WristUnlock: Secure and Usable Smartphone Unlocking with Wrist Wearables

Lili Zhang, Dianqi Han, Ang Li, Tao Li, Yan Zhang, and Yanchao Zhang
School of Electrical, Computer and Energy Engineering
Arizona State University, Tempe, Arizona, USA
{lilizhang,dqhan,anglee,tli,yanzhangyz,yczhang}@asu.edu

Abstract—**We propose WristUnlock, a novel technique that uses a wrist wearable to unlock a smartphone in a secure and usable fashion. WristUnlock explores both the physical proximity and secure Bluetooth connection between the smartphone and wrist wearable. There are two modes in WristUnlock with different security and usability features. In the WristRaise mode, the user raises his smartphone in his natural way with the same arm carrying the wrist wearable; the smartphone gets unlocked if the acceleration data on the smartphone and wrist wearable satisfy an anticipated relationship specific to the user himself. In the WristTouch mode, the wrist wearable sends a random number to the smartphone through both the Bluetooth channel and a touch-based physical channel; the smartphone gets unlocked if the numbers received from both channels are equal. We thoroughly analyze the security of WristUnlock and confirm its high efficacy through detailed experiments.**

Index Terms—**Smartphone authentication, security, usability**

I. Introduction

Unlocking is probably the most routine operation for smartphone users. In particular, average iPhone and Android users unlock their phones 80 and 110 times per day, respectively [1]. It is thus imperative to make smartphone unlockings both secure and usable. The security requirement means that illegal smartphone unlocking can be thwarted almost for certain, while the usability requirement indicates that legitimate users can very conveniently unlock their smartphones.

Current unlocking or authentication methods on COTS smartphones fall into two categories. In the first password-based category, a user needs to input his unlocking password, which can be a numeric PIN, an alphanumerical password, or an Android lock pattern. A complex password is more secure but also more difficult to memorize or input, and vice versa. In addition, it is quite challenging for senior citizens, children, users with fat fingers, and visual impairment people to memorize or input complex passwords on smartphones. In the second biometric-based category, a user relies on fingerprint or face authentication to unlock his phone, which is considered more convenient than inputting a password. However, there has been widespread news on successful hacking of face or finger authentication modules on smartphones. Once hacked, face or fingerprint biometrics are almost impossible to change and thus even less secure than the password-based approach. Moreover, face or fingerprint authentication may not be successful all the time, in which case the user still needs to input the password. Many older or cheap smartphones do not even have a face or finger authentication module. The lack of a secure and usable authentication method is possibly why over 50% smartphones are not password-protected [2] and thus highly vulnerable.

The growing prevalence of wrist wearables such as smartwatches and fitness trackers presents an excellent opportunity to unlock smartphones in a secure and usable fashion. In particular, the sales of smartwatches are forecast to reach 141 million units in 2018 [3]; the global fitness tracker market will grow from 20.88 billion USD in 2017 to 59.22 billion USD by 2023.[1] Almost every wrist wearable is paired via a secure Bluetooth channel with a host device which is most often a smartphone. In addition, many people often carry their smartphones along with their paired wrist wearables. The proximity of a wrist wearable to the paired smartphone and its physical attachment to the user's wrist can be considered a strong factor of physical security for exploration.

How can we explore wrist wearables for smart and usable smartphone unlocking? It is tempting to directly use wireless channels for verifying the physical proximity of a smartphone and its paired wrist wearable. In particular, most wrist wearables have a Bluetooth/WiFi/NFC interface for communicating with nearby smartphones. One may consider establishing a secure WiFi/Bluetooth/NFC connection between a smartphone and a wrist wearable. When a smartphone needs to be unlocked, it sends a cryptographic challenge via the wireless connection and automatically unlocks if an unforgeable cryptographic response is returned by the paired wrist wearable on time. This approach is vulnerable to an easily-conducted *relay attack* [4], which involves one attacker close to the smartphone owner and the other faraway attacker who possesses the stolen smartphone. The two attackers set up a stealthy, high-speed wireless channel to relay WiFi/Bluetooth/NFC signals between the stolen smartphone and wrist wearable that is still on the smartphone owner's wrist, so the smartphone can be unlocked by mistake. Since WiFi and Bluetooth transmission ranges can be several tens of meters or more, an attacker may not even need the relay channel if the WiFi/Bluetooth connection is used for proximity verification: he can just steal the smartphone and then stay sufficiently close to the legitimate user until the smartphone gets unlocked.

There have been attempts to use wrist wearables for smart-

[1]https://www.businesswire.com/news/home/20181206005489/en/Global-Fitness-Tracker-Market-Opportunities—Forecast

phone unlocking without using WiFi/Bluetooth/NFC channels. ShakeUnlock [5] asks a user to quickly shake his smartphone with the same hand wearing the smartwatch for about $2\,\mathrm{s}$, and the strong similarity between the acceleration data of the two devices is used to unlock the smartphone. An obvious drawback is that many users may feel socially awkward for frequent hand shaking in public environments. WearLock [6] exchanges controlled audio signals between the smartphone and smartwatch to verify their proximity within $1\,\mathrm{m}$, but it is still vulnerable to the relay attack as noted in [6].

In this paper, we propose WristUnlock, a novel technique that uses a wrist wearable to unlock a smartphone in a secure and usable fashion. WristUnlock explores both the physical proximity and secure Bluetooth connection between the smartphone and wrist wearable. There are two modes in WristUnlock with different security and usability features.

