Short Paper – What You See is Not What You Get: Leakage-Resilient Password Entry Schemes for Smart Glasses∗

Yan Li, Yao Cheng, Yingjiu Li, Robert H. Deng
School of Information Systems, Singapore Management University
{yan.li.2009, ycheng, yjli, robertdeng}@smu.edu.sg

ABSTRACT

Smart glasses are becoming popular for users to access various services such as email. To protect these services, password-based user authentication is widely used. Unfortunately, the password-based user authentication has inherent vulnerability against password leakage. Many efforts have been put on designing leakage-resilient password entry schemes on PCs and mobile phones with traditional input equipment including keyboards and touch screens. However, such traditional input equipment is not available on smart glasses. Existing password entry on smart glasses relies on additional PCs or mobile devices. Such solutions force users to switch between different systems, which causes interrupted experience and may lower the practicability and usability of smart glasses. In this paper, we propose a series of leakage-resilient password entry schemes on stand-alone smart glasses, which are gTapper, gRotator, and gTalker. These schemes ensure no leakage in password entry by breaking the correlation between the underlying password and the interaction observable to adversaries. They are practical in the sense that they only require a touch pad, a gyroscope, and a microphone which are commonly available on smart glasses. The usability of the proposed schemes is evaluated by user study under various test conditions which are common in users' daily usage. The results of our user study reveal that the proposed schemes are easy-to-use so that users enter their passwords within moderate time, at high accuracy, and in various situations.

Keywords

Leakage-resilient password entry; eavesdropping attack; smart glasses

1. INTRODUCTION

Nowadays, smart glasses are becoming popular commodities. Promising applications of smart glasses include Augmented Reality (AR) and Virtual Reality (VR). By wearing smart glasses, users can access various services, such as personal email and online so-cial network, in a hand-free manner at any place and at any time. To defeat unauthorized access to these services, password-based user authentication has been widely used for users' identity verification. Unfortunately, the password-based user authentication has its intrinsic vulnerability against password leakage. This threat is severe as smart glasses are usually used in various environments such as public areas and outdoors that are vulnerable to password leakage.

In order to thwart the threat of password leakage, prior research improves the leakage resilience of password entry on PCs and mobile devices [8, 13, 6]. Despite two decades of intensive research, most systems were broken soon after their proposals, while the remnants are very difficult to use (some schemes take up to 221 seconds per login attempt) [6, 17]. To achieve high security and usability, it is necessary to use a protected environment to hide certain user interaction during password entry [16]. However, these works require keyboards or touch screens as traditional input equipment. They are not suitable for password entry on smart glasses due to the following reasons. Firstly, the traditional input equipment is not available on the smart glasses due to compact and lightweight design. Secondly, the screen of the smart glasses is much smaller than monitors on PCs and touch screens on mobile devices. Thirdly, the smart glasses have limited hardware support. Lastly, the smart glasses need to be used in hand-free manner in various indoor and outdoor environments.

Most of existing password entry methods on smart glasses require users to type in their passwords via additional PCs or mobile devices. However, the additional devices may not be always available in certain scenarios such as public places and outdoors. Even worse, users are forced to switch between smart glasses and mobile devices for password entry. The interrupted user experience may lead to more errors and raise users' stress and anxiety [1]. On the other hand, new smart glasses features such as *Near-Eye-Display (NED)* screens have been exploited in a recent work for password entry on Google Glass [15]. While achieving reasonable usability, the proposed schemes in this work are subject to partial leakage of passwords as explained in Section 5. Designing leakage-resilient password systems for smart glasses remains a challenge today.

In this paper, we propose three password entry schemes on smart glasses, which are named *gTapper, gRotator,* and *gTalker*, respectively. The three proposed schemes are concise yet effective to achieve perfect leakage resilience of password entry with acceptable usability. The NED screen on smart glasses plays an important role in our schemes. The NED screen of smart glasses is a tiny optical instrument which reflects and magnifies the display to users' eyes [3]. The NED screen is fixed on the smart glass frame and placed physically close to the users' eyes [3]. Due to its compact size and physical proximity to the users' eyes, the NED screen can privately display information to users without being observed

by others. Thus the NED screen is used to deliver hidden information, which can break the correlation between the underlying password and the interaction observable to an adversary. For the practicability, our schemes only use a touch pad, a gyroscope, and a microphone which are commonly available on smart glasses. To enter password, the three schemes require the user to simply perform gestures on the touch pad, slightly rotate head, or speak numbers based on the hidden information displayed on the NED screen.

We implement the proposed schemes on Google Glass and evaluate them with a user study. The user study considers practical conditions related to *time pressure*. These conditions are used to simulate common situations in users' daily usage of password entry. The experimental results show that users enter their passwords by our schemes easily with moderate cost of time at high accuracy.

The contributions of the paper are summarized as follows.

