The following publication H. Zheng and H. Hu, "MISSILE: A System of Mobile Inertial Sensor-Based Sensitive Indoor Location Eavesdropping," in IEEE Transactions on Information Forensics and Security, vol. 15, pp. 3137-3151, 2020 is available at https://doi.org/10.1109/TIFS.2019.2944034.

MISSILE: A System of Mobile Inertial Sensor-Based Sensitive Indoor Location Eavesdropping

Huadi Zheng and Haibo Hu, *Member, IEEE*

***Abstract*—Privacy concerns on smartphones have been raised by the public as more and more personal data are now stored on them. In this paper, we show that location information can be compromised through mobile inertial sensors which are considered insensitive and accessible by any mobile application in both iOS and Android without special privilege. We present MISSILE, an automatic system that can infer users' indoor location using labeled sensor data as prior knowledge. The key idea is that when a user reaches a particular indoor location, it is very likely that he/she has passed through some unique interior structures of a building, such as winding corridors, fire stop doors or elevators. These structures exhibit repeatable motion and environment patterns in mobile sensors that can be recognized by supervised learning. In our MISSILE system, the location labels of training data are automatically attained by Bluetooth beacons deployed in sensitive locations. With effective feature extraction procedure robust modeling, MISSILE shows good success rate for inference attack. For example, in a university campus with 15 sensitive locations, MISSILE achieves up to 73% correct prediction score whereas a random guess can only achieve $1/(15+1) = 6.25\%$. Further improvements on system performance and countermeasures are also discussed.**

***Index Terms*—Mobile Sensing, Location Eavesdropping, Side-Channel Attack, Supervised Learning**

I. Introduction

THE increasing sensory capability and accuracy in mobile and wearable devices have nourished many convenient applications, such as turn-by-turn navigation, fitness tracking, virtual reality, and interactive mobile game. However, privacy infringement arising from these applications has recently drawn much attention throughout the world. To combat this, US Federal Trade Commission has filed more than 130 lawsuits against spyware and 50 against general violation of privacy corruption practice [1]; and EU has adopted the more stringent "General Data Protection Regulation" (GDPR) to supercede the "Data Protection Directive" and enforced it in May 2018 [2]. Unfortunately, as more and more personal data, such as locations, passwords and daily schedules, are accessed through smartphones, even the best practice of privacy protection cannot protect them against mobile attacks that exploit side-channel information, such as UI state [3], power usage [4], or cellular network signal strength [5].

In the literature of side-channel attacks, many works have succeeded in exposing information about victim's location such as tracking their driving or public transport routes without using GPS, either through cellular/Wi-Fi networks [5] or by inertial sensors [6], [7]. However, these works focus on outdoor location, so it remains unresolved on the risk of **indoor** location leakage from unprivileged sensory data, which are usually more private and sensitive. In this paper, we develop the **m**obile **i**nertial **s**ensor-based **s**ensitive **i**ndoor **l**ocation **e**avesdropping (**MISSILE**) system to infer sensitive indoor locations using side-channel information only from unprivileged sensors such as accelerometer, gyroscope and magnetic field sensor. Our key idea is to identify a sensitive indoor location (e.g., an office) using multiple structural characteristics (e.g., turnings in a corridor, pausing of motion to open a fire stop door, or taking an elevator). These characteristics lead to unique patterns in sensor readings and constitute the signature of this location.

There are four challenges in MISSILE, namely, to acquire reliable location labels, to handle data inconsistency caused by device placement and movement, to transform raw data into features, and to build an effective learning model. To address these challenges, we propose a general-purpose machine learning system without prior knowledge of structural characteristics. To feed this system with sufficient training data, we develop an automatic location labeling mechanism using Bluetooth beacons with latency calibration method. Raw sensory data collected from different sources are made consistent with normalization and noise reduction techniques. After an efficient feature extraction procedure, calibration for anomalies is further applied in modeling to reduce the impact of data contamination from mislabeling and low-quality sensor output. Finally, a lightweight classifier is trained and embedded in a spyware to eavesdrop a victim's sensitive indoor location. Through our extensive experiments in a real indoor environment, we show the feasibility of MISSILE and the high risk of indoor sensitive location eavesdropping. To complement this research, we also discuss the potential extension of this attack and two countermeasures in addition to lifting up the privilege requirement of accessing sensory data.

To summarize, our contributions of this study are in the following three perspectives.

1) We adopt a general adversarial framework for side-channel attacks on mobile devices, based on which we propose our indoor location eavesdropping attack.
2) We develop a real-life indoor location eavesdropping attack system which comprises automatic data labeling, data processing and machine learning pipeline on mobile inertial sensor data.
3) We propose a labeling mechanism with BLE beacons and a calibration method to compensate for latency using maximum likelihood estimation.
4) We conduct extensive experiments to demonstrate the feasibility of such an attack and thus the risk of indoor

© 2019 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

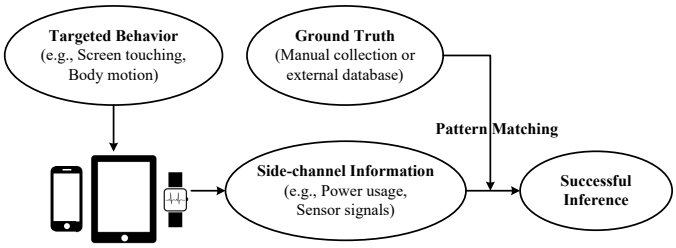


Fig. 1. The framework for side-channel attack on mobile devices

location exposure in practice.

The rest of this paper is organized as follows. Section II formally defines the privacy problem from side-channel attack and threat model with challenges. Section III dives into the detail of MISSILE system and its associated algorithms. Section IV presents the system evaluation and severity of this threat. We further discuss the extension of this system and potential countermeasures in Section V, followed by a review of related work on unprivileged mobile sensor side-channel attacks and indoor positioning in Section VI. Finally we draw our conclusion in Section VII.

II. Problem Statement

A. General Side-channel Attack Framework

A typical side-channel attack on mobile devices is described in Fig. 1. Side channels in these devices may react to user interaction or exterior environment change, which can be exploited by attackers to infer sensitive information. For example, a slight but distinct acceleration change in motion sensor can leak a user's keystroke on soft keypad. A direct consequence of such attack is the loss of users' privacy, which may further lead to even more serious attacks such as social engineering on the victim, blackmailing ransomware, and hijacking. As most sensors (especially multiple inertial sensors) do not require permission to access, such attacks can be camouflaged in normal applications, which makes them hard to detect. Based on this general adversarial framework, in what follows we define the MISSILE attack on indoor sensitive locations.