- WristRaise: This mode allows the user to unlock the smartphone in his natural way. Smartphone unlocking is often automatically triggered when the user raises the phone to look at it. Such raise-to-wake features are available in most iOS and Android devices. WristRaise asks the user to raise his smartphone with the same arm wearing the wearable, during which both devices generate a time series of acceleration data. The wrist wearable then transmits its acceleration data via the secure Bluetooth connection to the smartphone which then unlocks if an anticipated relationship does exist. In particular, let d_{wear} and d_{phone} represent the distance from the user's elbow to the internal accelerometers of the wrist wearable and smartphone, respectively. According to the principles of physics, the smartphone's acceleration to that of the wrist wearable is approximately $d_{\text{phone}}/d_{\text{wear}}$. We show that the relationship between the two acceleration time series can be explored for secure unlocking.
- WristTouch: This mode lets the user use his wrist wearable to touch and unlock his smartphone. When an unlocking event is triggered, the wrist wearable sends a random challenge through both the secure Bluetooth channel and a physical *vibration* channel between its internal vibrator and the smartphone's accelerometer. The smartphone unlocks itself if the random challenges from the Bluetooth and vibration channels match.

WristRaise is more usable than ShakeUnlock [5] because it does not require users to perform awkward quick device shaking. It is also more secure than WearLock [6], as it does not depend on the wireless channel alone to verify the proximity of the smartphone and wrist wearable.

We thoroughly analyze the security of WristRaise and WristTouch and also confirm their high efficacy with detailed user experiments on Google Nexus 7 and Huawei Watch 2. With WristRaise in place, the legitimate user and the attacker can unlock the smartphone with the success rate over 95% and below 5% in most cases. In addition, WristTouch can admit the legitimate user and reject the attacker in almost all cases.

The rest of this paper is organized as follows. Section II introduces the application context of SmartUnlock. Section III presents the adversary model. Section IV and Section V illustrate the WristRaise and WristTouch modes, respectively. Section VI analyzes the security of WristRaise and WristTouch. Section VII experimentally evaluate the security and usability of WristRaise and WristTouch. Section VIII briefs the related work. Section IX concludes this paper.

II. Application Context

WristUnlock targets the following common application scenario. A user owns both a smartphone and a wrist wearable which can be a smartwatch or less powerful fitness tracker with/without a display. The wrist wearable has been paired with the smartphone via a secure Bluetooth channel in the sense that every message transmission is encrypted and authenticated with cryptographic operations. The user carries his smartphone with the wearable on this wrist.

The smartphone is protected by a password that the user must input when powering up the smartphone or performing critical system operations such as system updates. As mentioned before, iPhone and Andriod users perform unlocking operations about 80 and 110 times per day. It is quite annoying for most users to input a complex but secure password for each unlocking operation. So many users may opt for a weaker password or even none at all. WristUnlock aims to address this dilemma by using the wrist wearable for secure and usable smartphone unlocking. Since the wrist wearable ought to be sufficiently close to the paired smartwatch in normal scenarios, the smartphone can safely get unlocked as long as its proximity to the paired wrist wearable can be verified.

WristUnlock is active when the user has the wearable on his wrist. In particular, wrist detection is a common functionality in wrist wearables to detect the attachment (detachment) of a wearable to (from) the wrist. WristUnlock does not require the wrist wearable to be password-protected, as many low-end devices do not have password-protection functionalities. Instead, each time the user puts the wearable on his wrist, he is prompted to input the smartphone password to activate WristUnlock, which can be combined with the common practice of unlocking a password-protected wrist wearable through the paired smartphone. This requirement is necessary to deal with the worst-case scenario that the attacker possesses both the smartphone and wrist wearable of the user. WristUnlock remains active until the wearable is removed from or worn too loosely on the user's wrist.

III. Adversary Model

The adversary attempts to unlock a smartphone he stole or found without causing the legitimate user's attention in a workplace, a supermarket, a subway, a night bar, or any other public environment. The smartphone is locked, which can be automatically done once it leaves the vicinity of the legitimate owner [7]. It is possible that the attacker possesses the paired wrist wearable as well, which requires him to have a much stronger capability. Since the legitimate user needs to input his smartphone password to activate WristUnlock, the security in this worst-case scenario falls back onto that of the smartphone's existing password mechanism. So we shall assume that the legitimate user's wearable is still on his wrist and that WristUnlock has been activated.

The adversary cannot bypass the existing password mechanism on the smartphone, so he can only try to break WristUnlock to unlock the smartphone. The adversary has perfect knowledge about how WristUnlock works and thus may attempt a few attack strategies including relay, co-located, and mimicry attacks. To facilitate the presentation and avoid redundant illustrations, we defer the explanation of such attack strategies and how WristUnlock defeat them to Section VI.

IV. WristUnlock: The WristRaise Mode

In this section, we illustrate the WristRaise mode of WristUnlock, which requires an internal accelerometer available on almost all COTS wrist wearables and smartphones. As shown in Fig. 1a, WristRaise lets the user unlock his smartphone by raising it in his natural way with the same arm carrying the wrist wearable. Most iOS and Android devices have a raise-to-wake feature which automatically wakes up the device screen as soon as the user lifts the device up. WristRaise is thus super natural and much less socially awkward than ShakeUnlock [5]. In what follows, we outline the design principle of WristRaise and then present the implementation details.

A. Design Principle

WristRaise is rooted in the principles of non-uniform circular motion in physics. In particular, when the user raises his arm with the smartphone in his hand and wearable on his wrist, the resulting gesture can be regarded as a non-uniform circular motion around the user's elbow. The internal accelerometers of the wrist wearable and smartphone experience the same angular velocity denoted by ω. Let d_{wear} and d_{phone} represent the distance from the user's elbow to the internal accelerometers of the wrist wearable and smartphone, respectively. The ratio $r = \frac{d_{\text{phone}}}{d_{\text{wear}}}$ can be considered a constant for each instance of arm raising which does not last long.