- We propose and implement three leakage-resilient password entry schemes on smart glasses. Our schemes only require a touch pad, a gyroscope, and a microphone which are commonly available on smart glasses.

- We evaluate the usability of our proposed schemes by conducting a user study on Google Glass.

2. PRELIMINARIES

2.1 Background

Smart glasses, such as Google Glass by Google, are playing an important role in the promising applications of AR and VR. The smart glasses generally have a built-in operating system, such as Android Wear, which allows users to access various services, like email, online chatting, and personalized digital map. In order to support user interaction, a typical smart glasses is usually equipped with a tiny head-mounted NED screen and multiple sensors for collecting the information about users and environments, which include a small touch pad, a gyroscope, and a microphone. Due to the compact and lightweight design of the smart glasses, traditional input equipments, such as keyboard and touch screen, are not available on the smart glasses, as these traditional equipments are too heavy or too large to fit in the smart glasses.

In this work, Google Glass Explorer Edition (XE) 2 is chosen as the platform for our design, implementation, and evaluation. Google Glass is powered by Android Glass OS which has common functionalities of smart glasses and provides Android-based programmable API. Google Glass is equipped with a 0.5-inch NED screen (0.75 inch in length and 0.375 inch in width), as shown in Figure 1. The NED screen is placed approximately 1 inch away from the user's eye [3].
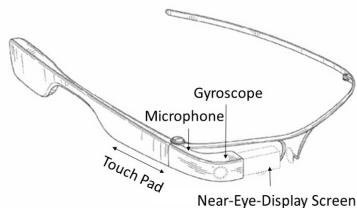

Figure 1: The design of Google Glass.

To enable user operation, multiple common sensors are embedded on Google Glass frame, including a touch pad, a gyroscope, and a microphone. In particular, the 3.25-inch long touch pad enables gesture-based user operation on Google Glass [3], which supports simple gestures such as finger tap and finger swipe. As a head-mounted device, Google Glass also tracks the user's head movement by motion sensor (i.e. a gyroscope) and responds accordingly. For example, the user can select items from a menu list on Google Glass by rotating his/her head. Another sensor, the microphone, enables voice commands on Google Glass which supports speech recognition-based user operation.

2.2 Threat Model

In order to protect various services, the legacy password based user authentication is widely adopted. However, the password-based user authentication has intrinsic vulnerability against *password leakage*. In password leakage attack, an adversary may disclose or infer a victim's password by observing and analyzing the victim's password entry process.

Due to the compact size and light weight of smart glasses, the existing password entry method employed by smart glasses commodities always requires users to enter their passwords via additional PCs with keyboards or mobile phones/tablets with touch screens. This makes the password entry on smart glasses especially vulnerable to the password leakage, as the passwords may be captured via various eavesdropping attacks such as recording camera, malware, and key logger. Depending on what information can be accessed by an adversary, the eavesdropping attacks to smart glasses can be categorized as *external* eavesdropping and *internal* eavesdropping.

In external eavesdropping attacks, an adversary can exploit channels outside smart glasses to obtain the victim's password. Based on the exploitable channels, we classify the external eavesdropping attacks into *vision*-based attacks, *motion*-based attacks, and *acoustics*-based attacks. Because smart glasses is often used in various environments such as open area and outdoor, it is more likely for the adversary to perform the attacks via these exploitable channels.

In vision-based attacks, an adversary can directly watch or video-tape a victim's password entry process such as observing victim's finger movements and head movements. The adversary may infer the password by analyzing the observed movements. Note that the vision-based attacks do not necessarily require physical proximity since video surveillance systems are widely deployed in some public places and even connected to the Internet. The existing password entry method used on smart glasses is especially vulnerable to the vision-based attacks, as users are required to enter their passwords in plaintext via an additional keyboard or touch screen.

In motion-based attacks, an adversary can estimate and track a victim's movements including finger movements and arm movements by additional equipment. Based on the estimated movements, the adversary may infer the input password. The adversary can remotely launch the attacks without any requirement on the physical proximity to the victim. The threat of the motion-based attacks becomes more serious as the boom of smart watches which can be used to estimate and track users' finger/arm movements.

Acoustics-based attacks allow an adversary to capture audio signals in the password entry process. Since speech recognition becomes popular and important on mobile phones and smart glasses, an adversary may have opportunities to launch effective acoustics-based attacks when voice commands are used in password entry.

On the other hand, internal eavesdropping is more powerful where an adversary can access internal states of smart glasses (e.g., memory and network packets) and infer users' passwords. Mitigations to the internal eavesdropping attacks can be securing executive environment. Our proposed schemes do not address this type of attacks because it is independent of password entry. Our work mainly focuses on the external eavesdropping which are directly linked to password leakage in password entry.

3. DESIGN OVERVIEW

In this section, we present the design of our leakage-resilient password entry schemes on smart glasses.

3.1 Design Goals

The design goals of our schemes can be explained from security, practicability, and usability perspectives.

