CoSafe: Securing Mobile Devices through Mutual Mobility Consistency Verification

Shan Chang, *Member, IEEE*, Hang Chen, Hongzi Zhu, *Member, IEEE*, Xinggang Hu, and Di Cao

Abstract—As mobile devices play increasingly important roles in our daily lives, it is of great significance to protect personal mobile devices from being lost. Noticing the trend that one person normally carries more than one mobile device, we propose an innovative scheme, called *CoSafe*, to detect device loss by verifying the motion consistency between a pair of devices. The rationale is that the vibrations perceived on devices carried by the same person should be tightly coupled whereas a lost device would show distinct mobility characteristics from others. Specifically, CoSafe compares the mobility consistency between a pair of devices on three levels, where coarse features (i.e., the mobility state and motion periodicity) are first compared to give fast response and more complex comparison on subtle feature (i.e., the relative phase) is conducted only when needed. In this way, CoSafe can instantly respond and introduce very low computation and communication costs. We implement CoSafe on a Commercial-Off-The-Shelf Android smartphone and a smartwatch, and conduct both trace-driven simulations and real-world experiments to evaluate the performance of CoSafe. The results show that CoSafe achieves a mean false negative ratio and false positive ratio of 1.46 and 3.12 percent, respectively, even under sophisticated stealing attacks.

Index Terms—Device loss detection, mobile device, motion consistency, SVM, cross wavelet analysis

1 Introduction

While smartphones are now firmly established as an integral part of people's lives, the need for always-on connectivity and accessibility has led to the emergence of more wearable devices such as smartwatches and fitness trackers. According to a new study by Cisco, by 2020, the average number of devices per person is edging closer to 4 [1]. However, these devices are vulnerable to device loss. For example, 70 million smartphones are lost each year, with only 7 percent recovered [2]. In the U.S., according to the IDG Research's report [3], 44 percent phone loss is because the owner leave the phone behind in a public setting and another 11 percent is stolen off the victim's body. Beyond the price of replacement, private information leakage resulting from device loss or theft has been the number one information security risk for mobile device users. The situation becomes even severe when a person carries more mobile devices. To detect device loss or theft on site, therefore, is of the greatest concern to users.

In the literature and industry, most existing loss detection solutions [4], [5] are based on the stability of a wireless link (e.g., Bluetooth) established between a pair of mobile devices. A device is considered lost when the signal-to-noise-ratio (SNR) of the link is lower than a threshold. SNR-based schemes have large false positive as well as false negative errors due to the unpredictable fading behavior of wireless channels. In contrast, 'Kill switch' based solutions (e.g., the Apple's Activation Lock feature in iOS 7 and Samsung's Absolute LoJack) are designed to make lost devices essentially useless. Though these *reactive remedies* might mitigate the leakage of private data stored in lost devices, they cannot prevent a mobile device from being lost in the first place. One recent scheme, called iGuard [6], tries to detect stealing actions by discriminating the 'take-out' motions performed by a walking carrier (owner) and a pickpocket. iGuard relies on two strong assumptions: first, a walking carrier of a smartphone will slow down before taking out the smartphone, on the contrary, a pickpocket will speed up after stealing a smartphone. This assumption makes iGuard vulnerable to imitation attacks, resulting high false negative rate. Second, 'take-out' motions performed by the same person are similar. The assumption makes iGuard suffer from high false positive rate in certain scenarios, for example, the carrier takes out a smartphone from shoulder bag rather than pocket, or the carrier takes out the phone while he is climbing stairs.

Our Approach. In this paper, we propose an online device loss detection scheme, called *CoSafe*, which can be implemented as software on mobile devices. CoSafe is inspired by the following two key insights. First, *an individual usually carries several types of mobile devices like smartphone, smartwatch, bracelet, and earphones*. Second, *the main reason for a device loss is that the device undergoes a different mobility status from that of its owner and all other carry-on devices, which eventually makes the device out of reach of its owner*. For instance, as illustrated in Fig. 1, after the phone of a walking victim is stolen by a pickpocket, the phone experiences a different gait pattern from the smartwatch worn on the victim's wrist. In another distinct situation, where a person may accidentally leave his/her

• S. Chang, H. Chen, X. Hu, and D. Cao are with the School of Computer Science & Technology, Donghua University, Shanghai 201620, China. E-mail: changshan@dhu.edu.cn, {chenhang, hxg, caodi}@mail.dhu.edu.cn.
• H. Zhu is with the Department of Computer Science and Technology, Shanghai Jiao Tong University, Shanghai 200000, China. E-mail: hongzi@cs.sjtu.edu.cn.

Manuscript received 6 Apr. 2019; revised 9 Jan. 2020; accepted 10 Feb. 2020. Date of publication 17 Feb. 2020; date of current version 2 Apr. 2021. (Corresponding Author: Hongzi Zhu) Digital Object Identifier no. 10.1109/TMC.2020.2974222

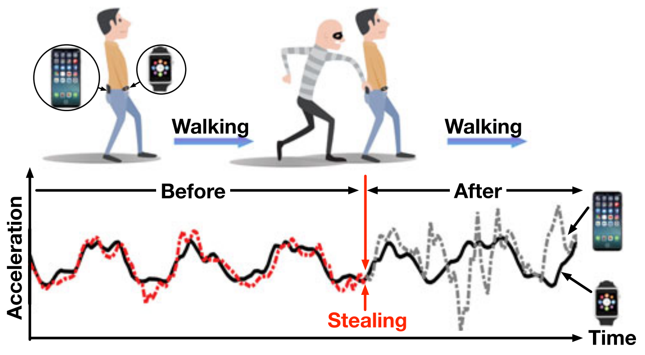


Fig. 1. An illustration of a pickpocket stealing the smartphone of a victim.

phone behind on a coffee table, as long as he/she starts to walk, the phone and other wearable devices of the person realize different mobility states.

Thus, the core idea of CoSafe is to make device loss decisions based on *mobility consistency comparison* between devices, which can deal with both unintentional drop and deliberate theft of a device. More specifically, each mobile device detects human mobility (e.g., being still, walking or running) based on its accelerometer readings. Once a human mobility change (e.g., from being still to walking, or walking with different gait frequencies) is identified on one of the devices, a pair of devices exchange their sensor readings through low-power wireless communication. Then, the mobility consistency is compared between this pair of device on three levels with different computational costs, i.e., coarse features such as the perceived mobility state and motion periodicity are compared first, and more complex comparison on the relative phase between vibration signals is conducted only when needed. If two devices are believed to be inconsistent in any level, a device loss is detected and triggers alarms on both devices.

Challenges. There are two main challenges to realize CoSafe. First, it is difficult to tell whether two devices are located on the same person. Raw sensor readings collected from two devices on the same person may appear very different, due to the diversities of device position (e.g., one device worn on the left wrist and the other put in a bag on the right shoulder) and unstable human mobility (e.g., walking speed varies over time). Second, it is very hard to judge whether two devices are located on different persons under an imitation attack, in which a pickpocket mimics the walking behavior of a victim.

To tackle the first challenge, we conduct an empirical study by collecting and analyzing a real-world dataset, and find that although the shapes of the vibration signals perceived on each device look quite different but they are tightly coupled. In CoSafe, we do not directly compare how similar two motion signals but compare *how they are coupled as per a mobility change*. More specifically, we calculate the linear correlation between two sequences of stride periods perceived on each device.

To deal with the second challenge, we examine the mobility consistence at a finer granularity. Specifically, the relative phase changes of two motion signals around the gait frequency are analyzed by the Cross Wavelet Transform (XWT) over time. Effective relative phase features are extracted to train a one-class Support Vector Machine (SVM) for consistency verification.

We implement CoSafe as an app on two Android devices, i.e., a smartphone and a smartwatch. To start, a user needs to carry both devices and walk in various speeds for three minutes for training a SVM classifier. Afterwards, CoSafe can begin to protect both devices. CoSafe is a light-weight protocol, having the minimal requirement on hardware. We conduct extensive trace-driven simulations and real-world experiments. The results show that our CoSafe achieves an overall average false negative ratio (FNR) and false positive ratio (FPR) of 1.46 and 3.12 percent, respectively, even under the presence of sophisticated imitation attacks.