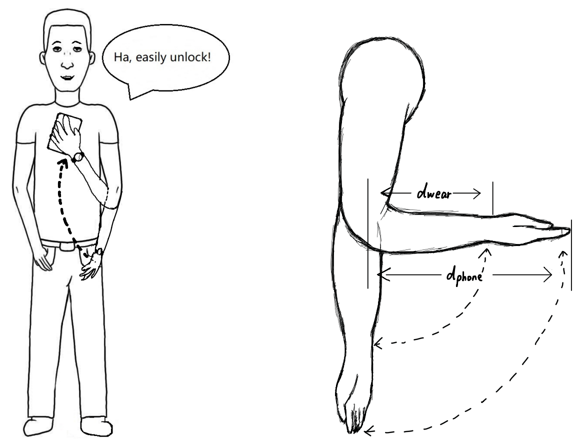
The tangential acceleration and centripetal acceleration at the wrist wearable's accelerometer can be approximated by $\alpha_{t,\text{wear}} = d_{\text{wear}}\frac{\Delta\omega}{\Delta t}$ and $\alpha_{c,\text{wear}} = d_{\text{wear}}\omega^2$. So the acceleration at the wrist wearable's accelerometer is approximately

$$\alpha_{\text{wear}} = \sqrt{\alpha_{t,\text{wear}}^2 + \alpha_{c,\text{wear}}^2} = d_{\text{wear}}\sqrt{(\frac{\Delta\omega}{\Delta t})^2 + \omega^4}.$$

Similarly, we can derive the acceleration at the smartphone's accelerometer as

$$\alpha_{\text{phone}} = \sqrt{\alpha_{t,\text{phone}}^2 + \alpha_{c,\text{phone}}^2} = d_{\text{phone}}\sqrt{(\frac{\Delta\omega}{\Delta t})^2 + \omega^4}.$$

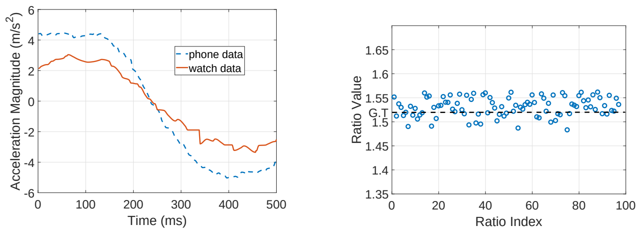
It follows that $\frac{\alpha_{\text{phone}}}{\alpha_{\text{wear}}} = \frac{d_{\text{phone}}}{d_{\text{wear}}} = r$. Therefore, we can tell whether the wrist wearable and smartphone are raised together on the same arm by continuously checking whether their *acceleration ratio* is approximately constant or has a very small variance around r.



(a) Typical usage. (b) Gesture decomposition.

Fig. 1: Illustration of WristUnlock.

We use a simple experiment to shed more light on the above principle. In the experiment, a volunteer raised a Google Nexus 7 smartphone with a Huawei Watch 2 on his wrist in his usual raise-to-wake way. The volunteer has $d_{\text{wear}} = 38\,\text{cm}$ and $d_{\text{phone}} = 25\,\text{cm}$, leading to $r = 1.52$ as the ground truth. After proper time alignment and data smoothing, we divided a sample duration of $0.5\,\text{s}$ into equal-length windows of $20\,\text{ms}$ with a $15\,\text{ms}$ overlap between any two adjacent windows. Then we computed the acceleration ratio in each window as the ratio of the smartphone's average acceleration to that of the wrist wearable. Fig. 2 plots the raw acceleration data of the smartphone and smartwatch as well as their acceleration ratio in each window. Except a few outliers, most acceleration ratios are bounded between [1.50, 1.55] with the mean and variance equal to 1.5309 and 0.0004, respectively. This preliminary result demonstrates the great potential of using the acceleration ratio to judge whether the smartphone and wrist wearable are raised with the same arm by the legitimate user.



(a) Raw acceleration data. (b) Acceleration ratio.

Fig. 2: Preliminary experimental result.

B. WristRaise Details

WristRaise involves two modules on the smartphone and wrist wearable, which communicate through the secure Bluetooth channel. It also consists of a light training phase and a testing phase that share significant similarity in data processing. So we illustrate the training phase first in what follows.

The first step in the training phase is to let the user decide how to trigger WristRaise. For example, the user can press the side button or rotate the phone from face down to up. Then the smartphone sends an activation signal via the Bluetooth channel to the wrist wearable. Then both devices start to monitor their own acceleration.

A design challenge in WristRaise is to find out the ground-truth acceleration ratio $r = \frac{d_{\text{phone}}}{d_{\text{wear}}}$. It is not user-friendly to directly measure d_{wear} and d_{phone} as in the motivating experiment, which requires knowing the approximate positions of the accelerometers on the smartphone and wrist wearable. In addition, the acceleration ratio may slightly change during the gesture due to body motion, measurement errors, etc. So we resort to the training phase for estimating r.

In the second step of the training phase, the user first holds his smartphone in a way that is easy to memorize and reproduce with the wrist wearable on the same arm. For example, if the user uses his left hand, he can use the thumb and index finger to touch particular buttons on the left and right sides, respectively. Such phone-holding gestures are fairly easy to reproduce especially for bigger-screen phones, but they are also user-specific and difficult for an attacker to emulate due to different holding habits and hand geometry. Then the user performs the raise-to-wake gesture in his natural way, i.e., lifting the phone towards his face. The user should try to keep his forearm straight and also minimize other body movement such as wrist rotation during this very short duration (usually shorter than $2\,\text{s}$). The user needs to do the gesture multiple times. Intuitively speaking, the more gestures performed, the more accurate the training parameters, and vice versa.

The third step is to pre-process the acceleration data from the smartphone and wrist wearable after each gesture. COTS smartphones and wrist wearables all have an internal 3D accelerometer that produces a time series of 3D acceleration vectors. Each acceleration value we use is the magnitude of the corresponding calibrated 3D acceleration vector. We use a simple sliding-window approach to detect the onset and end of the gesture. In particular, the onset (end) of the gesture is detected once the average acceleration within a sliding window of $20\,\text{ms}$ is above (below) a certain threshold. We assume that the smartphone and wrist wearable generate M and N time-indexed acceleration values for the gesture, respectively. The wrist wearable sends its acceleration vector to the more powerful smartphone for further processing. Note that M and N are usually different for each gesture.