Firstly, the schemes should minimize password leakage during the password entry process in various environments where an adversary has more opportunities to perform password attacks including the external eavesdropping as presented in Section 2.2. The adversary may infer the input password by analyzing the correlation between the observed information about the password entry and underlying password. Thus, it is important to decouple the link between the observation and the underlying input password.

Secondly, the schemes should be pervasively accessible on smart glasses in practical settings. In order to achieve this goal, no additional devices or external hardware should be involved as the devices or hardware may not be always available. The schemes should only use built-in hardware and functionalities that are commonly available on existing commodity smart glasses. However, the built-in hardware and functionalities on smart glasses are rather resource-limited. The design of the password entry schemes should meet the challenges of limited resources.

Thirdly, the schemes should preserve the benefits of legacy password in terms of usability [2]. Thus intuitive and simple operations are preferable for users perform during password entry.

3.2 Design of Password Entry Schemes

Due to the tiny size and physical proximity to a user's eye, the NED screen can be used to display information privately to a user without being noticed by others. Via the touch pad, gyroscope, and microphone, common user-device interaction channels on smart glasses include finger gestures, head movements, and human voice, respectively. In the proposed schemes, we take advantage of these hardware and interaction channels on smart glasses.

In our design, we assume a server and a user agree on a n-length password $pwd = (p_1, p_2, ..., p_n)$. During pass entry, for each $i \in \{1, 2, ..., n\}$, a hidden keypad $\Gamma_i(\cdot)$ is privately displayed to the user by the NED screen on smart glasses. The hidden keypad $\Gamma_i(\cdot)$ defines a random mapping $\Omega \rightarrow \Phi$ where Ω is the set of all elements contained in the password alphabet and Φ is the set of all candidate user operations via a user-device interaction channel. Note that in each round i, a new random mapping $\Gamma_i(\cdot)$ (i.e. a new hidden keypad) is drawn from the universal set of the candidate mappings $\Omega \rightarrow \Phi$ following uniform distribution. Secondly, given the hidden keypad $\Gamma_i(\cdot)$, in order to select the correct underlying password element p_i in pwd, the user needs to perform corresponding operations $op_i = \Gamma_i(p_i)$ via the interaction channel.

Since the hidden keypad is privately delivered to the user on the NED screen, it is difficult for adversaries to compromise this delivery channel. The observable response operation op_i by the user for the same password element is uniformly randomized due to the hidden random keypad $\Gamma_i(\cdot)$. Therefore, as long as $\Gamma_i(\cdot)$ is not disclosed, an adversary cannot obtain any useful information from op_i to infer the underline password element p_i through external eavesdropping attacks. The detailed security analysis will be provided later in this section.

During the password entry process, the types of the response operations and the designs of the hidden random mappings vary in different user-device interaction channels. According to the interaction channels, we design and implement three password entry schemes, which are named as gTapper, gRotator, and gTalker.

3.3 gTapper

gTapper utilizes a small touch pad which accepts users' finger gestures as input signals.

In gTapper, the password alphabet Ω is comprised of all single-digit numbers from 0 to 9. The hidden keypad of gTapper includes 10 numbers as shown in Figure 2(a). In each round i, gTapper randomly selects a number $s_i \in \{0, 1, 2, ..., 9\}$ and sets the focus on s_i (e.g. 5 is focused in Figure 2(a)) initially while the locations of the keys in the hidden keypad remain unchanged such that users can locate the keys easily and swiftly. A user can shift the focus onto the other numbers in descending or ascending order by swiping forward/backward on the touch pad with one finger, as shown in Figure 2(b). Thus the focus is shifted onto the number $(s_i - 1) \bmod 10$ by swiping forward once with one finger, and the focus is moved onto the number $(s_i + 1) \bmod 10$ by swiping backward once.

(a) Demonstration of gTapper. (b) The top left figure, bottom left figure, and right figure show one-finger operations, two-finger operations, and swiping directions, respectively.

Figure 2: Demonstration of gTapper and gestures

To enter a password element $p_i \in \{0, 1, 2, ..., 9\}$ in round i, a user needs to shift the focus onto the number p_i on the keypad from the initially focused number s_i by swiping forward or backward on the touch pad for op_i times, where $op_i = (s_i - p_i) \bmod 10$ or $op_i = (p_i - s_i) \bmod 10$.

An adversary can observe user operations on gTapper including swiping directions on the touch pad and the number of swiping operations. In each round, the adversary knows the number and the directions of shifts from an initially focused number to the element of the password. However, due to the protected hidden keypad, the adversary does not know what the initially focused number is and cannot infer the element of the password. The proof of the analysis is provided in Appendix A.1.

3.4 gRotator

gRotator relies on a gyroscope on smart glasses and allows the user to select and enter password elements with head rotation.