2 Related Work

Recently, researchers realize that the fusion of multiple mobile devices can improve the accuracy of activity recognition. T. Vilarinho *et al.* design a fall detection system based on an off-the-shelf smartwatch and smartphone [7]. M. Shoaib *et al.* study the fusion of a wrist-worn device and a smartphone for daily physical activity recognition, including smoking, eating, typing, drinking, walking, jogging etc. [8]. CrossMotion [9] matches the acceleration of a mobile device to similar acceleration observed in the infrared and depth images, in oder to locating and tracking a mobile device and its user in video reliably (even when the user is not in direct view of the camera). F. Nurwanto *et al.* propose a light sport exercise activity detection system. By placing a smartphone and a smartwatch in the upper arm and on the wrist of a user, respectively, the system is able to recognize the movements made by the user and calculate the number movements [10]. A. Muaremi *et al.* present a solution for assessing the stress experience of people, using features derived from smartphones and wearable chest belts [11]. However, these works focus on achieving fine-grained activity recognition by introducing more sensors, and cannot be used directly for loss detection.

Existing online loss detection proposals can be further classified into the following categories.

Device-to-Device Secure Pairing Based. To establish secure channel, movements of a user perceived by multiple devices can be used as a secret shared between involved devices for mutual authentication. Existing solutions are either shaking or walking based. R. Mayrhofer *et al.* [12] and D. Bichler *et al.* [13] share the similar idea of shaking two devices together to pair them. Similarly, K. Chen *et al.* pair two devices through waving a hand from one device towards another, for the purpose of sharing files between them [14]. These methods require the user to participate the establishment of secure channel. These shaking based solutions require a user to actively participate in the establishment of a secure channel. J. Lester *et al.* determine if two devices are carried by the same people by exploring the coherence characteristics of the acceleration data of the devices when the wearer is walking [15]. This work suffers from long pairing time (8 secs), and two devices should be placed in the same location on the body in order to achieve high accuracy.

Continuous Authentication Based. Continuous authentication (also mentioned as active authentication) aims to check the user unobtrusively and continuously throughout the use of the devices. These proposals could be modified to detect device loss, for example, smartphone-based gait authentication. M. Muhammad *et al.* try to learn the unique features of individual's limb-movements from collected samples, and then use them for continuous identification [16]. However, due to the fact that those movement features are dramatically affected by sensor placement, inter-day performance, clothes, and shoes, existing studies put forward strict requirements that the user should wear the same shoes and clothes, and place the phone on the waist or inside the same pocket of sufficiently tight trousers. A. Sarkisyan *et al.* demonstrate a method for inferring smartphone PINs through the analysis of smartwatch motion sensors [17]. The method also requires both devices on the same hand.

Human behavior recognition based: Inertial sensors in mobile devices have been used to identity particular human gestures in certain scenarios. The most relevant work to CoSafe is an *anti-pickpocket system*, iGuard [6]. iGuard claims that phone owners and pickpockets have consistent but distinguishable order of motions when they taking out a smartphone from the pocket, and thus those features captured by smartphones can be used to detect abnormal 'take-out' motions. However, iGuard relies on some strong assumptions (the take-out and walking motions of certain user are similar, and the stealing behaviors of different pickpockets show the same motion sequences) that may be invalid in practice, which limits the accuracy of it.

Wireless Link Strength Based. Bluetooth-based anti-loss tags, for example Blue Watchdog [4] and Vnfire [5], can be attached to small objects to protect such objects from being lost. These anti-loss devices are generally fobs kept in a pocket or bag, and the owner will be informed when devices move out of a secure distance range. The main idea is using a host device such as a smartphone to check the RSSI of signals transmitted (or SNR of the link established) between the host and tag. Loss decisions are made based on a low SNR or RSSI threshold. However, the unpredictable fading behavior of wireless channels makes the performance unstable. Moreover, dedicated hardware is required which imposes additional cost and difficulties for users like the elderly or the children.

Comparing with existing solutions, CoSafe has advantages that it does not make any restriction on the placement of devices or usage scenarios and can deal with both unintentional losses and deliberate thefts. We compare CoSafe with schemes proposed by J. Lester *et al.* [15] (device-to-device secure pairing based), M. Muhammad *et al.* [16] (continuous Authentication-based), and iGuard [6] (human behavior recognition based) in Section 7. Futhermore, we conduct a set of experiments in which we place two devices several metres (from 0.1m to 5m at a step of 0.1m) away from each other, and measure the RSSIs of signals transmitted between them, the CDFs of RSSI are depicted in Fig. 2. It can be seen that given certain RSSI value, it is hard, if not impossible, to decide the corresponding device distance. The estimation error would be too large to distinguish whether two devices are carried by the same person or not. Thus, we didn't compare CoSafe with those wireless link strength based schemes.

3 Models and Design Goals

3.1 System and Threat Models

We require a user of our system to carry at least two mobile devices. These mobile devices can be placed or worn at any

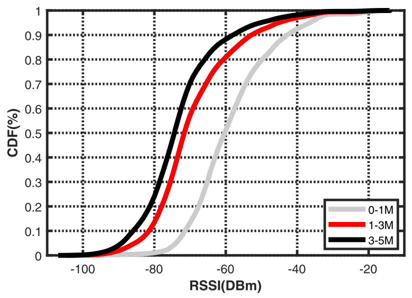


Fig. 2. The CDFs of RSSI of signals transmitted between two devices that the distance between them is 0-1m, 1-3m, and 3-5m, respectively.

body positions of the user. Such devices are equipped with Inertial Measurement Unit (IMU) sensors and can exchange data through secure wireless communication, such as Bluetooth version 4.0 (called Bluetooth Smart [18]) which has a low energy protocol stack. In addition, devices may have different computational power but they should be able to conduct simple signal processing.

We consider our system to confront the following threats:

- *Stealing Attack:* Mobile devices are being stolen off the victim's body. A pickpocket has knowledge of the system and take countermeasures against the system.
- *Inadvertent Loss:* Mobile devices are left behind in a public setting by their owners due to their casualness.
- *DoS Attack:* Mobile devices could be disabled by blocking wireless communication or turning off the power.

3.2 Design Goals

A practical device loss detection scheme should meet the following four goals:

- *High Accuracy:* The scheme should achieve a very low false negative ratio even under stealing attacks to securely protect devices. In addition, it should also achieve a low false positive ratio to minimize the disturbance to users.
- *Short Response Time:* Decisions should be made on site before a device has been stolen and powered off.
- *High Reliability:* The scheme should function properly in various usage scenarios, such as by different transportation modes, with arbitrary device placement, and in various human mobility states. In addition, the scheme should react to DoS attacks on wireless communication.
- *Low Power Consumption:* The scheme should be lightweighted so that it can be implemented on battery-powered mobile devices.

4 Overview of CoSafe

Without loss of generality, we consider a pair of mobile devices. In particular, the device (e.g., a smartphone or a tablet) with more computation power acts as *the master* while the other device (e.g., a smart watch or a bracelet) acts as *the slave*. If two devices have the same computation power, they can be master and slave in turn. Fig. 3 depicts the architecture of CoSafe. In essence, CoSafe is a protocol running on a pair of

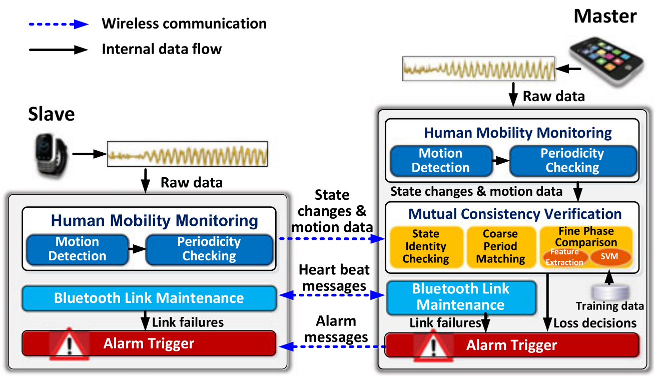


Fig. 3. Architecture of the CoSafe protocol.

master and slave devices, which integrates four components as follows:

Human Mobility Monitoring (H2M). The H2M component consists of two blocks, i.e., motion detection (MD) and periodicity checking (PC). The motion detection block keeps examining whether there is a motion associated with a device by analyzing the raw acceleration data, and if yes, the periodicity checking block further determines whether such motion is related to human mobility (e.g., walking or running). Both the slave and the master need to run H2M to identify their own mobility state. Once a mobility state change is identified on either device, the information of the current mobility state and the latest motion signals obtained on both devices are sent to the Mutual Consistency Verification (MCV) at the master for comparison.