B. MISSILE Motivating Scenario

We assume there are a finite number of sensitive locations within the premises concerned (such as a campus, a shopping center, or a hospital). An adversary would like to stalk the daily routine of a frequent visitor (such as students/staff in a university campus) and eavesdrop whether and how often a victim user visits some sensitive location such as an office room, a particular clinic, or even restroom. We assume the adversary can intrigue the victim to install a legitimate application on her mobile phone.[1] Victim users are often tricked into downloading such apps especially when they do not require special permissions such as location. For example, Kaspersky Labs found and removed 58,000 instances of stalkerware in 2018 [9]. Even popular "trustworthy" apps might have vulnerabilities that open the door for spying, such as the one found in Whatsapp that allows injection of spyware onto people's phones [10]. Through such applications, the adversary can then collect unprivileged sensory information on the victim's mobile device in the background (both Android and iOS allow such collection without permission). A classifier embedded in this application can then identify the unique sensor pattern of a sensitive location.

The sensors considered in this paper include accelerometer, linear acceleration sensor, gyroscope, and magnetic field sensor. While accelerometer and linear acceleration sensor are common motion sensors shipped in modern mobile devices for detecting device acceleration, gyroscope is another important sensor. By detecting a sudden turn or a subtle slow winding, it indicates if a victim user is changing his/her direction in a regular degree due to a hallway or corridor. Magnetic field sensor is an environmental sensor whose readings change as the victim user moves indoor. Magnetic local variation exists in all buildings due to the geolocation and magnetic materials used in construction (e.g., a large amount of steel in an elevator) [11]. By combining the above sensor readings in mobile devices, each sensitive location may have a unique pattern in the sensory data stream for location inference.

C. Threat Model

The major threat comes from a mobile application that only silently collects sensory data and eavesdrops sensitive location. In this paper, we assume the attacker and its client-side application has the following capabilities or characteristics:

Adversary Application and Network: For both Android and iOS, application packages from any sources can be installed on the devices.[2] As such, the malicious party can easily develop legitimate spyware or repackage popular applications (such as Facebook, Messenger, and WhatsApp) with malicious codes and distribute them through social networks, third-party app markets or emails. We assume this application has network access, either Wi-Fi or cellular network, to upload the eavesdropping results to or update the classifier regularly from the attacker's server.

Stealthy Side Channels: Side channels obtained from the unprivileged accelerometer, gyroscope, and magnetic field sensor are accessible to the adversary application. While both Android and iOS have permission protection mechanism for GPS, Wi-Fi and Bluetooth (iOS and Android over 6.0 require on-the-fly approval), there is no specific permission protection for sensors on both operating systems. The attack is not assumed to be zero-permission. Instead, since the attack only targets at the permission-free inertial sensors, no additional permissions (particularly location-related) are needed. In other words, any installed application can acquire sensor readings without the consent or even knowledge of users. Existing antivirus apps cannot prevent MISSILE from running in the background as

[1]Some studies have also revealed the possibility of attaining sensor data through web browsers using Javascript [8], notwithstanding limited sensor types (e.g., motion sensors only) and sampling frequency.

[2]Apple Developer Enterprise Program allows a developer to create and distribute custom apps to any iOS device without submitting them to App Store.

MISSILE only monitors sensor readings with low CPU and battery consumption like most legitimate applications.

Computational Power and Machine Intelligent: The application can access the CPU (or even GPU) of the mobile device for sensory data processing and classification. Nonetheless, the computationally intensive training of the classifier is still performed on the server side. However, as the computational capability of mobile devices keeps increasing, especially with the advent of dedicated AI chip on SoC (e.g., ARM Machine Learning Processor), certain machine learning tasks can be processed on the mobile devices to offload the MISSILE server and improve location inference response time.

D. Technical Challenges

Indoor pedestrian location inference using sensory data is more challenging than route inference in outdoor environments [6], [12]. We summarize four major challenges as below.

Reliable Label Acquisition: To perform indoor location eavesdropping attack, we need to capture sensor readings with proper labels. As GPS and open map data are usually not available under indoor scenarios, an automatic, highly-efficient, and reliable mechanism is needed to collect a large number of location labels as ground truth for training data.

Data Inconsistency: Since the output coordinates of inertial sensors depend on the relative posture of mobile devices, we need to normalize various device placements such as vertically in a pocket or horizontally in a handbag. Furthermore, motion sensors capture not only the location pattern but also the walking style of users. The diversity of walking speed and moving behavior of individuals has a negative impact on the inference as it causes inconsistency in sensory data.

Raw Data Optimization: Raw sensor values are not suitable to be directly fed into a machine learning pipeline since processing high-dimensional and high-frequency data consumes a significant amount of computational resources. To maximize the attack performance under limited computational power, we need an optimal set of low-dimensional features selected by an automated feature extraction procedure without prior information of location details.

Robust Modeling: The collected training data may be in low quality, as they can be contaminated by corrupted devices, label signal delay, and internal software or device faults. The performance of the machine learning model can be impacted by such anomalies and therefore a robust model with anomaly calibration is always preferred.

III. MISSILE System

In this section, we first present the overall design of MISSILE system, followed by the detailed discussion on individual component implementation.

A. Design Overview

As shown in Fig. 2, the proposed MISSILE system is composed of two stages.

In the training stage, the attacker first identifies target indoor sensitive locations, physically walks through these locations

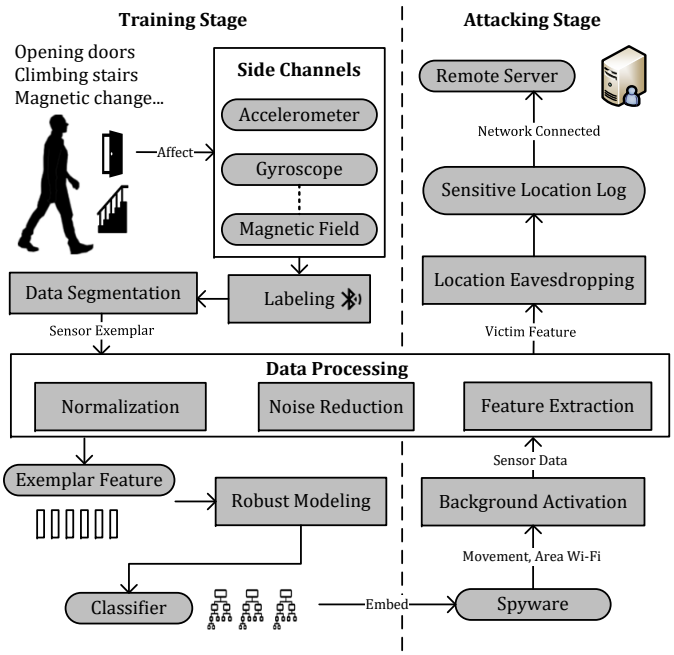


Fig. 2. Overview of MISSILE system

with stock mobile devices, and collects sensor readings as they pass by these locations.[3] To automate the collection process and increase its accuracy, MISSILE deploys a Bluetooth Low Energy (BLE) beacon in each sensitive location to activate sensor readings automatically as the attacker walks by. BLE beacons (e.g. Apple's iBeacon) are small, inexpensive, and long-lasting devices that continuously emit identifiable radio signals in the neighborhood (normally within a range of up to 10 meters in our system). In practice, BLE beacons have been widely deployed by many indoor positioning services for navigation and advertisement, so the attacker can even leverage these existing beacons without any extra deployment cost.