In gRotator, the alphabet of password Ω consists of all single-digit numbers from 0 to 9. The hidden keypad of gRotator contains two number screens, a small number screen $C_s = \{0, 1, 2, 3, 4\}$ and a big number screen $C_b = \{5, 6, 7, 8, 9\}$ as shown in Figure 3(a). At any time, only one number screen is displayed. In each round i, the positions of the five numbers in each number screen are randomly shuffled. One of the two number screens is randomly displayed as the initial screen. A user can change the number screen by swiping forward with one finger on the touch pad of the smart glasses if the i-th underlying password element is not included in the number screen currently displayed. In order to select a number, the user needs to rotate his/her head according to the position of the number on the screen. In particular, the user may select a number located at top, at bottom, on the left, on the right, or in the center by raising head, lowering head, rotating head towards left, rotating

head towards right, or heading upright, respectively, as shown in Figure 3(b).
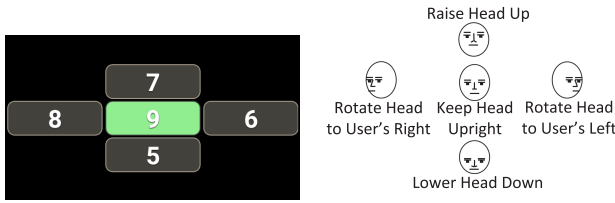
(a) Demonstration of gRotator.   (b) Head movements in gRotator

Figure 3: Demonstration of gRotator and head movements

To track the user's head movements, we estimate the user's head rotation based on the motion data generated by the gyroscope on smart glasses. The motion data include angular speeds on three orthogonal axes (i.e. axis X, axis Y, and axis Z) in the motion sensor coordinate system. The typical motion sensor coordinate system on smart glasses is defined relative to the NED screen. In the system, axis X is horizontal, pointing to the right. Axis Y is vertical, pointing to the up. Axis Z points toward the user's face. Based on the angular speed, we estimate the user's head rotation using a dead-reckoning algorithm [5]. Let $R_{t_i} = (r_{x,t_i}, r_{y,t_i}, r_{z,t_i})$ be the angular speed generated by the gyroscope at time t_i. The rotation angle along each axis can be calculated by the trapezoidal rule for integral approximation as follows

$$\theta_{s,t_i} = (r_{s,t_{i-1}} + r_{s,t_i}) \cdot (t_i - t_{i-1})/2 \qquad (1)$$

where $s \in \{x, y, z\}$. We calculate angle θ_{x,t_i} and angle θ_{y,t_i} of head rotation. The head rotation direction is determined by comparing the angle θ_{x,t_i} and angle θ_{y,t_i} with thresholds of ξ_v and ξ_h, respectively. The initial head pose is defined as the user's frontal face head pose. The estimation of head rotation direction H_{t_i} at time t_i is given below.

$$H_{t_i} = \begin{cases} \text{up} & \theta_{x,t_i} \leq (-1) \cdot \xi_v \text{ and } |\theta_{y,t_i}| < \xi_h \\ \text{down} & \theta_{x,t_i} \geq \xi_v \text{ and } |\theta_{y,t_i}| < \xi_h \\ \text{left} & \theta_{y,t_i} \geq \xi_h \text{ and } |\theta_{x,t_i}| < \xi_v \\ \text{right} & \theta_{y,t_i} \leq (-1) \cdot \xi_h \text{ and } |\theta_{x,t_i}| < \xi_v \\ \text{upright} & |\theta_{x,t_i}| < \xi_v \text{ and } |\theta_{y,t_i}| < \xi_h \end{cases} \qquad (2)$$

According to our pilot study, the best performance for determining head rotation directions can be achieved at $\xi_v = 15°$ and $\xi_h = 25°$. With the head rotation estimation, users can input password elements by performing corresponding head rotations.

The observed user operations by an adversary include swiping on the touch pad and head rotations. The adversary may know in each round whether the user changes the number screen displayed initially and know the exact positions of the underlying password elements located in the displayed number screen. However, as long as the hidden keypad is not disclosed, the adversary would not know which number screen is chosen by the user nor the mapping between the 5 numbers and the positions in the screen. So the adversary cannot infer any element of the underlying password. The proof is presented in Appendix A.2

3.5 gTalker

gTalker uses a built-in microphone and speech recognition techniques, by which smart glasses recognize the content and the meaning of the user's speech. The implementation of gTalker utilizes an offline speech recognition function available in Android API to recognize the user's speech, .

In gTalker, the alphabet of password is $\Omega = \{0, 1, 2, ..., 9\}$. The layout of the hidden keypad of gTalker is shown in Figure 4, where each white number p is followed by an underlined red number s. The hidden keypad consists of two keypads, one original keypad with all white numbers and one transformed keypad with all underlined red numbers. In each round i, the positions of white numbers remain unchanged while the positions of underlined red numbers shuffle randomly. For each white number $p_k = k$, we use s_{ik} to denote the corresponding underlined red number in round i, where $k \in \Omega$ and $s_{ik} \in \Omega$. For $\forall j, k \in \Omega$ and $j \neq k$, $s_{ij} \neq s_{ik}$ holds. In order to enter an underlying password element k via gTalker, a user may firstly identify the position of the white number p_k in the original keypad easily and quickly because the positions of the white numbers follow the traditional layout of number pad and remain unchanged. Then, the user should speak out the underlined red number s_{ik}. Finally, gTalker recognizes the number s_{ik} spoken by the user and enters p_k based the mapping between the original keypad and the transformed keypad in that round.