Mutual Consistency Verification (MCV). The MCV component consists of three blocks, i.e., State Identity Checking (SIC), Coarse Period Matching (CPM), and Fine Phase Comparison (FPC). Specifically, the SIC block first examines whether both devices are associated with the same mobility state. Distinct states (e.g., one device in a still state versus the other in a walking state) indicates an immediate device loss. If both devices are in the same mobility state, the CPM block checks the similarity of the stride periods perceived on each device. An obvious period difference leads to a device loss decision. Finally, if the stride periods on each device are similar, the FPC block examines phase features in the frequency domain with a pre-trained one-class SVM classifier. Inconsistent relative phases over time observed in selected frequencies cause a device loss decision. If there is no loss decision made in MCV, both devices are considered to be safe; otherwise, an alarm is triggered.

Alarm Trigger (AT). The AT component sits on both the master and the slave. Once a device loss decision or a link failure is detected on either device, the corresponding AT component triggers alarms through sounds and LED Flashes immediately. In addition, it also sends an alarm message to its counterpart device to trigger alarms on that device too. Alarms can only be manually cancelled by the legitimate user through standard identification schemes.

Bluetooth Link Maintenance (BLM). Considering that Bluetooth is prevalent over mobile devices and has ultra-low power consumption, the BLM component maintains a Bluetooth link between the master and the slave for the following purposes. First, the link is used to transmit the information of the current mobility state and latest motion

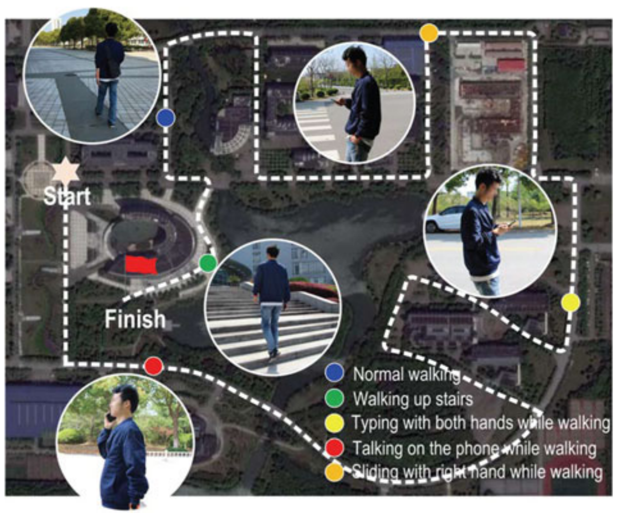

Start
Finish
Normal walking
Walking up stairs
Typing with both hands while walking
Talking on the phone while walking
Sliding with right hand while walking

Fig. 4. Illustration of *Trace A* collection campaign.

data from the salve to the master. Second, heart beat messages are exchanged between devices to defend against DoS attacks which block wireless communication (e.g., through jamming) or turn off the power of devices. If a valid heart beat message is not received within a given short period of time, a link failure alarm is raised. In addition, data synchronization between devices is also achieved through heart beat messages.

5 Human Mobility Monitoring

Detecting whether a device is experiencing human mobility is key to the CoSafe protocol. The H2M component analyzes raw acceleration data to identify human mobility.

5.1 Empirical Data Collection

To study the characteristics of motion signals related to human mobility, we conduct a data collection campaign on campus. Three types of COTS Android smartphones, i.e., Huawei Honor 7, Huawei Honor 5x, and Google Nexus 4, and a MOTO 360 Android Wear smartwatch are used to record the inertial sensor data at a sampling frequency of 50 Hz. In specific, we recruit ten volunteers, four females and six males, aging from 18 to 35, to perform field experiments. Volunteers are asked to choose a location (including left and right coat pockets, left and right front trousers pockets, a shoulder bag and a backpack) to place an experiment phone and wear the smartwatch on the left wrist. Then, they are asked to walk along a 3.8 km long trail (as illustrated in Fig. 4).

During a walk, they are required to operate the phone as usual, including taking out the phone, holding the phone, making a phone call, typing, and putting the phone back. We tape the process for reference. Motion data are collected over two periods in the year of 2018, i.e., one week from January 25 to January 31 and over two weeks from February 26 to March 11. Thus, we get a dataset, denoted as *Trace A*, containing 36 traces with each lasting for 55 to 80 minutes.

5.2 Motion Detection

Given raw acceleration reading on 3 axes, i.e., $\vec{a} = (a_x, a_y, a_z)$, because the posture of a device may continuously change, we

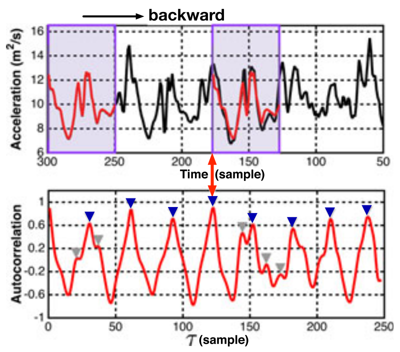


Fig. 5. Illustration of backward autocorrelation calculation.

study signal $a = \|\vec{a}\| = \sqrt{(a_x)^2 + (a_y)^2 + (a_z)^2}$. In order to determine the starting and ending points of each motion, we examine the standard deviation of a within a short sliding window of w_1 samples. We empirically take the length of the sliding window of 4 samples (e.g., 80 ms with the sampling rate of 50 Hz). We consider that a motion starts when the standard deviation exceeds a threshold α and stops when the standard deviation goes below the threshold for w_1 samples. An empirical threshold $\alpha = 1.1$ is used in this work.

5.3 Periodicity Checking

When a motion is detected, the acceleration signal a is further examined for periodicity through *backward autocorrelation*. In specific, given a time series of n samples of a, i.e., $\{a_1, a_2, \ldots, a_n\}$, denote the latest w_2 samples $\{a_{n-w_2+1}, \ldots, a_n\}$ as X_n and the w_2 samples $\{a_{n-w_2+1-\tau}, \ldots, a_{n-\tau}\}$ that are τ samples earlier than X_n as $X_{n-\tau}$. For a positive integer $\tau \in [1, w_3]$, the backward autocorrelation is defined as

$$R(\tau) = \frac{1}{w_2 \sigma_n \sigma_{n-\tau}} \sum_{t=1}^{w_2} (X_n - \mu_n)(X_{n-\tau} - \mu_{n-\tau}), \tag{1}$$

where μ_n and σ_n are the mean and the standard deviation of X_n, respectively, and $\mu_{n-\tau}$ and $\sigma_{n-\tau}$ are the mean and the standard deviation of $X_{n-\tau}$, respectively. We refer to X_n as the *templet window* and the range of τ, i.e., $[1, w_3]$, as the *window of interest* (WoI).

With backward autocorrelation, new repetitive patterns embedded in the acceleration signal can be identified where the autocorrelation value reaches maxima. Fig. 5 illustrates an example of the backward autocorrelation calculation conducted on the acceleration signal of a device while its user is walking. As depicted in the upper subplot of Fig. 5, a templet window (denoted as a colored box) of 50 samples slides backward within a WoI of 250 samples to calculate backward autocorrelation. The corresponding autocorrelation values are plotted in the lower subplot of Fig. 5. It can be seen that the periodicity of walking strides can be clearly identified by those blue peaks. Nevertheless, as denoted by light grey marks, there also exist other noisy autocorrelation peaks. We filter out noisy peaks according to the following two rules: 1) As small peaks indicate weak correlation with the templet window, peaks that are smaller than a threshold β will be removed; 2) as we expect to identify human mobility, where stride periods normally exceed 0.2s, peaks should be separated by at least δ samples. Based on the analysis on *Trace A*, we choose an empirical $\beta = 0.4$ and $\delta = 18$.