To acquire a desired length of data with proper label, **segmentation** is performed on the long continuous data stream. Such a small segment from the whole stream is called an **exemplar**, which is assumed to contain the unique signature of a sensitive location. The length of exemplar is a hyperparameter that ensures it is long enough to contain the desired sensor data patterns. Each exemplar can contain signals from multiple sensors to capture a comprehensive set of location characteristics, so that they can reveal the structural (e.g., door opening, stairs walking), ambient (e.g., magnetic field), and even environmental (e.g., air pressure) patterns and increase robustness against dynamic environment such as high user density.

Exemplars are further normalized to resolve the inconsistency problem of device placement. Noise reduction is applied next due to the high-density noise in normalized exemplar from body movement. Then automatic feature extraction is performed to obtain an optimal low-dimensional representation of the sensor pattern. The generality of this procedure

[3]Many premises are semi-private/semi-public and accessible to the attacker. For example, everyone can enter most of the buildings in a university campus or a hospital even though they are privately owned premises.

Algorithm 1 Starting Timestamp Determination

Input: RSSI sequence
$S = \{s_1, s_2, \ldots, s_n\}$
$s_i = \{timestamp, major, minor, rssi\}$
Step threshold st
RSSI threshold rt
Output: Starting points $Start$
Procedure:
CP = $\emptyset$, $Pruned$ = $\emptyset$, $Start$ = $\emptyset$
Segment S into sub-sequences $\{S_1, S_2, \ldots\}$ with same $minor$
for each sequence S_i **do**
    Add all climbing points in S_i to CP
for $j = 1$ to $|CP|$ **do**
    if $CP_j.rssi \leq rt$ **then**
        Prune j-th point from CP
$CP = DescendSortRSSI(CP)$
for k = 1 to $|CP|$ **do**
    if CP_k not in $Pruned$ **then**
        $l = CP_k.timestamp - st$
        $r = CP_k.timestamp + st$
        $O = FindOverlap(l, r, CP, CP_k.major)$
        $Pruned$ = $Pruned \cup O$
$Start$ = CP - $Pruned$
Return $Start$

allows the attacker repeat MISSILE to other premises without knowledge of the actual sensor pattern and feature engineering. When the features of clean exemplars are ready, a robust supervised learning scheme using anomaly calibration technique is used to construct a classifier to recognize the sensor pattern for each sensitive location.

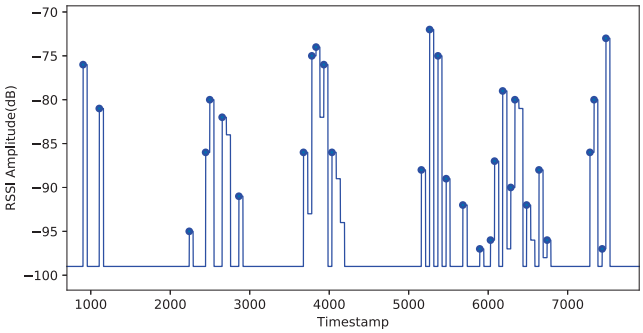
In the attacking stage, the attacker embeds this classifier into a legitimate mobile application for victims to install on their mobile devices. This application then continuously collects the sensor readings in the background and captures indoor sensitive locations when the expected sensor patterns occur. To preserve battery life, two activation techniques are introduced to reduce unnecessary eavesdropping when the victim is far from the concerned premises or is stationary. Finally, the eavesdropped sensitive location log can be delivered to the attacker when the network is available.

B. Labeling and Data Segmentation

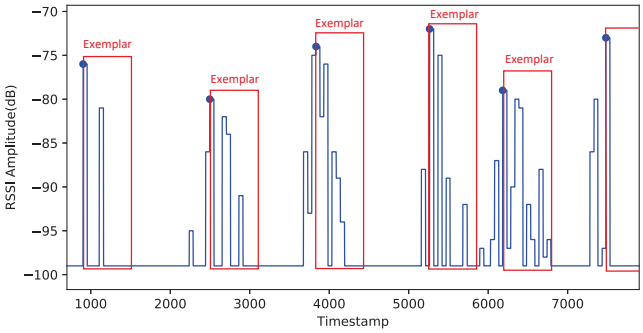
The first key component is to segment the short, recognizable pattern exemplar of sensitive locations from the stream of continuous sensory readings. To determine the starting timestamp of an exemplar, an intuitive choice is to use the estimated distance from the beacon. However, since this distance is hard and inaccurate to estimate,[4] we instead use the raw Received Signal Strength Indicator (RSSI) and its change. Typically, RSSI ranges between around -20dB to -80dB in short proximity and less than -95dB in the farthest distance under the setting of experiment deployment.

1) Climbing Point as Starting Timestamp: Since this is the training stage, the attacker can have the full control to keep the device moving while collecting the BLE signals. The challenge

[4]Theoretically, we can estimate the distance between a receiver and a beacon based on the received signal and the reference signal strength of 1-meter distance. However, due to the environmental absorption and power change, such distance estimation can suffer from significant delay and fluctuations.



(a) Potential starting timestamps of exemplar

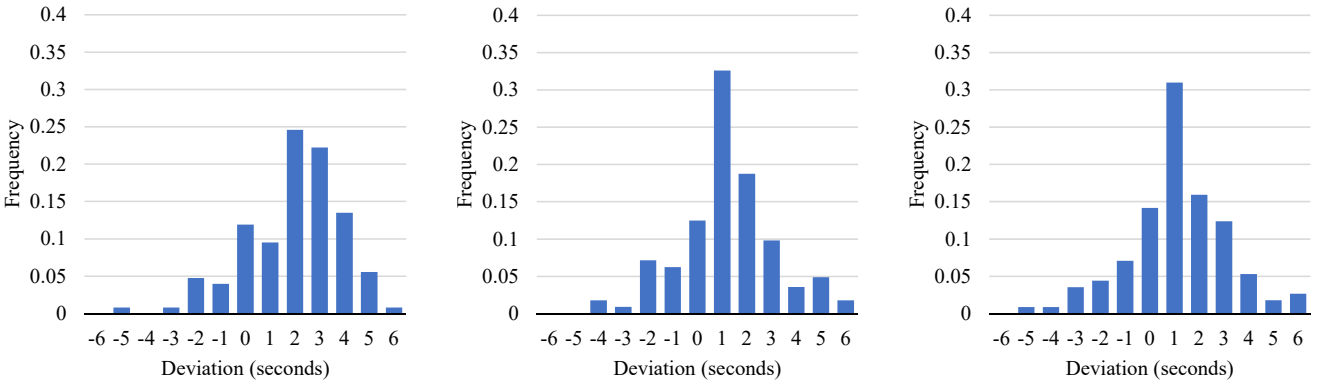


(b) Selected starting timestamps with st = 750 and rt = -85dB

Fig. 3. Exemplar timestamp is detected in RSSI sequence collected from one reference point. The sensitivity is set to -99dB since RSSI lower than this threshold occurs from a remote beacon.