Figure 4: Demonstration of gTalker.

For gTalker, an adversary may know the number spoken by the user in each round. As long as the transformed keypad is not disclosed, the adversary does not know the random mapping between the original keypad and the transformed keypad. So the adversary cannot infer the element of the underlying password in each round. The proof is provided in Appendix A.3.

4. DATA COLLECTION AND EVALUATION

4.1 Data Collection

Our user study recruits 57 participants, including 29 males and 28 females whose ages range between 19 and 28. There are two parts in the user study.

Firstly, we briefly explain to each participant the purpose of our study and the operations and gestures used in the three proposed schemes, including gTapper, gRotator, and gTalker. We provide each participant with Google Glass and show the tutorials of each scheme and experiment processes in an interactive step-by-step manner before the experiments start.

In the second part, each participant is asked to use gTapper, gRotator, and gTalker as three *test groups*. The order of the three proposed schemes is randomized so as to avoid the learning effect which could have impact on the performance of the schemes. In each test group, the participant is required to memorize a password randomly generated at the beginning. The same password is used in the same test group. The password is set to a 6-digit PIN, whose strength has been commonly used protect important services such as online banking. In case that the participant forgets the assigned password, we provide a "show the password" function for the three schemes by swiping up with one finger on the touch pad.

In each test group, there are two tests under different *test conditions* which evaluate the impact of *time pressure* during password entry. We use these test conditions to mimic situations which may

occur commonly during users' password entry in daily use. In particular, the time pressure-related conditions are used to mimic the situations that the users may need to log into a system/service emergently and complete the password entry process within a time limit. To simulate the conditions related to time pressure, we introduce a timer to the tests. The *timer* is designed to give a participant time pressure by showing how much time is left for the current test. Based on timer, the following time pressure-related test conditions are used in our user study.

Normal condition: a participant is required to minimize the failure rate in a fixed number of login attempts where there is no time limit enforced. The number of login attempts is 3 in our tests.

Timed condition: a participant is required to achieve as many successful logins as possible within a fixed time limit. In the tests, the time limit for gTapper and gRotator is set to one minute while the time limit for gTalker is two minutes.

4.2 Experiment Results

Based on the user data, we measure the performance of the proposed schemes by two metrics which are average login time and login success rates. The *average login time* evaluates the speed of a login process while the *login success rate* evaluates the accuracy of login attempts. With the metrics, we analyze the impacts of the test conditions to users' password entry via the proposed schemes.
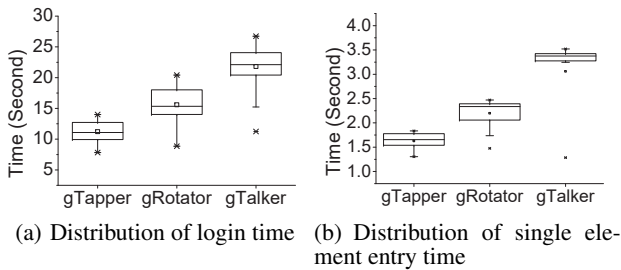

(a) Distribution of login time   (b) Distribution of single element entry time

Figure 5: Average login time and single element entry time

Figure 5(a) shows the average time for a successful login under normal condition for the three schemes. In particular, the average login time for gTapper is 11.2 seconds, which is generally shorter than the average login time for the other two schemes, which is 15.6 seconds for gRotator and 21.8 seconds for gTalker, respectively. In order to investigate the cause of the difference in the average login time, we examine the distribution of the *single element entry time* in the three schemes. As shown in Figure 5(b), the average time of single element entry for gTapper is 1.63 seconds, which is shorter than the average time of single element entry for gRotator and gTalker which are 2.20 seconds and 3.05 seconds, respectively. Making a tap takes shorter time than rotating head or speaking a number on Google Glass at the current stage. The average login success rates in the tests under normal condition are 98.3%, 98.2%, and 98.2% for gTapper, gRotator, and gTalker, respectively.

Figure 6 shows the comparison of the performance under the two conditions. Our results show that the participants' password entry processes become faster under time pressure. The average time for a successful login in the tests under timed condition is shorter than that under normal condition, which is 9.3 seconds for gTapper, 14.1 seconds for gRotator, and 20.1 seconds for gTalker. Similar to the average login time, the average time of a single element entry also becomes shorter, which is 1.36 seconds for gTapper, 1.98 seconds for gRotator, and 2.84 seconds for gTalker.