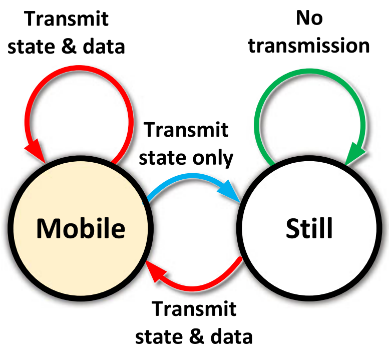


Fig. 6. Mobility state transitions and data transmission to the master.

After that, if there is a peak found within the WoI, it is believed that the device is experiencing a human mobility and the mobility state of the device is marked as mobile; otherwise, the mobility state of the devices is marked as still. The corresponding values of τ of all found peaks, denoted as $P = (\tau_1, \tau_2, \ldots, \tau_k)$, correspond to all identified periods.

On both devices, the backward autocorrelation is continuously conducted every time a new templet window is collected. If the mobility state of a device changes or if the device is mobile, mutual consistency verification should be conducted on the master device. Fig. 6 illustrates the state transitions and the corresponding data to be sent to the master. More specifically, if the state stays in the still state, no transmission happens, since there is no new information about the state of the slave should be known by the master. If the state stays in the mobile state or the state changes from the still state to the mobile state, both the state change information and the motion data within the current WoI are transmitted. It is because that the period information of the slave might need to be compared with that of the master. In contrast, if the state changes from the mobile state to the still state, only state change information is transmitted, i.e., reminding the master that the slave turns into still state. Such design avoids unnecessary state transmissions and comparisons, thus the transmission cost of the slave, as well as the computational cost of the master can be reduced.

6 Mutual Consistency Verification

MCV is the core component in the CoSafe protocol, which makes loss decisions at the master. According to the difficulty of verifying the mobility consistency between both devices, MCV follows a three-level strategy.

6.1 State Identity Checking

On Level I, the mobility states of both devices are compared by the SIC block. The intuition is straight—if both devices colocate on the user, they should have the same mobility state. State inconsistency is easy to detect at almost no computational cost. Specifically, once the MIC block receives a mobility state change message either from the master itself or from the slave, it waits for a short period of time (e.g., due to wireless communication delay) to collect the state change message from the other device (if any), and then compares the mobility states of both devices. If states are inconsistent, the MIC block directly reports a device loss decision; otherwise, comparison goes to Level II.

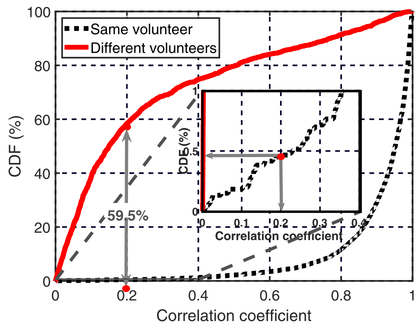


Fig. 7. The CDFs of ρ between two devices.

6.2 Coarse Period Matching

On Level II, both devices are reported in the mobile state, the CPM block compares the similarity of periods identified on each device.

We have the observation that stride periods may slightly change even in a short period of time. Moreover, the same changes should be perceived on all carry-on devices. Therefore, the CPM block compares all identified periods within the same WoI on both devices by calculating the Pearson's correlation coefficient,

$$\rho = \frac{cov(P_m, P_s)}{\sigma_{P_m}\sigma_{P_s}}, \tag{2}$$

where P_m and P_s are the identified periods on the master and on the slave, respectively, cov is the covariance, and σ is the standard deviation.

With *Trace A*, we study the distributions of ρ between the two devices carried on the same volunteer and on two randomly selected volunteers, respectively. Fig. 7 plots the cumulative distribution function (CDF) of ρ. It can be seen that there exists big difference between the CDF of ρ when two devices are co-located on the same volunteer and located on different volunteers. For instance, less than 0.5 percent ρ values are smaller than 0.2 for the same volunteer whereas the ratio of that is 60 percent for different volunteers. Thus, the CPM block reports a device loss decision if the calculated ρ is smaller than a threshold. We take 0.2 as the threshold in this work.

6.3 Fine Phase Comparison

In the extreme situation, where both devices have the same mobility state and very similar identified stride periods, consistency comparison enters Level III. The FPC block examines subtle relative phase changes around stride frequencies over time between both devices.

6.3.1 Background of Cross Wavelet Transform

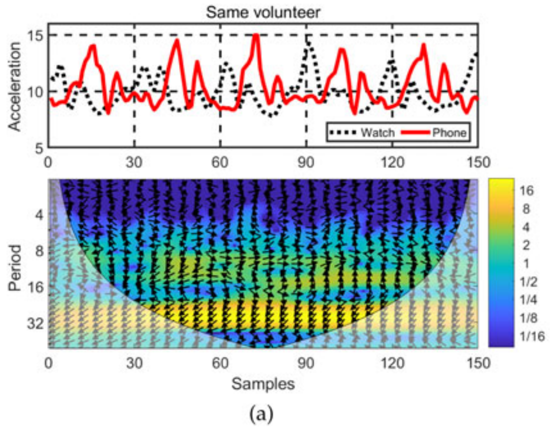
The Cross Wavelet Transform (XWT) can be used to check whether two time series contain common frequencies with high amplitude and a consistent phase relationship in the time frequency domain. Therefore, XWT is very powerful to analyze the interdependency between two time series. Given the acceleration signals collected at the master and the slave, i.e., a_m and a_s, respectively, the XWT of a_m and a_s is defined as $W^{a_m}W^{a_s}$, where W^{a_m} and W^{a_s} are the Continuous Wavelet Transform (CWT) on a_m and a_s, respectively.

The CWT of an acceleration signal $a[n], n = 1, \ldots, N$ with uniform time steps Δt is defined as the convolution of $a[n]$ with a scaled and translated version of a wavelet basis $\psi_0(\eta)$,

$$W_n^a(s) = \sqrt{\frac{\Delta t}{s}} \sum_{i=0}^{N-1} x_i \psi_0^* \left[\frac{(i-n)\Delta t}{s}\right], \tag{3}$$

where the $*$ indicates the complex conjugate and s is the wavelet scale. By varying the wavelet scale s and translating along the localized time index n, we can construct a matrix representing the amplitude of any features versus the scale and how this amplitude varies with time. We use the Morlet wavelet (with the dimensionless frequency $\varpi_0 = 6$) as wavelet basis $\psi_0(\eta)$, which provides a good balance between time and frequency localization [19], and conduct the fast convolution in Fourier space.

We calculate the XWTs of a_m and a_s of WoI when both devices co-locate at the same walking volunteer and when they locate at two different walking volunteers, respectively. Fig. 8a illustrates the acceleration signals a_m and a_s and the corresponding XWT when both devices locate on the same volunteers, where colors represent the cross common power and arrows represent the relative phase between the two signals over frequencies and time. It can be seen that frequencies around the stride frequency (e.g., about 29 samples per period) have high cross common power and stable relative phases over the whole WoI, even though a_m and a_s seem quite non-analogous in the time domain. In contrast, Fig. 8b illustrates the acceleration signals a_m and a_s and the corresponding XWT when devices are located on two different walking