The fourth step is to reconcile the difference between M and N and also mitigate the non-uniform distribution of data points in the two acceleration time series. More specifically, M is normally much larger than N, as the smartphone accelerometer usually has a much higher sampling frequency than that on a wrist wearable (e.g., about $230\,\text{Hz}$ on Google Nexus 7 vs. $100\,\text{Hz}$ on Huawei Watch 2). In addition, the acceleration values in both vectors are not uniformly distributed in time due to hardware and/or software constraints. To solve this problem, we apply piecewise Cubic Hermite Interpolation [8] with the same query points to both acceleration vectors. The resulting two new vectors contain approximately the same number of data points uniformly distributed in time.

The fifth step is to align the two new acceleration vectors in time. The timestamps associated with acceleration values are not helpful due to measurement errors and clock difference between the smartphone and wrist wearable. So we propose a new synchronization method. Since the two devices are raised together, we can expect them to reach their respective maximum speed almost simultaneously. Assume that the speed and acceleration are constant between adjacent data points in both vectors. We can easily locate the data index where the maximum speed is achieved in each vector and then align the two vectors at the two indexes. Next, we retain the same maximum possible number of data points on both sides of the maximum speed index in both vectors for the final use. In other words, the two vectors finally contain the same number of synchronized acceleration data points.

The sixth step is to compute a series of acceleration ratios with a sliding-window approach. In particular, we choose the sliding window that can contain 20 data points with a forwarding step of 5, so two adjacent sliding windows have 15 common data points for both acceleration vectors. Then we compute the ratio of the smartphone's average acceleration to that of the wrist wearable in each window, leading to a series of acceleration ratios.

The last step is to aggregate the series of acceleration ratios for all the gestures performed by the user in the training phase. In this paper, we choose two simple methods to derive the average acceleration ration $\hat{r}$ and compare their performance with experiments. In the first method, $\hat{r}$ is the average of all the acceleration ratios. In the second method, we obtain a discrete distribution of the acceleration ratios and then locate the interval with the maximum number of data points whose average is used as $\hat{r}$.

In each testing phase, the user triggers WristRaise and then performs his usual raise-to-wake gesture. Then his wrist wearable transmits the detected acceleration data via the Bluetooth link to the smartphone where a sequence of acceleration ratios are generated in the same way as in the training phase. Assume that the testing array consists of λ acceleration ratios denoted by $\{r_i\}_{i=1}^{\lambda}$. Then we compute a decision metric

$$D = \sqrt{\frac{\sum_{i=1}^{\lambda}(r_i - \mu)^2}{\lambda}}.$$

If D is below a distance threshold θ, the smartphone gets unlocked. θ can be obtained through sophisticated machine learning methods. If the user fails to unlock the smartphone after a threshold number of tries, the password-based mechanism is invoked as the fall-back method, which is similar to the case when face or finger authentication fails.

More sophisticated statistical methods can be used to aggregate the training data and verify the consistency of the acceleration vector in the testing phase. We leave further investigation on this matter to future work.

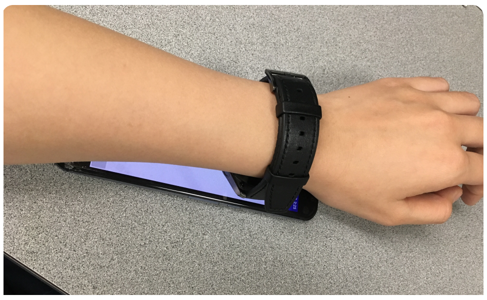


Fig. 3: A typical application scenario of WristTouch.

V. WristUnlock: The WristTouch Mode

In this section, we illustrate the WristTouch mode of WristUnlock. As illustrated in Fig. 3, WristTouch lets the user unlock his smartphone by touching it with his wrist wearable. During this process, the wrist wearable sends a random number to the smartphone through both the Bluetooth channel and the touch-based physical channel. The smartphone gets unlocked if the numbers received from both channels are equal, which confirms the unforgeable touch event. WristTouch is more secure than WristRaise with slight sacrifice in usability. In what follows, we outline the design principle of WristTouch and then detail the implementation details.

A. Design Principle

WristTouch explores the secure attachment of the wrist wearable to the legitimate user and its secure Bluetooth pairing to the smartphone in a different way from WristRaise. It lets the wrist wearable send a random challenge through the Bluetooth channel and also an auxiliary channel to the smartphone. If the messages received from the two channels match, the smartphone gets unlocked. The Bluetooth channel is secure in the sense that the messages transmitted over it can be cryptographically encrypted and authenticated, but it is vulnerable to the relay attack. In other words, an authenticated message arriving through the Bluetooth channel cannot assure the smartphone that it is adjacent to the wrist wearable which is supposedly possessed by the legitimate user. The auxiliary channel aims to provide this guarantee.

How should we establish the auxiliary channel? Wearlock [6] uses the audio channel which is subject to both relay and co-located attacks (cf. Section VI). Another option is to let the wrist wearable display an image or a QR code in which the random challenge is embedded. The smartphone then scans the image or QR code via its camera to decode the embedded number for comparison with the one received via the Bluetooth channel. This optical channel has been widely explored to pair wrist wearables with smartphones, but it requires the wrist wearable to have a display which may not be available on low-end devices such as fitness trackers. In addition, the optical channel requires the user to aim the smartphone camera at the image or QR code, which is infeasible or too demanding for visually impaired users. So we choose to establish a *touch-based vibration channel* based on the vibrator available on most wrist wearables and the accelerometer on the smartphone. In particular, the user uses the wrist wearable to touch the smartphone for a short duration (say, $2\,\mathrm{s}$), during which the wrist wearable activates its internal vibrator according to the pattern determined by the random challenge. Then the smartphone decodes the random challenge through its accelerometer data. This vibration channel is obviously robust to relay and co-located attacks, as the attacker would have no way to receive the random challenge other than touching the stolen smartphone against the wrist wearable. More security analysis is postponed to Section VI.