On the other hand, the average login success rates for gTapper and gRotator under timed condition are 96.3% and 94.5% respec-

tively, which are lightly lower than the average login success rates for the two schemes under normal condition, as shown in Figure 6(b). The average login success rate for gTalker under timed condition is 98.8%, which is close to the average login success rate under normal condition.
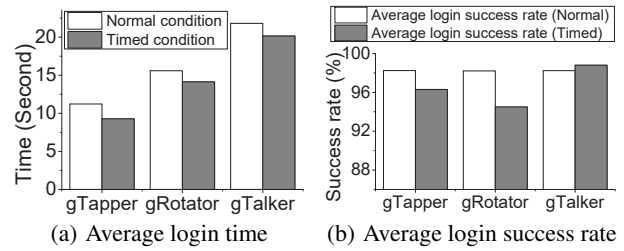

(a) Average login time   (b) Average login success rate

Figure 6: Impact of time pressure

For each scheme, a random password is generated to each participant. The participants are allowed to trigger "Show Password" functionality by swiping up with one finger on the touch pad if they forget their passwords. Our results show that no "Show Password" is triggered in the tests under normal condition for gTapper. For gRotator and gTalker, the average number of "Show Password" is generally higher in the tests appearing at first position than the tests at the other positions. In the tests appearing at the last position, the average number of "Show Password" decreases to only 0.1 times for gRotator and 0 times for gTalker. The results imply that our schemes do not incur significant interference on password recall.

4.2.1 Comparison with Existing Password Entry Method on Smart Glasses in Practice

We compare our password entry schemes with the existing real-world password entry method adopted by Google Glass based on the security-deployability-usability metrics [2]. The existing password entry method requires users to type in their passwords in plaintext via a standard keyboard on PCs or a touch screen on mobile phones.

The comparison results show that our schemes improve security by offering the benefits of *Resilient-to-Physical-Observation* and *No-Trusted-Third-Party* and partial benefit of *Resilient-to-Internal-Observation* because the schemes are secure against external eavesdropping attacks without relying on any trusted third party. For deployability, our schemes offer the benefit of *Negligible-Cost-per-User* since they use common built-in hardware on smart glasses only, while the existing method requires additional PC or mobile phone in connection to smart glasses. For usability, our schemes preserve most benefits of the existing method and offer the benefit of *Nothing-to-Carry* which is only partially offered by most of the existing methods due to the requirement of extra mobile phones or PCs. The detailed comparison results are provided in Appendix B

5. RELATED WORK

In this section, we summarize closely related work on leakage-resilient password entry, including external eavesdropping attacks, design principles, and schemes.

The external eavesdropping attacks against password entry include vision-based attack, motion-based attack, and acoustics-based attack. In a vision-based attack, an adversary directly view or record videos about password entry and infer the password based on various clues in the video [18, 11]. In a motion-based attack, an adversary may attack remotely by accessing arm-mounted motion

sensors on smart watch. Liu et al. [7] explored such threat by showing the feasibility of inferring users' PINs and typed texts through sensor data on smart watch. Wang et al. [12] further demonstrated that motion data from wrist-worn devices can be used to discriminate mm-level distances for users' hand movements during password entries. In an acoustics-based attack, an adversary may record audio signals about password entry and infer the password by analyzing the ringtones and keystroke acoustics [10].

Various design principles have been proposed for designing leakage-resilient password entry schemes. Roth et al. [9] proposed to use a cognitive trapdoor game to transform the knowledge of underlying password into obfuscated responses for password entry. Li et al. [6] pointed out several design principles, including time-variant responses, uncertainty, and balance. Recently, Yan et al. [16] included the design principles against brute force attacks and generic statistical attacks. These works indicate that it is necessary to use certain secure channel between a user and the device during password entry so as to achieve provable security and high usability. Our schemes are designed following these principles.

Many user authentication schemes have been proposed to achieve leakage resilience. Ginzburg et al. [4] proposed a scheme where a user needs to memorize a formula so that user authentication can be performed based on the calculated results from the formula. But the scheme burdens users with a high workload on mental memorization and calculation. Weinshal [13] designed a schemes, CAS, relying on the cognitive capability of human beings. In CAS, a user needs to identify about 30 secret pictures and find out a path among 80 pictures randomly displayed on the screen in a single round. An authentication attempt includes 10 rounds. According to the user study conducted in [13], CAS imposes a high usability cost, which may take up to 221 seconds per login attempt. A recent work, named CoverPad, was proposed to protect the password entry process on mobile phones and tablets [16] with acceptable usability. CoverPad leverages on a temporary secure channel between user and touch screen for transforming a password during password entry. CoverPad is designed for mobile phones and tablets with a standard touch screen. Due to the compact and lightweight design of smart glasses, it is difficult to apply the existing schemes on smart glasses.

