] presented several 2FA that are enhanced to strengthen security against both online and offline attacks. The main idea underlying all their 2FA protocols is for the server to store a randomized hash of the password, $h = H(p,\ s)$, and for the device to store the corresponding random secret s. The authentication protocol checks whether the user types the correct password p and also that it can access the device that stores s.

SlickLogin [5] (recently acquired by Google) minimizes the user phone interaction. It employs near-ultrasounds to transfer the verification code. The notion is to generate unique near-ultrasounds for each login attempt and use the very non-audible audio to authorize the attempt. To verify user's identity, a website plays a uniquely generated, nearly-silent sound through the computer's speakers. An app running on the user's phone picks up the sound, analyzes it, and sends the signal back to the site's server. The server verifies the user with the possession of the phone as a second factor.

Co-Presence Detection: Similar to 2FA systems, co-presence detection systems use different medium to verify the co-presence of two devices. The common application of such co-presence detection system is to prevent relay attacks. Although relay attacks are usually targeted attacks, we discuss them here as some of the systems using ambient audio can be thwarted with our attacks.

Schurmann et al. [22] use ambient sound to establish a secure communication channel among devices based on similar audio patterns. They exploit ambient sound as the seed to generate a common secret for the secure information exchange and authentication.

Halevi et al. [16] use ambient audio to detect the co-presence of two devices in payment scenario. In an NFC transaction, they correlate the audio collected from two devices (NFC phone and NFC

reader) and validate if both devices are together. They mention that if the attacker manipulates physical environment, it may succeed in launching relay attack, however, the tampering with audio looks like a daunting task and such activity can be easily detected.

Truong et al. [26] use various Radio Frequency (RF) sensors such as Wi-Fi, Bluetooth and GPS besides audio to detect the co-presence of two devices. In their work, they report that when the system uses only audio to detect the co-presence between two devices, F-measure is 0.933 with 4.09% FPR in benign setting. However, in an adversarial setting, the FPR rises to 100.00% where the attacker manipulates the audio signal only. In their adversarial setting, the attacker captures the ambient information (audio) near device 1 and reproduces that information (audio) near device 2 without suppressing the ambient information sensed by device 2.

10. CONCLUSION

Zero-effort two-factor authentication is a compelling notion that may push two-factor authentication towards wide-scale adoption on the web. The idea of using ambient sounds to verify the user's possession of (or proximity to) the authentication token (phone) is intriguing, which aims to remove the human user from the loop of authentication (except of password entry). In this paper, we demonstrated that the ambient audio approach to zero-effort two-factor authentication is highly susceptible to a remote attack that makes the phone record its own predictable sounds in the form of a ringer, app-based notifications and alarms. Since these sounds are predictable, the attacker can generate the same (very similar or highly correlated) sounds at the browser's end under its full control and succeed in logging into the user's account. Further, by proactively collecting statistics about a given population (not a specific user), the attacker can devise a strategy that can allow the attacker to compromise a large fraction of users' accounts in a short span of time.

Since the attack exploits the sounds of the phones, one obvious defense would be to deliberately mute the phone at the time the login takes place. However, this may reduce user experience since important calls or notifications may be missed or delayed. Further work is necessary to assess this and other potential mitigation strategies we presented in the paper. Overall, our work serves to call the security of a prominent, deployment-ready zero-effort two-factor authentication scheme into question, highlight the tension between the security and usability of two-factor authentication and raise a demand for utmost care when attempting to design authentication approaches transparent to the human user.

Acknowledgments & Disclosure

We would like to thank CCS 2016 anonymous reviewers, and Abhishek Anand, Tzipora Halevi and Hugo Krawczyk for their useful feedback on a previous version of the paper. This work has been funded in part by NSF CNS-1526524 and CNS-1547350 grants.

We notified the authors of the Sound-Proof system [17] about the reported vulnerability. The authors acknowledged the vulnerability and indicated that they had applied several patches, in line with the potential mitigations suggested in Section 7. While these fixes may serve to defeat the vulnerability, the usability of the system may be reduced (as discussed in Section 7), which we believe is the fundamental challenge in the design of *zero-effort* authentication systems demanding continued future research.