One may wonder why we do not let the smartphone transmit to the wrist wearable via the touch-based vibration channel. Our design choice is in view of the diverse capabilities of the smartphone and wrist wearable. In particular, the accelerometers on typical smartphones have much higher sampling rates than those on wrist wearables, e.g., about $230\,\mathrm{Hz}$ on Google Nexus 7 in contrast to about $100\,\mathrm{Hz}$ on Huawei Watch 2. Smartphones also have more resources to implement sophisticated filters to mitigate noise and interference. So it makes more sense to designate the smartphone as the receiver to sense the vibrations and decode the random challenge from the wrist wearable in a much more reliable way.

B. WristTouch Details

WristTouch is triggered when the user touches the smartphone with his wrist wearable and also presses some side phone button(s) that are predetermined in user enrollment. Then the smartphone sends a probe message via the Bluetooth channel to the wrist wearable.

Once receiving the probe message, the wrist wearable sends a random number s of n bits to the smartphone via both the Bluetooth and vibration channels, where n is a security parameter. The larger n, the more resilient WristTouch to random guessing attacks to be discussed in Section VI. The frame sent over the vibration channel comprises a preamble, the payload, and an ending flag in sequence. The preamble is for synchronization and consists of m repetitions of “11110” where m is a system parameter. The larger m, the easier synchronization between the wrist wearable and smartphone, and vice versa. In contrast, the ending flag is just “11110”. The payload corresponds to s after bit stuffing to prevent the occurrence of “11110”. In particular, an extra 0-bit is inserted after any three consecutive 1-bits in s. So the payload length is no less than n bits. We use the simplest on-off keying (OOK) modulation scheme. More specifically, the wrist wearable vibrates and pauses for T seconds for a bit-1 and a bit-0, respectively, where T is a system parameter. In our experiments, we found that $T = 200\,\mathrm{ms}$ suffices to achieve good transmission performance.

The smartphone keeps monitoring its acceleration data to detect the frame. During the execution of WristTouch, the phone should lay still on a flat surface like a table or be hand-held by the user when he stands, walks, runs, or is in a vehicle. Although the user should try the best to keep the phone still in the latter case, subtle human motion is inevitable and may affect the acceleration data. We used a Google Nexus 7 smartphone to collect the acceleration data in four representative contexts with a sampling rate of about $230\,\text{Hz}$: a Huawei Watch 2 vibrated on the phone front; a volunteer held the phone while walking, taking an elevator, and sitting in a moving car. After performing FFT, we found that the four contexts resulted in acceleration signals in [20, 50] Hz, $< 10\,\text{Hz}$, $< 10\,\text{Hz}$, and $< 15\,\text{Hz}$, respectively. So we pass the raw acceleration data through a Butterworth filter to retain the data mainly caused by the wrist wearable's vibrations. For this purpose, a one-time calibration process during user enrollment is needed to obtain the cutoff frequencies just as in our preceding experiment.

Then we use a sliding-window approach to decode the filtered acceleration data. Fig. 4 shows an example with Google Nexus 7 and Huawei Watch 2. The vibration interval $T = 200\,\text{ms}$, and the sliding-window size is $50\,\text{ms}$ with a forwarding step of $25\,\text{ms}$. We use a variable BIT to indicate the current bit which starts from zero. There are three cases for each sliding window: (1) if the data variance is above some threshold with BIT equal to zero, the phone records the starting time of this window and changes BIT to one; (2) if the data variance is blow the threshold with BIT equal to one, the phone records the starting time of this window and changes BIT to zero; (3) if the data variance is above (below) some threshold with BIT equal to one (zero), the window slides forward. In the end, the phone records the time segments with and without vibrations for bit-1s and bit-0s, respectively. Then we divide each time segment by T and round the result to the nearest integer, which indicates the number of bit-1s or bit-0s in that segment. In this example, we obtained the bit string 1100010001, which was exactly sent by the wrist watch.

The smartphone proceeds to process the bit array to identify and verify the random challenge s. It first looks for "11110" flags in the front of the big array, the last of which marks the beginning of the payload. The smartphone may miss some initial bits in the preamble due to synchronization issues, so we let the preamble consist of m "11110" flags. Then the smartphone locates the "11110" flag in the end of the bit array, which marks the end of the payload. The random number s is finally recovered from the payload after possible bit de-stuffing. If it equals the one received via the Bluetooth channel, the smartphone gets unlocked. To deal with possible bit errors on the vibration channel, we can relax the perfect-match constraint by claiming a successful match as long as the number of common bits in the two numbers exceeds a system threshold, which can be obtained via simple training. If the user fails after a few attempts, the password-based mechanisms is invoked as the fall-back authentication method.

VI. Security Analysis

In this section, we analyze the resilience of WristUnlock (including WristRaise and WristTouch) to common attacks.

Relay and co-located attacks. The relay attack involves two attackers very close to and far away from the victim user, respectively. The smartphone is possessed by the faraway attacker. The attackers set up a stealthy, low-latency wireless channel such as WiFi for relaying Bluetooth signals between the user's wrist wearable and the smartphone possessed by the faraway attacker. Under the relay attack, the smartphone and wrist wearable are outside each other's Bluetooth transmission range but can still exchange Bluetooth signals via the adversarial wireless link. In contrast, the co-located attack involves a single attacker who possesses the smartphone and stays sufficiently close to the user such that the smartphone and wrist wearable can directly exchange Bluetooth signals. Since the Bluetooth communication range can be several tens of meters, the co-located attacker may not risk exposing himself especially in a crowded public environment such as a subway.

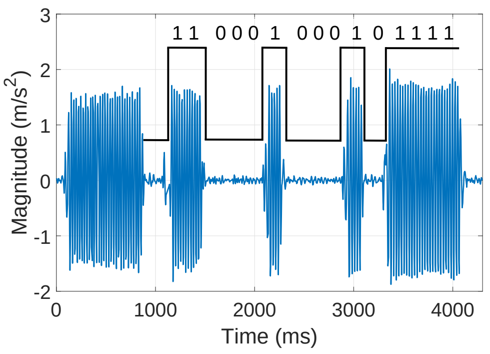


Fig. 4: Decoding acceleration data into bits.