Recently, Winkler et al. [14] proposed GlassUnlock which displays a randomized number pad on Google Glass and a blank keypad on mobile phone simultaneously. To enter a password, a user needs to press on the corresponding keys on mobile phone based on the number pad on Google Glass. A disadvantage of GlassUnlock is that users have to switch between Google Glass and mobile phone, which leads to interrupted usage experience. Another two schemes, namely VB-PIN and TB-PIN [15], improve GlassUnlock in terms of usability. The two schemes shuffle the number pad once and display it on the NED screen. To enter a password, a user is required to speak out the position of each password element in VB-PIN or tap the touch pad in TB-PIN. However, they are subject to partial password leakage as an adversary may know certain information about an input password. For example, if a password contains two or more identical password elements, the adversary can know the distribution of these identical password elements. In addition, the user may spend more time locating the correct password elements on a completely shuffled number pad. In comparison, our schemes do not require any additional device or hardware. It is easier for users to locate the password elements in the hidden keypad and perform simple and intuitive operations uninterruptedly with our schemes on smart glasses. Moreover, our schemes can achieve zero leakage (except password length) in password entry against the eavesdropping attacks.

6. CONCLUSION

In this paper, we propose three leakage-resilient password entry schemes for smart glasses. The schemes are proven to achieve zero leakage in password entry against the eavesdropping attacks. The schemes require no extra hardware beyond what is commonly available on smart glasses, nor additional device beyond smart glasses. The usability of our schemes is evaluated in a user study which shows that our schemes are easy to use in various scenarios.

7. REFERENCES

[1] P. D. Adamczyk and B. P. Bailey. If not now, when?: the effects of interruption at different moments within task execution. In *CHI 2004*, pages 271–278, 2004.

[2] J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *S&P 2012*, 2012.

[3] J. Dolcourt. http://www.cnet.com/news/everything-you-need-to-know-about-google-glass-faq.

[4] L. Ginzburg, P. Sitar, and G. K. Flanagin. User authentication system and method, May 25 2010. US Patent 7,725,712.

[5] I. Kamal. WFR, a dead reckoning robot–a practical application to understand the thoery, 2008.

[6] S. Li and H.-Y. Shum. Secure human-computer identification (interface) systems against peeping attacks: SecHCI. 2005.

[7] X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang. When good becomes evil: Keystroke inference with smartwatch. In *CCS 2015*, pages 1273–1285. ACM, 2015.

[8] T. Matsumoto and H. Imai. Human identification through insecure channel. In *EUROCRYPT'91*, pages 409–421, 1991.

[9] V. Roth, K. Richter, and R. Freidinger. A pin-entry method resilient against shoulder surfing. In *CCS 2004*, 2004.

[10] R. Schlegel, K. Zhang, X.-y. Zhou, M. Intwala, A. Kapadia, and X. Wang. Soundcomber: A stealthy and context-aware sound trojan for smartphones. In *NDSS*, 2011.

[11] J. Sun, X. Jin, Y. Chen, J. Zhang, R. Zhang, and Y. Zhang. Visible: Video-assisted keystroke inference from tablet backside motion. In *NDSS 2016*, 2016.

[12] C. Wang, X. Guo, Y. Wang, Y. Chen, and B. Liu. Friend or foe?: Your wearable devices reveal your personal pin. In *ASIACCS 2016*, 2016.

[13] D. Weinshall. Cognitive authentication schemes safe against spyware. In *S&P 2006*, pages 6–pp, 2006.

[14] C. Winkler, J. Gugenheimer, A. De Luca, G. Haas, P. Speidel, D. Dobbelstein, and E. Rukzio. Glass unlock: enhancing security of smartphone unlocking through leveraging a private near-eye display. In *CHI 2015*, 2015.

[15] D. K. Yadav, B. Ionascu, S. V. K. Ongole, A. Roy, and N. Memon. Design and analysis of shoulder surfing resistant pin based authentication mechanisms on google glass. In *FC 2015*, pages 281–297, 2015.

[16] Q. Yan, J. Han, Y. Li, and R. Deng. On limitations of designing usable leakage-resilient password systems: Attacks, principles and usability. In *NDSS 2012*, 2012.

[17] Q. Yan, J. Han, Y. Li, J. Zhou, and R. H. Deng. Designing leakage-resilient password entry on touchscreen mobile devices. In *ASIACCS 2013*, 2013.

[18] Q. Yue, Z. Ling, X. Fu, B. Liu, K. Ren, and W. Zhao. Blind recognition of touched keys on mobile devices. In *CCS*, 2014.

APPENDIX

A. SECURITY PROOF

A.1 Security Proof of gTapper

For gTapper, in each round i, an adversary knows the number and the directions of shifts. However, due to the protected hidden keypad, the adversary does not know what the initially focused number is and therefore cannot infer the i-th element of the password.