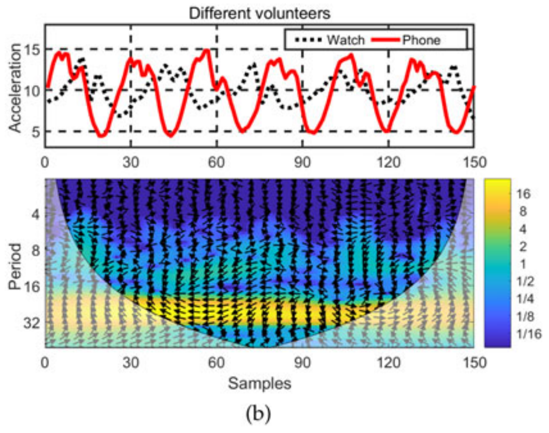
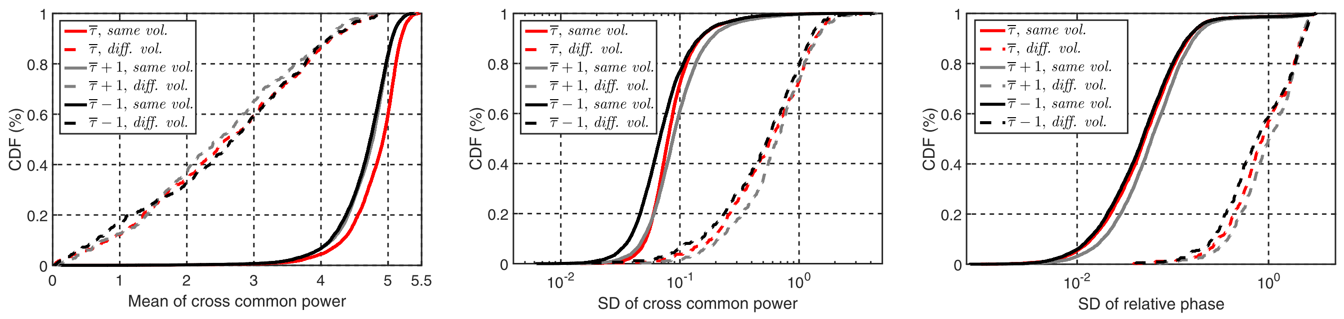


Fig. 8. The a_m and a_s and the corresponding XWT when devices locate on the same and different volunteers.



(a) CDFs of the mean of cross common power (b) CDFs of the SD of cross common power (c) CDFs of the SD of relative phase

Fig. 9. Efficacy of cross features.

volunteers. It can be seen that frequencies around the stride frequency also have high cross common power but the relative phases at such frequencies vary dramatically over time.

6.3.2 Extracting Relative Phase Features

It is effective to utilize XWT analysis for consistency comparison between the master and the slave devices. The computational cost of XWT, however, is prohibitive for mobile devices. Fortunately, as demonstrated in Fig. 8, XWT over all frequencies is not necessary.

Given the identified stride periods $P = (\tau_1, \tau_2, \ldots, \tau_k)$ from the master and from the slave, we calculate the overall average stride period, denoted as $\bar{\tau}$, and only calculate the XWT at $\in \{\bar{\tau}-1, \bar{\tau}, \bar{\tau}+1\}$ three frequencies, i.e., scale $s \in \{\bar{\tau}-1, \bar{\tau}, \bar{\tau}+1\}$, as a tradeoff between accuracy and computation cost. For each frequency, we calculate the mean and the standard deviation of the cross common power and the standard deviation of relative phase as features.

We analyze the nine features over *Trace A*. Fig. 9a, 9b and 9c plot the CDFs of the mean cross common power, the standard deviation of cross common power, and the standard deviation of relative phase at all three frequencies, respectively. It can be seen that there are obvious gap between the CDFs derived from devices located on the same volunteer and the CDFs derived from devices located on different volunteers. Therefore, these features are effective and can be utilized to train a classifier to differentiate the two cases.

6.3.3 Training Pairing Classifier

To train a classifier, the master device initiates a training procedure, where the user is required to carry the master and the slave devices and walk at different speeds for a few minutes. A one-class Support Vector Machine (SVM) classifier with the Radial Basis Function (RBF) kernel function is then trained.

We apply the open source implementation of one-class SVM in libSVM [20]. In the RBF kernel function of SVM, there are two parameters, i.e., the cost parameter c and the gamma parameter g, which impact the effect of training model. To obtain the appropriate parameters of c and g for one-class SVM, we adopt the scheme proposed in [21] and conduct a grid search over the same range of $[2^{-10}, 2^{10}]$ with cross validation on the training group. As all training samples are all from single-carrier situation, cross validation during the grid search only measures the true positive rate (TPR). As learned from our empirical experiments with *Trace A*, an SVM classifier even with a rough configuration of both parameters trained using cross features of acceleration magnitude sequences can easily reject a testing stealing attack performed by an attacker. In CoSafe, we choose the parameter values of c and g when the grid search finds the highest value of TPR as the best configuration to train SVM classifier.

6.3.4 Relative-Phase-Based Verification

With a pre-trained SVM classifier, the FPC block performs relative-phase-based verification. Specifically, given P_m, P_s, a_m, and a_s, a vector of nine features is calculated according to Section 6.3.2. Then the FPC verifies the vector with the pre-trained classifier and reports a device loss decision if the classifier labels the vector as negative.

7 Evaluation

7.1 Methodology

We ask ten volunteers involved in *Trace A* to perform a role-playing task, and collect a new data trace, denoted as **Trace B**. In the task, we ask one volunteer to play the role of victim who wears a smartwatch on the left wrist and places a smartphone on one of the following four positions, i.e., right front trousers pocket, right coat pocket, right shoulder bag (note that this left-right configuration, i.e., two devices are placed on the different side of body, is most challenging for CoSafe), and backpack. The other plays the role of pickpocket, who follows and tries to steal the smartphone with care (as illustrated in Fig. 10), in turn. The data collection is performed in both indoor and outdoor settings including seven normal terrains (as depicted in Fig. 11), as well as by two transport modes. In each setting, a victim walks at a normal pace for 30 seconds twice, in the first 15 seconds of

Fig. 10. Dataset *Trace B* collection example: a pickpocket follows a victim and tries to steal his phone.

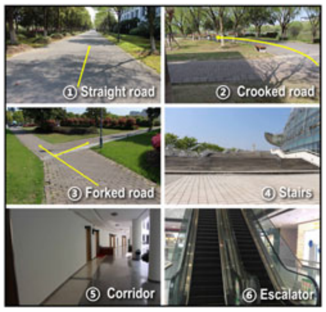


Fig. 11. Seven terrain environments considered in evaluation.

which the victim is asked to take out the smartphone by him/herself and to put it back into position once (for the purpose of verifying the robustness of CoSafe). In the second 15 seconds, a pickpocket steals the smartphone from the victim (for the purpose of verifying the effectiveness). In total, we collect 210 traces.

Moreover, we implement a prototype of CoSafe on a Google Nexus 4 Android smartphone equipped with a quad-core 1.5 GHz CPU and 2 GB memory (acting as the master device), and a MOTO 360 smartwatch with Qualcomm MSM8026 CPU and 512 MB memory (acting as the slave device), respectively.

We consider the following metrics:

- False Negative Rate (FNR): the fraction of the cases where CoSafe unable to recognize a theft event.
- False Positive Rate (FPR): the fraction of the cases where CoSafe mistakenly generates a false alarm when there is actually no theft event.

We first examine the effectiveness of system parameters through trace-driven simulations using *Trace B* and then conduct real-world attack experiments to evaluate the performance of CoSafe.

7.2 Effectiveness of WoI Size

Given a templet of size L, the size of WoI will be set as $k \cdot L$, $k \geq 2$. A large k may result in high accuracy but also high computational cost. For each volunteer, we train a one-class SVM classifier using dataset *Trace A* and use his/her corresponding data in *Trace B* as a victim for testing. We examine the effectiveness of k and vary k from 2 to 6.

Fig. 12 plots the average, maximum and minimum of FPR and FNR over all volunteers as a function of k. It can be seen that, on one hand, when k equals to 2, CoSafe gets a pretty low average FNR of 1.92 percent. As k increases to 3, the FNR decreases to 1.39 percent. When k further increases, the change of average FNR is not obvious. On the other hand, interestingly, FPR tends to rise with k increasing. The reason is that two devices may show more difference on the

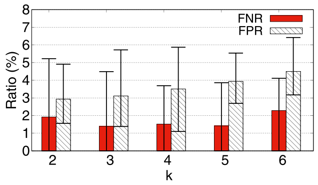


Fig. 12. Effectiveness of WoI size.

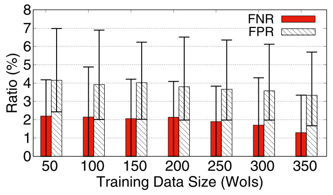


Fig. 13. Impact of training-data size.

variation of periods with a large WoI. Consequently, $k = 3$ can achieve a good balance on both FNR and FPR.