With WristRaise in place, the smartphone cannot produce acceleration data to match those of the wrist wearable with regard to the anticipated acceleration ratio. If WristTouch is used, the smartphone cannot detect the random challenge transmitted via the touch-based vibration channel unless the attacker risks exposing himself by touching the smartphone against the user's wrist watch. So both WristRaise and WristTouch are immune to the relay and co-located attacks.

Mimicry attack. In this attack, the attacker uses the possessed smartphone to send an activation signal to the wrist wearable and then synchronizes his movement with the victim. The attacker needs to observe the victim's behavior either directly or through an accomplice very close to the victim.

For the WristRaise mode, the attacker has to wait until the user performs the raise-to-wake gesture and then attempts to raise the smartphone in the same way. WristRaise is robust to the mimicry attack for two reasons. First, the user may never perform the raise-to-wake gesture without holding his smartphone. Although an advanced attacker may substitute the true smartphone with a fake one to trigger the user's raise-to-wake gesture, this attack may be too advanced. Second, the anticipated acceleration ratio obtained in the training phase is closely tied to the phone-holding gesture, hand geometry, and wrist-wearing habit of the legitimate user. In addition, the acceleration ratio is recalculated based on the fresh data obtained in each raise-to-wake gesture for comparison with the anticipated one. It is also very difficult to perfectly emulate the victim's gesture which usually lasts less than 2 s. Therefore, the attacker should have little chance to induce qualifying acceleration data on the stolen smartphone.

For the WristTouch mode, the mimicry attack is infeasible unless the victim touches his wrist wearable against a fake smartphone substituted by the attacker. In this worst-case scenario, the wrist wearable transmits a random challenge (encrypted and authenticated) via the secure Bluetooth channel to the stolen smartphone. Since the attacker does not know the exact random challenge a priori, he can only use his own wrist wearable to transmit a randomly guessed challenge via the vibration channel to the smartphone in a synchronized fashion with the victim. Given an n-bit random challenge, the attacker can succeed with probability $1/2^n$. The smartphone can disable WristTouch after a few failed unlocking attempts, so n need not be very long for sufficient security. The attacker may obtain the correct random challenge through the fake smartphone and then replay it to the stolen smartphone. This strategy requires a higher cost on the attacker because the fake phone need be indeed functional. A simple but effective defense is to additionally verify the decoding latency in WristTouch, which is at least doubled by the attacker. Therefore, WristTouch is invulnerable to the mimicry attack.

Fake-message attack. Aiming at both WristRaise and WristTouch, the attacker impersonates the wrist watch to send fake messages to the smartphone via the Bluetooth channel, which matches the acceleration data he induces on the possessed smartphone. We rely on the secure Bluetooth channel for the smartphone to detect and ignore such fake messages.

Denial-of-service (DoS) attack. The attacker does not possess the victim's smartphone in this scenario. Instead, he may disturb legitimate unlocking operations by jamming the Bluetooth channel when observing the victim's unlocking attempts. Such jamming attacks apply to all authentication protocols exploring wireless channels such as WearUnlock [6]. The attacker also needs great effort to continuously launch the jamming attack.

Another DoS attack is that the attacker keeps sending fake activation signals to the wrist wearable which may quickly drain its battery by verifying and then ignoring such fake signals. This attack can be easily mitigated by enforcing rate limiting in both WristRaise and WristTouch.

VII. Performance Evaluation

In this section, we report the evaluation of WristRaise and WristTouch through prototype implementations on a Google Nexus 7 smartphone and a Huawei Watch 2.

A. Evaluation of WristRaise

1) Experiment setup: We invited 10 volunteers to evaluate WristRaise, which are all graduate students on campus. Each volunteer was asked to hold the smartphone in his/her own way with the smartwatch on the same arm. Then each volunteer performed his/her own raise-to-wake gesture 30 times, leading to 300 samples of acceleration ratios in total. During the process, the volunteers were asked to stand or sit naturally and still, and try not to have movements.

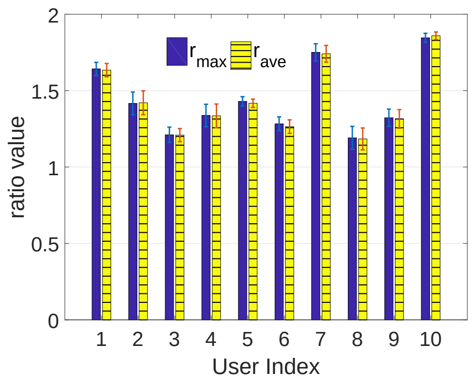


Fig. 5: Acceleration ratios of different users.

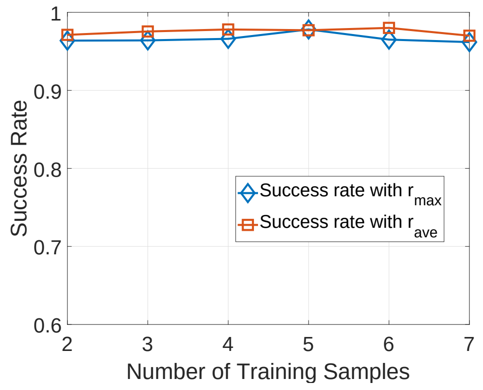


Fig. 6: Success rates of legitimate users.

2) Results: Fig. 5 depicts the acceleration ratios of different users obtained in the training phase, where r_{ave} and $r_{\max}$ denote the average acceleration ratios obtained from all the available data or only from the interval that contains the maximum number of data points (see Section IV-B). It is interesting to see that the volunteers have diverse acceleration ratios due to different arm structures, watch-wearing habits, and phone-holding gestures.