in segmentation is to determine the *starting* timestamp of a potential sensor pattern that indicates a sensitive location is reached. Intuitively, this timestamp should be associated with a maximal RSSI value (i.e., a climbing point). However, due to the fluctuation of radio signals, there are multiple climbing points when walking through a location, as illustrated in Fig. 3(a). To resolve the true starting timestamp, we introduce two thresholds to prune climbing point candidates caused by signal delay and other factors. *Step threshold* is the minimum length between two starting timestamps (of two locations), and the RSSI threshold defines the minimum RSSI for a starting timestamp. The former is based on the fact that sensitive locations are discrete and fall apart with one another, whereas the latter is based on the fact that the starting timestamp is usually associated with a strong RSSI. When multiple reference points are available in a location, we leverage the metadata emitted by the beacons to separate signal sources, namely major identifier and minor identifier. In our setting, major identifier denotes the location while minor identifier denotes the beacon itself.

Algorithm. 1 describes the details of determining starting timestamp for an exemplar. It first separates the RSSI sequence from different beacons by $minor$. After deriving all RSSI climbing points from each beacon, we store them in the set of CP as shown in Fig. 3(a). A point is defined as *climbing* if the current RSSI is larger than its previous moment in the sub-sequence. Based on the provided RSSI threshold rt, all climbing points whose RSSI values are below rt are pruned. The algorithm iteratively sorts and accesses remaining points in descending order of their RSSI values. In each iteration,



(a) 2 meter proximity of Beacon A (b) 1 meter proximity of Beacon A (c) 1 meter proximity of Beacon B

Fig. 4. The deviation histogram of derived starting timestamp

only one climbing point is retained for each beacon within the step threshold st while all other climbing points with different $major$ (i.e., signals from other locations) are pruned. After this step, only those strong climbing points survive in $Start$, the candidate set for starting timestamps of sensor pattern. Fig. 3(b) illustrates a running example of this algorithm for one reference point (i.e., RSSI measurements taken from one beacon for each location), where a red rectangle denotes an exemplar of length 15 seconds (i.e., 750 samples under a 50Hz sampling rate). In what follows, we propose a calibration method to refine this starting timestamp to further compensate for the latency of BLE signal.

2) Calibration for Latency: The latency of detecting BLE beacon signal consists of both discovery latency and propagation latency. The former arises from the fact that BLE is a slotted protocol that periodically sends data packet in designated time slots and sleeps in between. The emitting interval between two consecutive slots can range from 100ms to 2000ms. Discovery latency happens when broadcast packets miss the scanning window of a receiving mobile device, which has low BLE scanning frequency by default. Since data collection is managed by attacker, such latency can be significantly reduced by minimizing the emitting interval and maximizing the scanning frequency [13].

The propagation latency is caused by radio signal propagation due to absorption, congestion or reflection. Although such latency could be large and fluctuating in general, we only care about the latency when the device is in close proximity to the BLE beacon to annotate the starting timestamp. As shown in Fig. 4, we plot the deviation between the actual starting timestamp, which is recorded manually, and the derived starting timestamp from Algorithm 1 under different proximity distances and beacon models. We observe that the deviation can be approximated by Gaussian distribution with a mean proportional to its proximity distance. Under this assumption, we propose a calibration method using *Maximum Likelihood Estimation* (MLE) [14] as follows to refine the annotated starting timestamp.

According to *Bayes' Theorem*, the conditional probability of actual starting timestamp t given a derived starting timestamp α from RSSI is

$$P(t|\alpha) = \frac{P(\alpha|t)P(t)}{P(\alpha)}.$$

Since both P(α) and P(t) are constant (because both α and t are uniformly distributed drawn from their domains), maximizing $P(t|\alpha)$ is equivalent to maximizing P($\alpha|t$), and further P($\alpha - t|t$), the conditional probability of deviation $\alpha - t$. According to our assumption, the latter follows a Gaussian distribution, i.e., $P(\alpha - t|t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(\alpha-t-\mu)^2}{2\sigma^2}}$. Therefore, we can calibrate the starting timestamp t^* by maximizing the following likelihood

$$\begin{aligned} t^* &= \arg\max_t \ P(\alpha - t|t) \\ &= \arg\max_t \ \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(\alpha-t-\mu)^2}{2\sigma^2}} = \arg\min_t \frac{(\alpha - t - \mu)^2}{2\sigma^2}. \end{aligned}$$

After solving the above equation, we have

$$t^* = \alpha - \mu.$$

The above equation means that in the single reference point case, the calibration can simply be carried out by deducting a mean deviation from the derived starting timestamp.

Now we generalize the derivation to the case of two reference points (e.g., beacons on both sides of the location) whose derived timestamps are α_1 and α_2 respectively.[5] The joint conditional likelihood of α_1 and α_2 can be derived from their individual distribution independently:

$$P(\alpha_1, \alpha_2|t) = P(\alpha_1|t) \cdot P(\alpha_2|t)$$

Similar to the single reference point case, we can calibrate the starting timestamp t^* by maximizing the joint likelihood of $\alpha_1 - t$ and $\alpha_2 - t$ instead:

$$\begin{aligned} t^* &= \arg\max_t \ P(\alpha_1 - t|t) \cdot P(\alpha_2 - t|t) \\ &= \arg\max_t \ \frac{1}{\sigma_1\sqrt{2\pi}} e^{-\frac{(\alpha_1-t-\mu_1)^2}{2{\sigma_1}^2}} \cdot \frac{1}{\sigma_2\sqrt{2\pi}} e^{-\frac{(\alpha_2-t-\mu_2)^2}{2{\sigma_2}^2}} \\ &= \arg\min_t \ \frac{(\alpha_1 - t - \mu_1)^2}{2{\sigma_1}^2} + \frac{(\alpha_2 - t - \mu_2)^2}{2{\sigma_2}^2} \end{aligned}$$

[5]We assume no collision in the beacon signal as BLE can transmit through 40 channels.