7.3 Effectiveness of Training Data Size

In this experiment, we study how much history data are sufficient to train an SVM classifier. We use a similar setting as above experiment except that k is set to 3. We vary the training data size from 50 to 350 pairs of motion segments at an interval of 50 pairs with each segment being $3L$ samples.

Fig. 13 shows the average, maximum and minimum of FPR and FNR over all volunteers as a function of the training data size. We have two observations. First, even a small training data size can achieve a very low average FNR, e.g., the mean FNR is 2.19 percent when the training-data size is 50. Second, the average FPR and FNR drop gradually with the increasing of training data size. Thus, CoSafe can use a very small initial training dataset, e.g., 50 pairs of motion segments (i.e., for a sampling rate of 50 Hz and a typical L of 50, the overall length of training sequence is $3 * 50 = 150$ seconds), to obtain a well performed primary pairing classifier. In the use of CoSafe, more motion data can be collected and utilized to further improve the performance of itself.

7.4 Real-World Attack Experiments

We run real-world attack experiments which have similar settings as the data collection of *Trace B*. We compare CoSafe with the following three schemes:

- *DTW-Based Scheme Proposed by J. Lester et al.* [16]: gait periods perceived on a pair of master and slave devices are compared by using Dynamic Time Warping (DTW). A lower DTW distance between two gait cycles indicates a higher similarity between the compared cycles.
- *Coherence-Based Scheme Proposed by M. Muhammad et al.* [15]: first, the Magnitude Squared Coherence (MSC) between two signals is calculated. To prevent all frequencies from being a unity magnitude, a windowing method called weighted overlapped segment averaging is utilized (a Hanning window of 50 samples is used). Second, use a normalized integration over the range from DC to 10 Hz (i.e., by multiplying the resulting integral by 0.1) to get a 0 to 1 measure of the coherence.
- *iGuard* [6]: a motion sequence of the behavior of taking out a smartphone is segmented into three consecutive segments, i.e., *walking*, *taking out the phone* and *walking*. For walking and taking out motions, DTW and logic regression classifier are used for estimating the probability that a motion is performed by the owner himself/herself (named PoS), respectively. A

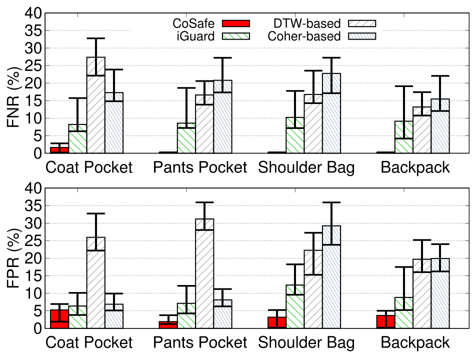


Fig. 14. FNR and FPR versus phone placement.

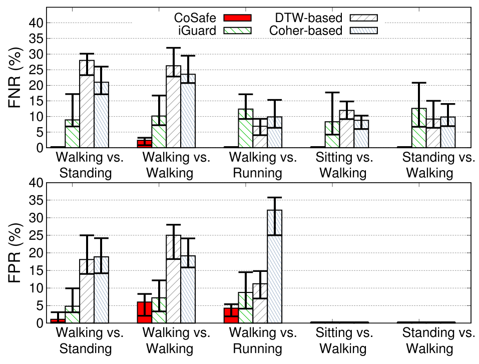


Fig. 15. FNR and FPR versus mobility states.

Markov-based model is introduced to incorporate PoS sequence and the transition probability between two consecutive motions, leading to a decision about whether the smartphone is stolen or not.

7.4.1 Zero-Effort Stealing Attack

We first examine the performance of CoSafe under zero-effort stealing attacks. Such an attack refers to an attacker without much knowledge or ability to predict the motion of owner, which might be the most common type of attacks against our mutual verification system.

Impact of Smartphone Placement. In this experiment, all victims walk along the same straight road. We examine *four* different positions (i.e., *coat pocket, front trousers pocket, shoulder bag and backpack*) of the smartphone carried by each victim. Three trials for each position. For each victim, 48 trials are conducted.

The statistical results under difference phone placements are displayed in Fig. 14. The upper and lower subfigures depict the average, maximum and minimum of FNR and FPR over all 10 volunteers, achieved by using CoSafe, iGuard, DTW-based and Coherence-based methods, respectively. It can be seen that CoSafe outperforms all other methods under all settings. The FNRs and FPRs of CoSafe are 1.63, 0, 0, 0 percent, and 5.26, 1.92, 3.18, 3.68 percent when the phone is placed in coat pocket, front trousers pocket, shoulder bag and backpack, respectively. Although the FPR increases slightly when the phone is placed in coat pocket, a pretty low FNR can still be guaranteed. The experimental results suggest that CoSafe is very reliable to use for common phone placements.

Comparatively, both DTW-based and Coherence-based methods result in high FNRs (around 20 percent) and FPRs (up to 30 percent). iGuard suffers high FPRs under shoulder bag and backpack settings (i.e., 12.34 and 8.83 percent). It is because that the decision of iGuard partially relies on the distinguishable patterns of taking out motions of a legal user, however, when the phone is placed in bag, the patterns of taking out motions may deviate from those learned in the training phase, which leads to false alarms.

Impact of the Mobility States of Victims and Attackers. In this experiment, both victims and attackers walk along the same straight road, and the smartphone is placed in the coat pocket of each victim. First, we examine three mobility states of attackers, i.e., *standing still, walking and running away versus a walking victim*. Second, we also examine two still states of victims, i.e., *sitting and standing, versus a walking attacker after stealing*. Three trials for each mobility setting. For each victim, 60 trials are obtained.

Fig. 15 shows the FNR and FPR of CoSafe over all volunteers, comparing with other three methods. It can be seen that CoSafe shows 0 percent FNRs in all settings, except an FNR of 2.38 percent in ‘walking-walking’ setting, and FNRs of iGuard in all settings are around 10 percent. The FNRs of DTW-based and Coherence-based methods are even higher. Especially, in ‘walking-standing’ and ‘walking-walking’ settings, their FNRs exceed 20 percent.

It can be seen that both CoSafe and the comparative methods lead zero FPR when victims are stationary, which is because of the big mobility differences between victims and attackers. In walking victim settings, CoSafe still keeps low FPRs (1.08, 6.01 and 4.25 percent). However, the FPRs of DTW-based and Coherence-based methods increase dramatically. That is because two devices are located on the different side of body, the corresponding motion sequences have an obvious dissimilarity. The FPRs of iGuard (4.81, 7.21 and 8.72 percent) are higher than CoSafe. Since, in iGuard, the similarity between walking segments before and after the taking-out motion impacts final decision, however, during walking, gait pattern of a victim may change over time.

Remarks. ‘walking-standing’ and ‘sitting-walking’ settings can also indicate two kinds of unintentional device loss, thus we do not conduct experiments of unintentional device loss separately. In particular, ‘walking-standing’ setting can be considered as the case that a walking user drops its smartphone at some place, while ‘sitting-walking’ setting implies that a user sitting or standing somewhere (e.g., in a restaurant) forgets to take its smartphone away when it leaves.

Impact of Terrain Environment. We examine *seven* different terrain environments, i.e., *straight, crooked and forked roads, stairs (up and down), corridor, and escalator* (see Fig. 11). For each victim, 84 trials are conducted. The smartphone is placed in the coat pocket of each victim.

From Fig. 16, we can see that, by using CoSafe, the FNRs are around 5 percent when victims walk on straight and forked roads, and down stairs. In other terrains, 0 percent FNRs are obtained. In most terrains, the FNRs of DTW-based and Coherence-based methods exceed 15 percent, and that of iGuard are around 10 percent. Furthermore, CoSafe maintains relatively stable FPRs around 4 percent,

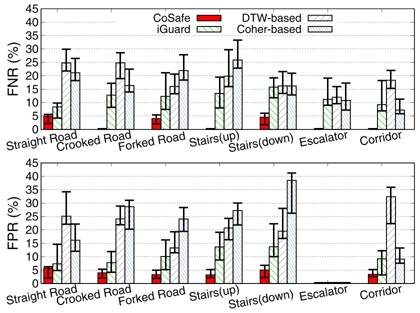


Fig. 16. FNR and FPR versus terrain.