Fig. 6 shows the success rates of legitimate users (i.e., true-positive rates), where the distance threshold $\theta = 0.01$. We

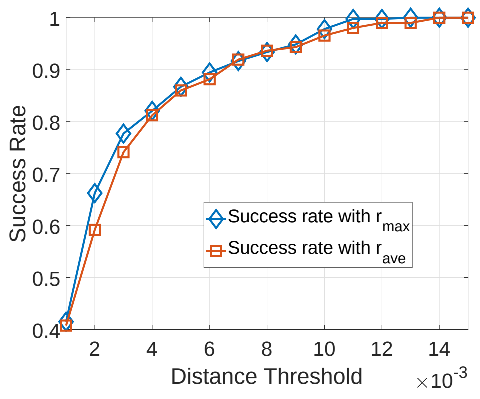


Fig. 7: Impact of the distance threshold θ.

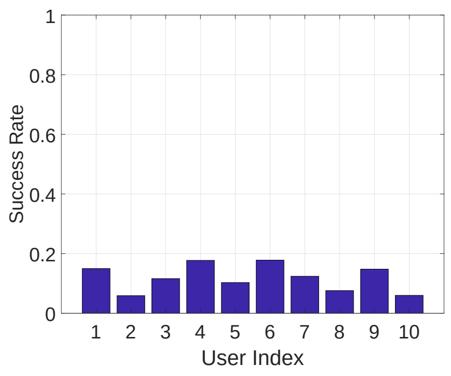


Fig. 8: Success rates of the mimicry attack.

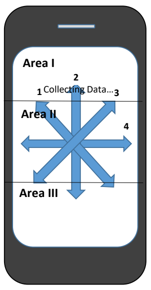


Fig. 9: Illustration of touch positions and orientations.

used repeated random sub-sampling to generate this result. In particular, we varied the number t of training samples to emulate different training time for each volunteer. For each $t \in [2, 7]$ and each volunteer, we randomly selected t samples from the 30 available ones for training and used the remaining $30 - t$ samples for testing. This process was repeated 10 times for each volunteer and each t. The average was reported in Fig. 6 and is over 95% in most cases.

Fig. 7 demonstrates the impact of the distance threshold θ. As expected, the larger θ (corresponding to more relaxed security requirements), the higher the success rate, and vice versa. θ can be trained and dynamically updated in practice.

Fig.8 shows the success rates of the mimicry attack (i.e., false-positive rates) on each volunteer, which are reasonably low under the distance threshold $\theta = 0.01$. In this experiment, we used the 270 samples of all other volunteers as the attack samples against each volunteer. We had no surprise to find that the volunteers with more unique arm structures, hand geometries, watch-wearing habits, and phone-holding gestures are much less vulnerable to the mimicry attacks.

B. Evaluation of WristTouch

1) Experiment setup: This experiment involved one volunteer. Unless otherwise stated, the smartphone laid flat on a table in a quiet lab; the volunteer touched the worn smartwatch against Area II on the smartphone with his fingers following Orientation 4 in Fig. 9. We also conducted experiments in other contexts. We wrote a pair of apps and installed them on the smartphone and smartwatch, respectively. A 20-bit frame was used in our experiments, including a beginning flag "11110", a random challenge s of 10 bits, and an ending flag "11110". Since WristTouch is highly secure as analyzed in Section VI, we focused on evaluating WristTouch in the legitimate context by adopting the success rate (or TPR) and authentication latency as the main performance metrics.

2) Results: We first evaluated the impact of bit vibration time T ranging from $50\,\text{ms}$ to $300\,\text{ms}$. To avoid doing bit stuffing, we chose decimal numbers 547, 593, 613, 657, 681, 785, 787, 789, 795, 835, 851, and 859 to emulate random challenges, as each of them has a 10-bit binary representation without three consecutive bit-1s. Table I shows the average result. As we can see, it is hard to achieve successful authentication if $T < 150\,\text{ms}$, which is mainly due to the hardware constraints of the vibrator and accelerometer on the smartwatch and smartphone, respectively. Although larger $T \geq 200\,\text{ms}$ ensures zero bit error rate via the vibration channel and thus a 100% success rate, the authentication latency may be too large to bear if T is too large. The optimal choice in our experiment is $T = 200\,\text{ms}$, which can serve as a reference parameter for other device models.

Then we checked the impact of environments and also body motion. In this experiment, the bit vibration time T was fixed as $200\,\text{ms}$. We did the experiments in five common environments: a quiet room, a noisy room, outdoors, a still car with the engine on, and a moving car. We also considered three cases during the vibration process in each environment: (1) the user is still without relative movement between the two devices; (2) the user has casual body movement without relative device movement; and (3) the user is still with slight relative device movement. Fig. 10 shows the average number of transmissions needed to achieve successful authentication for each combination of settings, where each point is the average of 9 to 15 trials. It is clear that WristTouch is highly robust and efficient in various contexts.

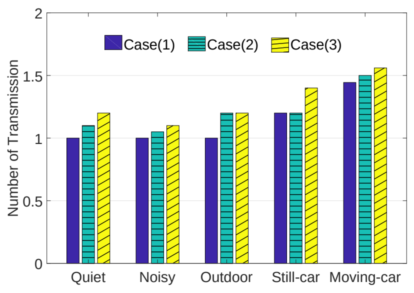


Fig. 10: Impact of environments and body motion.

Finally, we evaluated the impact of touch positions and orientations, which are plotted in Fig. 9. In particular, we divided the phone front into three areas and identified four possible relative wearable-phone orientations (represented by four double-arrow lines). Fig. 11 shows the success rate for different combinations of touch positions and orientations, where the real touch position and orientation were approximated to the closest in Fig. 9. Each result in Fig. 11 represents the average of

TABLE I: Impact of bit vibration time.