Proof: Given the user operation op_i in any round i, the initially focused number s_i, and any two elements p_x and p_y in a w-sized password alphabet (password alphabet $\{0, 1, 2, ..., 9\}$ with $w = 10$), let $Pr(op_i|p_x)$ and $Pr(op_i|p_y)$ be the probabilities for the operation op_i when the underlying password elements are p_x and p_y, respectively. If the observed user operation is swiping forward, we have $Pr(op_i|p_x) = Pr(op_i = s_i - p_x \bmod w) = Pr(s_i = p_x + op_i \bmod w) = Pr(s_i = C) = 1/w = Pr(op_i|p_y)$ for any i, x, and y, where C can by any integer randomly drawn from $\{0, 1, 2, ..., 9\}$. If the observed user operation is swiping backward, we have $Pr(op_i|p_x) = Pr(op_i = p_x - s_i \bmod w) = Pr(s_i = p_x - op_i \bmod w) = Pr(s_i = C) = 1/w = Pr(op_i|p_y)$ for any i, x, and y, where C can by any integer randomly drawn from $\{0, 1, 2, ..., 9\}$. Thus the sequence of the user operations observed by an adversary is equivalent to a random sequence. The adversary cannot distinguish the i-th element in the underlying password between any two elements in the password alphabet. $\square$

A.2 Security Proof of gRotator

For gRotator, an adversary may know in each round whether the user changes the number screen and know the exact positions of the underlying password elements in the displayed screen. But the adversary would not know which number screen is chosen nor the mapping between the numbers and the positions in the screen. So the adversary cannot infer any element of the underlying password.

Proof: Given the user operation op_i in any round i and any two elements p_x and p_y in 10-sized password alphabet $\{0, 1, 2, ..., 9\}$, let $Pr(op_i|p_x)$ and $Pr(op_i|p_y)$ be the probabilities for the operation op_i when the underlying password elements are p_x and p_y, respectively. Based on the design of gRotator, one of the two number screens C_s and C_b is randomly drawn from a uniform distribution and displayed initially (i.e. with a probability of $\frac{1}{2}$). Each number screen contains 5 numbers whose positions are randomly shuffled. Thus we have $Pr(op_i|p_x) = \frac{1}{2} \cdot Pr(p_x \in direction\ of\ op_i) = \frac{1}{2} \cdot P_4^4/P_5^5 = \frac{1}{2} \cdot 4!/5! = \frac{1}{10} = Pr(op_i|p_y)$ for any i, x, and y. That is, the adversary gains no advantage for distinguishing the i-th element in the underlying password between any two elements in the password alphabet by observing the user operations. $\square$

A.3 Security Proof of gTalker

For gTalker, an adversary may know the number spoken by the user in each round i. However, since transformed keypad is not disclosed, the adversary does not know the random mapping between the original keypad and the transformed keypad. Therefore, the adversary cannot infer the i-th element of the underlying password in each round i.

Proof: Given the number s_{ik} spoken by the user in any round i and any two elements p_x and p_y in a w-sized password alphabet ($w = 10$ in our implementation), let $Pr(s_{ik}|p_x)$ and $Pr(s_{ik}|p_y)$ be the probabilities for the observed number s_{ik} when the underlying password elements are p_x and p_y, respectively. Since the original keypad remains unchanged while the transformed keypad randomly shuffles in each round, we have $Pr(s_{ik}|p_x) = P_{w-1}^{w-1}/P_w^w = (w - 1)!/w! = 1/w = Pr(s_{ik}|p_y)$ for all i, x, and y. Thus the adversary gains no advantage for distinguishing the i-th element in the underlying password between any two elements in the password alphabet by observing the number spoken by the user. $\square$

B. COMPARISON RESULTS

Table 1 shows the comparison results. In particular, the metrics of security include the rows from "Resilient-to-Physical-Observation" to "Unlinkable". The metrics of deployability include the rows from "Accessible" to "Non-Proprietary". The metrics of usability include the rows from "Nothing-to-Carry" to "Easy-Recovery-from-Loss".

Table 1: Comparison between the proposed schemes and the existing password entry method on smart glasses using security-deployability-usability metrics [2] where ▲ indicates the benefit is offered, △ indicates the benefit is partially offered, while *blank* cell indicates the benefit is not offered

Metrics	Our schemes	Existing password entry on smart glasses
Resilient-to-Physical-Observation	▲	
Resilient-to-Targeted-Impersonation	△	△
Resilient-to-Internal-Observation	△	
Resilient-to-Theft	▲	▲
No-Trusted-Third-Party	▲	
Requiring-Explicit-Consent	▲	▲
Unlinkable	▲	▲
Accessible	▲	▲
Negligible-Cost-per-User	▲	△
Mature		▲
Non-Proprietary	▲	▲
Nothing-to-Carry	▲	△
Easy-to-Learn	▲	▲
Efficient-to-Use	△	▲
Infrequent-Errors	△	△
Easy-Recovery-from-Loss	▲	▲