As such,

$$t^* = \frac{{\sigma_2}^2(\alpha_1 - \mu_1) + {\sigma_1}^2(\alpha_2 - \mu_2)}{{\sigma_1}^2 + {\sigma_2}^2}.$$

As we observe from Fig. 4, the deviation follows the same distribution given the same beacon model and proximity distance. Therefore, we can simplify t^* by setting $\sigma_1 = \sigma_2 = \sigma$ and $\mu_1 = \mu_2 = \mu$:

$$t^* = \frac{\sigma^2(\alpha_1 - \mu) + \sigma^2(\alpha_2 - \mu)}{\sigma^2 + \sigma^2} = \frac{\alpha_1 + \alpha_2}{2} - \mu.$$

The above equation means that in case of two or more reference points, the calibration can simply be carried out by deducting a mean deviation from the average of all derived starting timestamps.

C. Normalization and Noise Reduction

Most inertial sensors (e.g., accelerometer) produce 3-dimensional readings in a coordinate system that is relative to the device's screen. As such, different device placements cause inconsistency of the sensor readings even when they come from the same location. Another key factor in data consistency is irrelevant noise caused by body movement. For example, walking has a major impact on motion sensors especially when the device is placed close to the leg (e.g., in the pant pocket). In such cases, the sensor signals caused by body movement can overshadow those caused by the physical environment.

1) Resolving Inconsistency by Device Placement: A straightforward solution is to convert the screen-based 3-axis coordinate vector, such as the accelerometer vector $A = [a_x, a_y, a_z]$, into an absolute value by taking the Euclidean norm:

$$\|A\| = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

This scalar is independent of device placement, but the details of device movement on each axis are removed. To preserve the details, we adopt the rotation-based normalization which transforms screen-based coordinate into world reference coordinate [15]. In what follows, we use the gravity sensor vector and the magnetic field sensor vector as example. Note that the former points to the core of the earth while the latter always provides an approximate geographical pole direction. A rotation matrix which maps between the screen-based coordinate and world coordinate can be derived as follows.

Unit vector of a vector v can be obtained from $v_u = \frac{v}{\|v\|}$. Let G and M be the unit vector of gravity and magnetic field in device reference, the cross product of G and M must be perpendicular to the plane spanned by G and M. Since M lies on the plane spanned by the gravity vector and south-north vector, this cross product produces vector EW, i.e., the west-east vector. Similarly, the south-north vector SN is the cross product of G and unit vector of EW:

$$G = [g_x, g_y, g_z]^\top, \quad M = [m_x, m_y, m_z]^\top$$

$$M \times G = EW, \quad G \times EW = SN$$

As such, we can use the unit vectors of EW, SN, G to form a rotation matrix that connects screen-based coordinate and world reference coordinate. To rotate a new sample K into world coordinate, we multiply it with the inverse of rotation matrix R as follows

$$R = \begin{bmatrix} ew_x & sn_x & g_x \\ ew_y & sn_y & g_y \\ ew_z & sn_z & g_z \end{bmatrix}$$

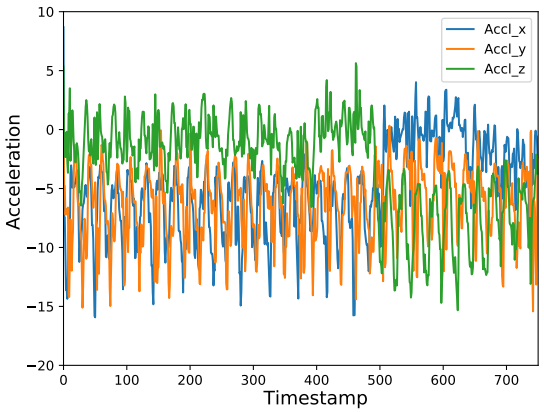
$$R^{-1} \cdot K = K_{rotated}$$

Obviously the computational cost of the rotation-matrix-based normalization is higher than the Euclidean-norm-based normalization, as the former involves matrix inverse and multiplication. In our experiment, we measure their CPU time (see Table I), and the latter is more than 20 times faster. Nonetheless, the former preserves more details in each axis and our experimental results in Table V show that the former consistently outperforms the latter in terms of F1-score under various classifiers.

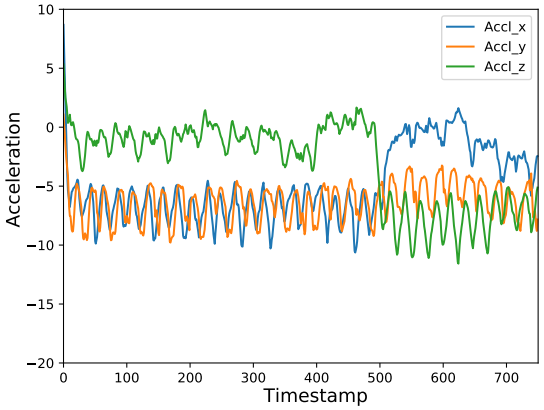
2) Resolving Inconsistency by Body Movement Noises: Body movement noises are mostly distributed in the high-frequency spectrum while sensor signals corresponding to location patterns lie in the low-frequency spectrum. To illustrate

TABLE I
RUNTIME ON GOOGLE PIXEL

Approach	Execution Time (ns)
Coordinate Rotation	20000 - 25000
Euclidean Norm	600 - 900



(a) Accelerometer raw data



(b) Filtered data with $\alpha = 0.15$

Fig. 5. Turning event is more evident after filtering movement noises

Algorithm 2 Significant Features Extractor

Input: Exemplars $E = \{E_1, E_2, \ldots, E_k\}$, $E_i \in \mathbf{R}^{m \times n}$
Location labels $L = \{l_1, l_2, \ldots, l_k\}$
Rank threshold r
Output: Final features $F' = \{F'_1, F'_2, \ldots, F'_k\}$, $F'_i \in \mathbf{R}^{m \times r}$
Procedure:
$F = \emptyset$, $tempPV = \emptyset$, $PV = \emptyset$, $Type = \emptyset$, $F' = \emptyset$
for E_i in $(E_1, E_2, \ldots, E_k)$ **do**
  F_i = ExtractCommonFeatures(E_i)
  Add F_i into F
for feature type f in F **do**
  if f is binary feature **then**
    $tempPV$ = FisherTest(f, L)
  else if f is real-valued feature **then**
    $tempPV$ = MannWhitneyTest(f, L)
  Add $tempPV$ to PV
$Type$ = RankFeature(PV, r)
F' = SelectFeature($F, Type$)
Return F'

this, we use the accelerometer as an example. Fig. 5(a) shows raw accelerometer readings of a pedestrian who encounters a sudden turn when walking inside a building. The original raw data have such a dense signal distribution over the whole recordings that it is hard to discover important event from the time domain. Therefore, we apply a low-pass filter to this sequence. In particular, we choose a moving average filter for noise reduction, which derives the moving average from the original sensor data as

$$y_i = y_{i-1} + \alpha * (x_i - y_{i-1}),$$

where the filtered sample y_i is based on its previous value y_{i-1} and the current x_i with parameter α lying between 0 to 1. As illustrated in Fig. 5(b), by applying this filter the data are properly smoothed and the turning motion is more evident from the original noisy data.