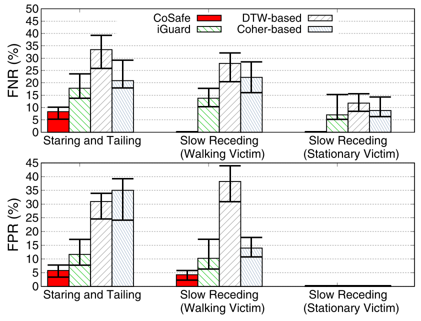


Fig. 18. FNR and FPR under sophisticated stealing attack.

and iGuard shows moderate FPRs, in most cases, while both DTW-based and Coherence-based methods suffer from high FPRs around 20 percent except escalator (in this setting, victims are standing on the escalator while stolen by passing stealers, which causes a sufficiently large motion differences for being identified. Consequently, low FPR is easy to be achieved), especially the FPR of Coherence-based method reaches to 38 percent in down stairs setting.

It is worth noticing that, iGuard shows high FNRs when victims walk up and down stairs (13.41 and 15.75 percent). We speculate that is because waking on stairs induces large variance of motion signals, which impedes iGuard from detecting 'take-out' motions in the signal. Meanwhile, the corresponding FPRs increase to 13.64 and 13.71 percent. The reason is that the inertial data exhibits larger variance, which incurs high noise in motion identification and comparison.

Impact of Transport Mode. We compare the performance of all schemes when victims and pickpockets are standing/walking on the ground and public transportations. Particularly, we do experiments on two common land transport modes, i.e., *subway trains* and *buses*. In the experiments, each victim takes out the phone and puts it back into position during a walking of 15 seconds, and then stands still for around 15 seconds during which a pickpocket launches a stealing attack. For each victim, 30 experiments are conducted by each transport mode.

It can be seen from Fig. 17 that the FNRs of CoSafe on subway trains and buses are 1.41, 1.89 percent, which are comparable with the FNR of standing on the ground (1.79 percent). For CoSafe, the largest average of FPR is 4.07 percent under the setting of taking bus. This is because bus routes are across the urban area, which introduces frequent sudden brakes and accelerations. Interestingly, the average FPR on subway trains is reduced to 1.29 percent, we speculate the reason is that subway trains move in constant speeds, and victims walk on trains with relatively low speed. Overall speaking, we can conclude that CoSafe is very reliable to normal land transport modes. In contrast, the DTW-based and Coherence-based methods lead to high average FNR and FPR around 20 percent (up to 29.71 percent) over all settings. iGuard also exhibits performance degeneration on trains and buses. Both FNRs and FPRs of it exceed 10 percent.

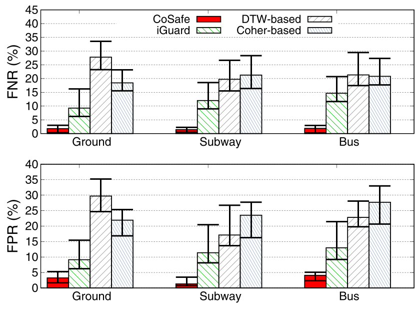


Fig. 17. FNR and FPR versus transport mode.

7.4.2 Sophisticated Stealing Attacks

In such attacks, pickpockets have common knowledge about our design, and mimic the movement periods of victims, trying to fool our CoSafe. In order to test the reliability of CoSafe under sophisticated stealing attacks, we develop a set of experiments in which victims walk along or stand on a straight road, and pickpockets perform according to the following two strategies:

- *Staring and Tailing Attack:* a pickpocket is asked to follow a victim close enough throughout the trail to make his or her step speed identical to that of the victim, and to take the smartphone out the pocket of victims while paying attention to the step speed as well. Notice that, after obtaining the smartphone, the attack still follows the victim until the end of the trial.
- *Slow Receding Attack:* we consider both walking and stationary victims. For a walking victim, a pickpocket performs the same as *staring and tailing attack*, except receding from the victim very slowly after obtaining the smartphone. For a stationary victim, a pickpocket stands beside a stationary victim while taking the smartphone out of the pocket of the victim. After obtaining the smartphone, the pickpocket recedes from the victim very slowly until the end of the trial.

For each victim, 60 trails are conducted, and the smartphone is placed in the coat or pants pocket of it. Fig. 18 depicts the performance of CoSafe and the comparison methods under the above attacks. We observe that under staring and tailing attacks, the average FNR of CoSafe is 8.34 percent, which increases slightly comparing with that of zero-effort stealing attacks. We argue that, during such attacks, the pickpocket never walks away from the victim, however, the pickpocket has to leave the victim eventually. Noticing CoSafe can achieve 0 percent FNR under slow receding attacks, it implies that as soon as the pickpocket begins to move gradually away from its victim, CoSafe will detect the theft in the first coming WoI. On the contrary, both iGuard and other two schemes show high FNRs, especially for walking victims. It should be mentioned that iGuard claims that a pickpocket will speed up after stealing, and utilizes such speed change to facilitate theft detection. However in above attacks, pickpockets either follow the victims walking at normal pace or slow down, which renders the related design useless, thus the FNRs of it increase significantly. Especially for walking victims, the FNRs of iGuard are 17.82 and 13.76 percent. We emphasize that in daily usage, the two error rates are not equally important. False negatives will result in device loss, while false alarms may only make the users slightly uncomfortable, thus, high FNRs are unacceptable by users. Furthermore, we can see that the FPRs of CoSafe under the two attacks are comparable with that under zero-effort stealing attacks. Overall, we can conclude that CoSafe is reliable under sophisticated stealing attacks.

7.5 Power Consumption

We use the functions `processMiscUsage()` and `processAppUsage()` in `BatteryStatsHelper` class to measure the power consumption, i.e., the multiplication of voltage (mV) and current (mA), of CoSafe according to its Uid, which equals to the summation of power consumptions of CPU, Wakelock, Bluetooth, and accelerometer. We measure the power consumption of the Google Nexus 4 smartphone, which runs the master application, and has build-in Android 4.4. Based on the analysis on *Trace B*, we find that 79.21, 36.25, 4.64 percent of WoIs are processed by the PC, CPM and FPC blocks, respectively. We break down the power consumption for each module and take the average of measurements. The average power consumptions (per hour) of Bluetooth, PC, CPM, and FPC are 98 mAh, 48 mAh, 6 mAh and 227 mAh, respectively. The average voltage on the smartphone is 3.67 mV. Then we can estimate the average power consumption of the master application is 456 mW. The battery capacity of Google Nexus 4 smartphone used in our experiments is 2100 mAh, i.e., $2100 \times 3.67 = 7707$ mWh. Thus, a full charged Nexus 4 smartphone running CoSafe has $7707/456 = 16.9$ hours of theoretical battery life.

8 Limitation and Discussion

We point out that the use of CoSafe has the following limitations:

- First, if a user carries only one mobile device, CoSafe fails, since mutual comparison cannot be conducted.
- Second, if the master or slave device does not equipped with wireless communication modules, CoSafe fails, since motion data cannot be exchanged.
- Third, if a user losts both master and slave simultaneously, for example, leaving behind both devices on the table, CoSafe fails, since their mobilities are consistent.

We also claim that there is no need to conduct device loss detection in all places. CoSafe may only work in public places where there is high risk of device loss (e.g., street, restaurant, station etc.). In such places, any of the devices should never be apart from its owner, thus it is reasonable to raise alarm as soon as mobility inconsistency between devices is detected. On the other hand, in private places, since there is no risk of device loss, CoSafe could be turned off automatically in private places, e.g., home, working place, which brings two benefits. First, avoiding high false positives caused by natural separations of two mobile devices, for example, when a person enters into its office or at home, he or she immediately takes off the watch and sets it on the desk, while brings the phone with him or her. Second, reducing power consumption of CoSafe. To this end, one simple solution is that once the master connects to a designated private Wi-Fi access point, CoSafe will sleep, until the connection is disabled.