References

[1] Anonymous leaks database from israeli musical act magazine site #opisrael. http://thehackernews.com/2012/12/anonymous-leaks-database-from-israeli.html.

[2] Bulgarian torrent tracker forum hacked and accused of collecting user ip. http://thehackernews.com/2012/11/bulgarian-torrent-tracker-forum-hacked.html.

[3] Celestix hotpin two factor authentication. http://www.celestixworks.com/HOTPin.asp.

[4] Coalition of law enforcement hacked & agents information leaked. http://thehackernews.com/2011/12/coalition-of-law-enforcement-hacked.html.

[5] Google acquires slicklogin, the sound-based password alternative. http://techcrunch.com/2014/02/16/google-acquires-slicklogin-the-sound-based-password-alternative/.

[6] Hacker leaks 250gb of nasa data, another group claims to hijack nasa drone. https://www.hackread.com/nasa-data-leaked-nasa-drone-hacked/.

[7] Snapchat hacked: 4.6 million user names, partial phone numbers leaked - ABC15 Arizona. http://www.abc15.com/news/local-news/water-cooler/snapchat-hacked-46-million-user-names-partial-phone-numbers-leaked.

[8] Sony europe hacked by lebanese hacker... again - naked security. https://nakedsecurity.sophos.com/2011/06/04/sony-europe-hacked-by-lebanese-hacker-again/.

[9] What's in the ashley madison database that hackers released online - quartz. http://qz.com/482875/whats-in-the-ashley-madison-database-that-hackers-released-online/.

[10] Authy Inc. Two-factor authentication - Authy. https://www.authy.com/.

[11] A. Czeskis, M. Dietz, T. Kohno, D. Wallach, and D. Balfanz. Strengthening user authentication through opportunistic cryptographic identity assertions. In *ACM conference on Computer and communications security*, 2012.

[12] Duo Security Inc. Easy, mobile two-factor authentication. https://duo.com/solutions/features/user-experience/easy-authentication.

[13] A. P. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner. Android permissions demystified. In *ACM conference on Computer and communications security*, 2011.

[14] Google Inc. Google 2-step verification. https://www.google.com/landing/2step/.

[15] Google Inc. WebRTC. https://webrtc.org/.

[16] T. Halevi, D. Ma, N. Saxena, and T. Xiang. Secure proximity detection for nfc devices based on ambient sensor data. In *European Symposium on Research in Computer Security*. 2012.

[17] N. Karapanos, C. Marforio, C. Soriente, and S. Capkun. Sound-proof: usable two-factor authentication based on ambient sound. In *USENIX Security Symposium*, 2015.

[18] D. Lane. Computing Pearson's r, 2003. http://cnx.org/contents/9zcGzDDu@4/Computing-Pearsons-r.

[19] MathWorks. Butterworth filter design. http://www.mathworks.com/help/signal/ref/butter.html.

[20] U. of Illinois at Urbana-Champaign. Center frequencies and high/low frequency limits for octave bands, 1/2- and 1/3-octave bands, 2011. https://courses.physics.illinois.edu/phys193/labs/octave_bands.pdf.

[21] RSA. Securid | rsa security token based authentication. https://www.rsa.com/en-us/products-services/identity-access-management/securid.

[22] D. Schurmann and S. Sigg. Secure communication based on ambient audio. *IEEE Transactions on Mobile Computing*, 12(2), 2013.

[23] M. Shirvanian, S. Jarecki, N. Saxena, and N. Nathan. Two-factor authentication resilient to server compromise using mix-bandwidth devices. In *Network and Distributed System Security Symposium*, 2014.

[24] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor. Crying wolf: An empirical study of ssl warning effectiveness. In *USENIX Security Symposium*, 2009.

[25] The Engineering ToolBox. Octave bands - frequency limits. http://www.engineeringtoolbox.com/octave-bands-frequency-limits-d_1602.html.

[26] H. T. T. Truong, X. Gao, B. Shrestha, N. Saxena, N. Asokan, and P. Nurmi. Comparing and fusing different sensor modalities for relay attack resistance in zero-interaction authentication. In *Pervasive Computing and Communications*, 2014.

[27] Yubico AB. Trust the net with yubikey strong two-factor authentication. https://www.yubico.com/.