D. Feature Extraction

Filtered exemplars are still in the form of raw sensor signals unsuitable for learning an effective model. To reduce the data volume for learning, we need to extract significant low-dimensional features from these exemplars. Features are commonly used in classification tasks to capture the properties of signal behavior. Further, since we assume the adversary has no prior domain knowledge on sensor patterns, this feature extraction and selection process should be fully automated without human intervention. In MISSILE, we adopt the FRESH procedure [16] to build an automatic significant features extractor, which remarkably reduces the effort on feature engineering. The detailed procedure is shown in Algorithm. 2, which consists of the following three phases.

1) Extraction of Feature Candidates : The raw time-series exemplars are first mapped into common features with a set of predefined parameters. Let us assume that there are k exemplars in the collection $E = \{E_1, E_2, ..., E_k\}$. For each exemplar, n samples are collected from each sensor axis and a total of m sensor axes are sampled. As such, each exemplar can be written as

$$E_i = \{(s^1_{i_{t1}}, s^1_{i_{t2}}, \ldots, s^1_{i_{tn}}), \ldots, (s^m_{i_{t1}}, s^m_{i_{t2}}, \ldots, s^m_{i_{tn}})\},$$

where $t1$ is the starting timestamp of exemplar E_i.

The features F_i of exemplar E_i are extracted from various statistics on samples including maximum, minmum and root mean square ($f_i^{rms_m} = \sqrt{\frac{\sum_{t1}^{tn} |s_i^m|^2}{n}}$):

$$F_i = \{(f_i^{max_1}, f_i^{min_1}, f_i^{rms_1}, \ldots), \ldots, (f_i^{max_m}, f_i^{min_m}, f_i^{rms_m}, \ldots)\}$$

2) Statistical Hypothesis Testing: After all the features are extracted, we need to select significant ones from them before feeding them into a classifier for learning. Statistical hypothesis test is conducted on each feature to evaluate its relevance to locations. The main idea is that, if a feature f can distinguish a particular location j_a, its conditional probability distribution on this location j_a, $P(f|j_a)$, must be significantly different from $P(f|j_b)$, the distribution on any other location j_b. Using this principle, the null hypothesis H_0^f and alternative hypothesis H_1^f to test relevance of feature f to location j_a are formulated as

$$\forall j_b \neq j_a,\ H_0^f = \{P(f|j_a) = P(f|j_b)\}$$
$$H_1^f = \{P(f|j_a) \neq P(f|j_b)\}$$

A set of probability values (p-value) PV will be returned after the tests. A smaller p-value suggests stronger evidence to reject the null hypothesis H_0^f, which means the feature is relevant to location j_a against location j_b since they do not share the same conditional distribution. In MISSILE, we use two hypothesis tests, namely, Fisher's exact test for those binary features and Mann–Whitney rank test for those real-valued features.[6]

3) Selection: In the final step, we sum up the total p-values across all axes for each feature and rank them in ascending order. Only top-r features which have the smallest p-values are selected as the refined feature set F'.

In MISSILE, we apply FRESH [16] with over 60 categories of pre-defined features. They can be divided into two sets. Time-domain features such as mean, variance, median, and the number of peaks mainly characterize signal intensity as in time series. For example, a magnetic field sensor produces different number of peaks based on the magnetic local variation in different locations. Frequency-domain features capture the characteristics of signal pattern in terms of frequency envelope and certain frequency component after Fourier transform. Certain location such as a winding corridor may not have obvious time-domain pattern but it has unique pattern on frequency domain. Table II shows the top-12 features after the selection step using our exemplar dataset. These features

[6]The Mann–Whitney rank test can examine the distribution of two real-valued random variables using statistics derived from ranking against each other. In the case of multiple locations, the test is conducted in one-vs-all style.

TABLE II
EMPIRICAL FEATURE EXTRACTION RESULT ON 4 SELECTED SENSORS

Feature Category	Description	Dimension	Average p-value
VAR	Variance of each axis s in a exemplar	1	$1.18 * 10^{-5}$
LSTD	Whether the standard deviation is larger than 0.25 times of max(s)-min(s)	1	$1.56 * 10^{-3}$
MEAN	Overall average of each axis s in a exemplar	1	$4.52 * 10^{-5}$
MEDIAN	Median value of each axis s in a exemplar	1	$5.92 * 10^{-5}$
SPKT	Cross power spectral density on the second coefficient	1	$6.24 * 10^{-6}$
RRSIGMA	Ratio of values more than the distance of 2 away from mean value	1	$1.99 * 10^{-6}$
PEAKS	Number of amplitude maxima with least support of 1 and 3	2	$2.66 * 10^{-4}$
LINTREND	Linear least-squares regression value over aggregated sequence with chunk size of 50	1	$9.10 * 10^{-4}$
FFTCOE	First and third Fourier coefficients of discrete Fourier Transform	2	$3.55 * 10^{-6}$
SYM	Symmetric shape of distribution with the level of 0.1	1	$9.94 * 10^{-4}$

constitute the inputs for location classification task in our experiment.

E. Modeling

In the core of MISSILE, we want to identify sensor patterns for different sensitive indoor locations, which is a typical classification task. There are a number of classification models suitable for this task, such as naive Bayesian, decision tree, support vector machine and neural network. All of them are capable of learning hidden pattern from data to labels. In MISSILE, we choose the non-parametric decision tree as the classifier. In particular, we use the CART (Classification And Regression Tree) classifier model [17]. This model recursively splits input attributes (i.e., features in training data) to generate a binary decision tree, where each leaf node corresponds to a class label. To split attributes, $gini$ index is employed for the impurity measurement function $H(\cdot)$:

$$H(P) = 1 - \sum_k p_k^2,$$

where p_k is the ratio of instances with label k among all the instances in node P. If $H(P) = 0$, then this node becomes a leaf node as only one label exists among all instances. We determine the order of attributes to split using $gini$ gain:

$$Gain(P) = H(P) - \sum_c \frac{|P_c|}{|P|} H(P_c),$$

where P_c represents child node c of node P, and $|\cdot|$ means the number of instances. A higher gain value indicates a better choice to split this node.