9 Conclusions and Future Work

In this paper, we have proposed a continuous mutual mobility verification scheme, called CoSafe, for online mobile device loss detection. CoSafe is resilient to DoS and imitation attacks as it can identify fine mobility differences of devices caused by sophisticated attackers. Furthermore, CoSafe is a light-weighted protocol, leveraging a three-level decision strategy according to the difficulty of verifying mobility consistency between devices. CoSafe is quite reliable and works well in various conditions.

Nevertheless, CoSafe also has several limitations. For example, at the current stage, CoSafe conducts pairwise comparison, which may incur longer detection delays and high computation and communication costs when there are two or more slave devices. This also points out the direction of our future work. In the future, we will consider to leverage the broadcasting nature of wireless communication to reduce pairwise communications. In addition, besides the star topology, we will investigate more adaptive network topology, where one or more devices can act as masters based on their available computation power and residual battery power.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China Grant no. 61672151, 61972081, and 61772340, under Shanghai Rising-Star Program (Grant No.17QA1400100), Graduate Student Innovation Fund of Donghua University (Grant No. GSIF-DH-M-2019012), DHU Distinguished Young Professor Program and 2017 CCF-IFAA Research Fund.

References

[1] Cisco. Global-2020 forecast highlights, 2015. [Online]. Available: https://www.cisco.com/c/dam/m/en_us/solutions/service-provider/vni-forecast-highlights/pdf/Global_2020_Forecast_Highlights.pdf
[2] P. Dalton, "Best practices with byod," NCC_Group_whitepaper, Tech. Rep., 2017. [Online]. Available: https://www.nccgroup.trust/uk/our-research/best-practices-with-byod/
[3] FCC, "Phone theft in america:breaking down the phone theft epidemic," Federal Communications Commission, Tech. Rep., 2014. [Online]. Available: https://transition.fcc.gov/cgb/events/Lookout-phone-theft-in-america.pdf
[4] H. Mike. Blue watchdog turns your mobile phone into an all-purpose anti-theft device, 2010, [Online]. Available: https://newatlas.com/blue-watchdog-mobile-phone-all-purpose-anti-theft-device/16471/
[5] Amazon. Vnfire wireless bluetooth 4.0 anti-lost anti-theft alarm device tracker gps locator keydogcatkidswallets finder tracer w camera remote shutter & recording for iphone ipad & android 4.3 or above white, 2017. [Online]. Available: https://www.amazon.com/Vnfire-Bluetooth-Anti-lost-Anti-Theft-Recording/dp/B01265F826
[6] M. Jin *et al.*, "iGuard: A real-time anti-theft system for smartphones," in *Proc. IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.

[7] T. Vilarinho *et al*., "A combined smartphone and smartwatch fall detection system," in *Proc. Int. Conf. Comput. Inf. Technol. Ubiquitous Comput. Commun. Depend. Auton. Secure Comput. Pervasive Intell. Comput*., 2015, pp. 1443–1448.

[8] M. Shoaib, S. Bosch, H. Scholten, P. J. Havinga, and O. D. Incel, "Towards detection of bad habits by fusing smartphone and smartwatch sensors," in *Proc. Workshop Sens. Syst. Appl. Using Wrist Worn Smart Devices*, 2015, pp. 591–596.

[9] A. D. Wilson and H. Benko, "Crossmotion: Fusing device and image motion for user identification, tracking and device association," in *Proc. ACM ICMI*, 2014, pp. 216–223.

[10] F. Nurwanto, I. Ardiyanto, and S. Wibirama, "Light sport exercise detection based on smartwatch and smartphone using k-nearest neighbor and dynamic time warping algorithm," in *Proc. 8th Int. Conf. Inf. Technol. Elect. Eng*., 2016, pp. 1–5.

[11] A. Muaremi, B. Arnrich, and G. Troster, "Towards measuring stress with smartphones and wearable devices during workday and sleep," *BioNanoScience*, vol. 3, no. 2, pp. 172–183, 2013.

[12] R. Mayrhofer and H. Gellersen, "Shake well before use: Authentication based on accelerometer data," *Proc. Int. Conf. Pervasive Comput*., 2007, pp. 144–161.

[13] D. Bichler, G. Stromberg, M. Huemer, and M. Low, "Key generation based on acceleration data of shaking processes," in *Proc. Int. Conf. Ubiquitous Comput*., 2007, vol. 4717, pp. 304–317.

[14] D. Ashbrook, D. Ashbrook, S. H. Lee, S. H. Lee, and S. Patel, "Airlink: Sharing files between multiple devices using in-air gestures," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput*., 2014, pp. 565–569.

[15] J. Lester, B. Hannaford, and G. Borriello, ""Are you with me?" - using accelerometers to determine if two devices are carried by the same person," *Proc. Int. Conf. Pervasive Comput*., 2004, vol. 3001, pp. 33–50.

[16] M. Muaaz and R. Mayrhofer, "Smartphone-based gait recognition: From authentication to imitation," *IEEE Trans. Mobile Comput*., vol. 16, no. 11, pp. 3209 – 3221, Nov. 2017.

[17] A. Sarkisyan, R. Debbiny, and A. Nahapetian, "Wristsnoop: smartphone pins prediction using smartwatch motion sensors," in *Proc. IEEE Int. Workshop Inf. Forensics Secur*., 2015, pp. 1–6.

[18] B. SI G. Bluetooth core 4.0, 2018, [Online]. Available: https://www.bluetooth.com/

[19] A. Grinsted, J. Moore, and S. Jevrejeva, "Application of the cross wavelet transform and wavelet coherence to geophysical time series," *Processes Geophys*., vol. 11, pp. 561–566, 2004.

[20] C.-C. Chang and C.-J. Lin, Libsvm – A library for support vector machines, 2016. [Online]. Available: https://www.csie.ntu.edu.tw/~cjlin/libsvm/

[21] M. Shahzad, A. X. Liu, and A. Samuel, "Secure unlocking of mobile touch screen devices by simple gestures:you can see it but you can not do it," in *Proc. Int. Conf. Mobile Comput. Netw*., 2013, pp. 39–50.

Shan Chang received the PhD degree in computer software and theory from the Xián Jiaotong University, in 2013. From 2009 to 2010, she was a visiting scholar with the Department of Computer Science and Engineering, the Hong Kong University of Science and Technology. She was also a visiting scholar with BBCR Research Lab, the University of Waterloo, from 2010 to 2011. She is now an associate professor with the Department of Computer Science and Technology, Donghua University, Shanghai. Her research interests include security and privacy in mobile networks and sensor networks. She is a member of IEEE Computer Society, Communication Society, and Vehicular Technology Society.

Hang Chen received the BS degree from the Shanghai University of Engineering and Science, in 2017. He is working toward the postgraduate degree in Department of Computer Science and Technology, Donghua University, Shanghai. His research interests include Internet of Things, mobile sensing and wearable computing. He is a member of the ACM.

Hongzi Zhu received the PhD degree in computer science from Shanghai Jiao Tong University, in 2009. He was a postdoctoral fellow with the Department of Computer Science and Engineering at Hong Kong University of Science and Technology and the Department of Electrical and Computer Engineering at University of Waterloo, in 2009 and 2010. He is now an associate professor with the Department of Computer Science and Engineering in Shanghai Jiao Tong University. His research interests include vehicular networks, mobile sensing, and computing. He received the Best Paper Award from IEEE Globecom 2016. He is an associate editor for *IEEE Transactions on Vehicular Technology* and *IEEE Internet of Things Journal*. He is a member of the IEEE Computer Society, Communication Society, and Vehicular Technology Society.

Xinggang Hu received the BS degree from Anhui Polytechnic University, in 2017. He is currently working toward the postgraduate degree in the Department of Computer Science and Technology, Donghua University, Shanghai. His research interests include Internet of Things, mobile sensing, and wearable computing. He is a member of the ACM.

Di Cao received the BS degree from Donghua University, in 2017. She is currently working toward the postgraduate degree in the Department of Computer Science and Technology, Donghua University, Shanghai. Her research interests include distributed deep learning, security, and privacy. She is a member of the ACM.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.