Vibration time	50 ms	100 ms	150 ms	200 ms	250 ms	300 ms
Success rate	25.00%	41.67%	91.67%	100.00%	100.00%	100.00%
Authentication latency	1.271 s	2.275 s	3.276 s	4.279 s	5.289 s	6.294 s

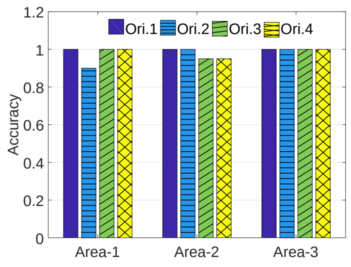


Fig. 11: Impact of touch positions and orientations

15 trials. It is clear that we can achieve a very high success rate in all scenarios. We also put the phone face down on the table and let the wrist wearable touch Area II on the device back along Orientation 4. The phone successfully got unlocked in all trials. These results indicate the high usability of WristTouch in the sense that the user can touch his smartphone with the wrist wearable in an arbitrary way.

VIII. Additional Related Work

Most related to WristUnlock are ShakeUnlock [5] and WearUnlock [6], which have been discussed and compared in Section I. In this section, we outline some other related work, which is by no means an exhaustive list.

There are some token-based smartphone authentication schemes, e.g., [9]–[12]. Such hardware tokens have to be specially built and are not available on the market. In contrast, WearUnlock explores COTS wrist wearables that have penetrated into everyday life.

Recent years have also seen a surge of interest (e.g., [13]–[16] in authenticating smartphone users based on behavioral biometrics. WearUnlock is orthogonal to and complimentary to this line of research.

In addition, the vibrator-accelerometer channel has been explored to communicate between mobile devices (e.g., [17], [18]. Our WristTouch implements and apply the vibration channel in a totally different way.

One may consider using RF distance bounding [19] to verify the proximity between the smartphone and wrist wearable. But distance bounding has a high hardware requirement that is not available on most COTS smartphones and wrist wearable.

IX. Conclusion

In this paper, we presented the design, analysis, and evaluation of WristUnlock, a secure and usable technique to unlock a smartphone with a wrist wearable on the same user. WristUnlock explores both the physical proximity and secure Bluetooth connection between the smartphone and wrist wearable. We thoroughly analyze the security of WristUnlock and confirmed its high efficacy through detailed experiments.

Acknowledgment

We would like to thank the anonymous reviewers for their insightful comments that help improve the quality of this paper. This work was supported in part by US National Science Foundation under grants CNS-1514381, CNS-1619251, and CNS-1824355.

References

[1] https://www.businessinsider.com/the-average-iphone-is-unlocked-80-times-per-day-2016-4.
[2] https://www.kaspersky.com/about/press-releases/2018_mobile-pickpockets.
[3] https://www.statista.com/statistics/538237/global-smartwatch-unit-sales/.
[4] L. Francis, G. HanckeKeith, and M. Markantonakis, "Practical NFC peer-to-peer relay attack using mobile phones," in *RFIDSec*, Istanbul, Turkey, Jun. 2010.
[5] R. Findling, M. Muaaz, D. Hintze, and R. Mayrhofer, "ShakeUnlock: Securely transfer authentication states between mobile devices," *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 1163–1175, 2017.
[6] S. Yi, Z. Qin, N. Carter, and Q. Li, "Wearlock: Unlocking your phone via acoustics using smartwatch," in *IEEE ICDCS*, Atlanta, GA, June 2017.
[7] T. Li, Y. Chen, J. Sun, X. Jin, and Y. Zhang, "iLock: Immediate and automatic locking of mobile devices against data theft," in *ACM CCS*, Vienna, Austria, October 2016.
[8] [Online]. Available: https://bit.ly/2SB1wJd
[9] V. Roth, P. Schmidt, and B. Guldenring, "The IR ring: Authenticating users' touches on a a multi-touch display," in *UIST*, New York City, NY, October 2010, pp. 259–262.
[10] H. Bojinov and D. Boneh, "Mobile token-based authentication on a budget," in *ACM HotMobile*, Phoenix, AZ, March 2011.
[11] T. Vu, A. Baid, S. Gao, M. Gruteser, R. Howard, J. Lindqvist, P. Spasojevic, and J. Walling, "Distinguishing users with capacitive touch communication," in *ACM MobiCom*, Istanbul, Turkey, August 2012, pp. 197–208.
[12] Y. Liu, M. Yang, Z. Ling, and J. Luo, "Implicit Authentication for Mobile Device based on 3D Magnetic Finger Motion Pattern," in *IEEE CSCWD*, Wellington, New Zealand, April 2017.
[13] J. Sun, X. Chen, J. Zhang, Y. Zhang, and J. Zhang, "TouchIn: Sightless two-factor authentication on multi-touch mobile devices," in *IEEE CNS*, San Francisco, CA, October 2014.
[14] Y. Chen, J. Sun, R. Zhang, and Y. Zhang, "Your song your way: Rhythm-Based Two-Factor authentication for Multi-Touch mobile devices," in *IEEE INFOCOM*, Hong Kong, China, April 2015.
[15] Y. Chen, J. Sun, X. Jin, T. Li, R. Zhang, and Y. Zhang, "Your face your heart: Secure mobile face authentication with photoplethysmograms," in *IEEE INFOCOM*, Atlanta, GA, April 2017.
[16] A. Lewis, Y. Li, and M. Xie, "Real time motion-based authentication for smartwatch," in *IEEE CNS*, Philadelphia, PA, Oct. 2016.
[17] I. Hwang, J. Cho, and S. Oh, "Vibecomm: Radio-free wireless communication for smart devices using vibration," *Sensors*, vol. 14, no. 11, pp. 21 151–21 173, 2014.
[18] A. Studer, T. Passaro, and L. Bauer, "Don't bump, shake on it: The exploitation of a popular accelerometer-based smart phone exchange and its secure replacement," in *ACSAC*, Orlando, Florida, Dec. 2011.
[19] K. B. Rasmussen and S. Čapkun, "Realization of rf distance bounding," in *USENIX Security*, Washington, DC, 2010, pp. 25–25.