F. Calibration for Anomalies

In the above classification, we trust the training set with their labels and input features. However, in reality there are anomalies in the training set. First, exemplars from the automatic collection may be labeled with incorrect location due to BLE signal delay or signal penetration from the floor or wall. Second, malfunctioned mobile sensors may produce low-quality data, which significantly contaminates the training set. To prevent the above anomalies from degrading the classifying accuracy, we adopt two advanced and orthogonal machine learning techniques, namely Ensemble Learning (e.g., Random Forest [18]) and Isolation Forest [19] to identify these anomalies.

1) Random Forest: Ensemble learning uses multiple learning algorithms with bootstrapping technique to achieve better classification accuracy than could be achieved from any of the constituent learning algorithms alone. Recent side-channel attack research [20] [7] suggests that an ensemble version of the decision tree, namely the random forest, is suitable to conduct learning tasks on a noisy dataset with distinguishing patterns. Random forest generates a multitude of decision trees, and trains each with random subset data of the given features. When classifying input features, the data pass through every tree in the forest and the final prediction is decided by the most predicted label of these trees. The various trees trained with different subset data provide significant variability for a prediction model, thus reducing the overfitting issue. However, an ensemble classifier is at the cost of consuming more CPU time for training and classification.

2) Isolation Forest: Isolation forest is a robust and efficient anomaly detection algorithm with linear time complexity. It can be used before the training phase to filter anomalies. The core idea of isolation forest is that anomalies are sensitive to isolation when separating attributes. In other words, an anomaly is usually far away from the dense distribution inside the class cluster, so it is singled out at the early stage of isolation. Essentially an isolation tree (i-Tree) is a full binary tree with random attribute split. It first randomly picks a feature from an input feature set and selects a random splitting point between the minimum and the maximum value of this attribute. All the instances will be separated into two partitions based on this splitting point, one assigned to the left child node and the other to the right child node. The isolation is then recursively conducted until all instances are isolated in the leaf nodes.

Isolation forest constitutes multiple i-Trees obtained by isolating different subsets of the original set of n instances. An anomaly score is then given to each instance x as follows

$$Score(x, n) = 2^{-\frac{E(h(x))}{A(n)}},$$

where the nominator is the expectation of the path length $h(x)$ (i.e., number of edges from the root node to the x instance) of

all i-Trees inside the forest. And the denominator $\mathcal{A}(n)$ is the average path length of an i-Tree given n samples (i.e., average number of edges from the root node to any external leaf node) to normalize $E(h(x))$, which can be derived by

$$\mathcal{A}(n) = 2H(n-1) - \frac{2(n-1)}{n},$$

where $H(n-1)$ is the $(n-1)$-th harmonic number.

Once the scores are ready, we filter those instances whose anomaly scores are higher than our designated threshold. The final score is ranged between $(0, 1]$ since the fractional component in $Score(\cdot)$ is greater than zero and it is bound by an exponential function. An exemplar is considered as an anomaly when its score is close to 1 (i.e., $E(h(x))$ is much smaller than average path length) and a normal one when it is close to 0 (i.e., $E(h(x))$ is much greater than average path length).

G. Attacking Stage

Unlike training stage, the attacking stage, i.e., location eavesdropping, is operated on the victim's device. As such, the key challenge in this stage is to operate in a stealthy manner, i.e., using as low footprint of CPU, memory, bandwidth and power as possible. Regarding low CPU and memory footprint, we employ *sliding window* [21] to process the sensory data stream by limiting the extent of data to a sequence of most recent samples. It is usually defined by tuple $\{win, str\}$ where win is the range of windowing and str is the stride when sliding. When it is applied to the attacking stage, the sensor data are sliced by the sliding window to form a specific length of exemplar and fed to the embedded classifier. Once a sensitive location is inferred, the spyware can take various actions such as notifying its command-and-control center or starting audio recording (if corresponding permission has been granted).

Regarding low power footprint, we propose two optional techniques to reduce the activity of the spyware. The main idea is to invoke the location eavesdropping only when the victim is walking and is not far away from a sensitive location.

Opportunistic Wi-Fi Activation: Nowadays many buildings or common areas are covered by a large public Wi-Fi. By scanning the available SSIDs, the spyware or repackage application can activate eavesdropping only when a victim device "sees" a specific SSID, which means it is close to the premises concerned. Note that scanning nearby SSIDs may require Internet-related permission in the recent release of operating system[7], but most users tend to grant it because it is the most common permission.

Motion Activation: To reduce unnecessary eavesdropping activity when victim is non-moving (standing or sitting still), the spyware can start monitoring only after a motion is detected on the victim through endpointing. Endpointing is a common technique used in speech recognition system to determine the start and end of a user speech and to separate speech region and non-speech region [22]. We can apply endpointing in motion sensor data as the energy in movement region, i.e., the sum of squared sample values, is typically much higher than the energy in non-movement region. We can allow sensors to deep sleep and wake up intermittently to see if the current average window of energy exceeds a predefined threshold. If it does not exceed, all sensors continue to sleep until the next wake-up cycle.

[7]Android does not restrict on scanning SSID until Oreo (8.0). Even in Oreo, SSID scanning is still allowed if an app has any of the three permissions (*CHANGE_WIFI_STATE, ACCESS_FINE_LOCATION, ACCESS_COARSE_LOCATION*).

Fig. 6. Indoor sensitive location examples

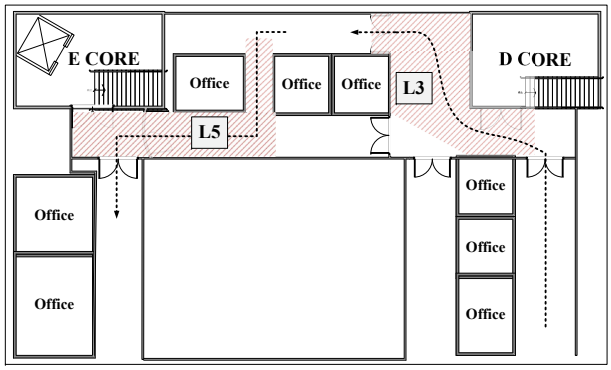
IV. Performance Evaluation

To evaluate the real-life performance of the MISSILE system, we conduct experiments on sensitive locations in a university campus, including students' laboratory, professor's office, common room, washrooms, ATM station, and canteen entrances. The disclosure of these locations can lead to significant privacy breach where, for example, the frequency of accessing washrooms and ATM can indicate personal health and financial status. In practice, entrances, exits and corridors connecting different zones are good targets of sensitive locations as they can be used to outline a victim's daily activity. Such knowledge can further lead to social engineering attacks. As for the selection of locations, we first identify sensitive indoor areas that imply strong semantics of personal activities and may arouse interests of attackers. Then for each chosen location we represent it (and its neighborhood) by a combination of visual characteristics (e.g., door, turn, corridors) as listed in Table. III. In our experiment, we choose 15 representative sensitive locations that exhibit different combinations of visual characteristics, which constitute the unique patterns when victims pass by. Fig. 6 shows photo snapshots of four sample locations whereas Fig. 7(a) and Fig. 7(b) plot them in their corresponding floor plan.

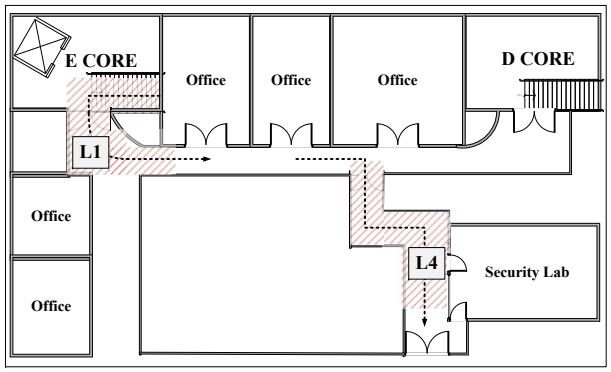
The sensory data are collected by 10 individuals with mixed genders and body figures. They carry the test devices with random placement (left/right/front pockets) for their daily use over a period of 90 days. As for location labels, we take advantage of existing BLE beacons deployed by other services (e.g., teaching facilitation) to label sensor data and each location has one beacon as reference point. To preclude the impact of the way we split training and test datasets, all experiments are conducted 10 times using random splits and the averaged results are reported. Specifically, among all 2580 exemplars

TABLE III
VISUAL CHARACTERISTICS OF SENSITIVE LOCATIONS

Vital Characteristics	Location ID														
	$L1$	$L2$	$L3$	$L4$	$L5$	$L6$	$L7$	$L8$	$L9$	$L10$	$L11$	$L12$	$L13$	$L14$	$L15$
Single Fire Stop Door		✓	✓	✓	✓						✓		✓	✓	✓
Double Fire Stop Door	✓												✓		✓
Corridor Quarter Turn		✓		✓	✓			✓				✓			
Winding Corridor	✓		✓			✓			✓		✓				
Straight Corridor						✓								✓	
Elevator	✓						✓	✓							
Stairways	✓	✓					✓			✓					



(a) Floor plan for $L3$ (falculty office) and $L5$ (inventory office)



(b) Floor plan for $L1$ (research laboratory) and $L4$ (teaching laboratory)

Fig. 7. Experiment floor plan with example trajectory (shadow area illustrates sensitive area).

of sensitive locations, 6 individuals' exemplars (around 1548 exemplars) are used for training while the other 4 individuals' (around 1032 exemplars) are used for testing. As for non-sensitive locations, we randomly extract 350 exemplars for training into a "non-sensitive location" class, which is close to the number of exemplars of the most popular sensitive location. For testing, we extract another 1400 exemplars with non-sensitive locations. This ratio, $1032 : 1400$, approximates the statistics of ratios of sensitive to non-sensitive locations of all exemplars in our experiment. Other system parameters are listed in Table IV.

Our comparative study includes: (1) the performance of different normalization approaches (i.e., rotation-matrix-based versus Euclidean-norm-based normalization), (2) the impact of ensemble classifiers (i.e., decision tree versus random forest),

TABLE IV
SYSTEM PARAMETERS

Parameter	**Value**
Exemplar Length	15s
Sampling Frequency	50Hz
Noise Filter α	0.15
Step Threshold	750
RSSI Threshold	-85dB
Features	12 features per sensor axis
Anomaly Threshold	0.8
Models	Decision Tree, Random Forest
Devices	(Android Version, RAM, CPU) LG G3 (5.0, 3GB, 2.5GHz), Redmi Note4X (6.0, 4GB, 2.0GHz), Google Pixel (7.1, 4GB, 2.15GHz), HTC U Ultra (8.0, 4GB, 2.15GHz), Samsung Galaxy S8 (8.0, 4GB, 2.35GHz)
Selected Sensors	Gyroscope (3-axis), Magnetic Field Sensor (3-axis), Linear Acceleration Sensor (3-axis), Accelerometer (3-axis)

(3) the impact of isolation forest, (4) the impact of training data size, (5) the impact of sensor types, (6) the impact of different location characteristics, (7) the impact of system parameters (e.g. exemplar length, tree numbers in random forest and feature setting), and (8) the power consumption on popular devices. To evaluate the effectiveness of MISSILE attack, we categorize all classification results of location label i into 4 cases in one-vs-all style: true positive (TP_i, recognizing a location i correctly), true negative (TN_i, ignoring a location i correctly), false positive (FP_i, recognizing a location i incorrectly), and false negative (FN_i, ignoring a location i incorrectly). Based on these cases, we define $precision$ and $recall$ for each location label i as follows

$$precision_i = \frac{|TP_i|}{|TP_i| + |FP_i|}$$

$$recall_i = \frac{|TP_i|}{|TP_i| + |FN_i|}$$

The $precision_i$ essentially tells how well the system can distinguish location i from other locations while $recall_i$ shows how well the system can detect a particular location label. As these two metrics are sometimes contradicting to each other,

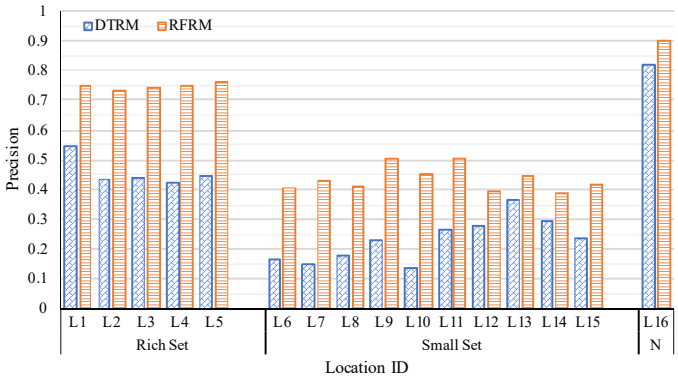


Fig. 8. Impact of training data size by decision tree and random forest

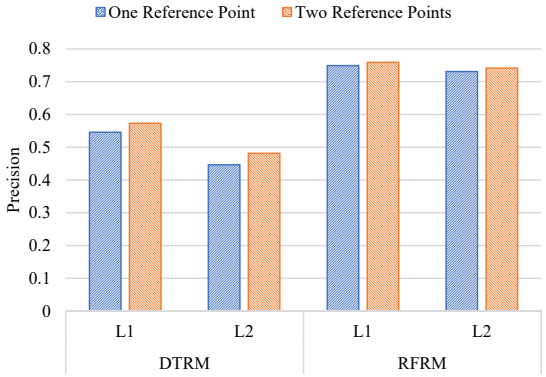


Fig. 9. System precision with one and two reference points calibration

we also employ the F1-score [23] as an overall metric for each location label i, which is

$$F1_i = \frac{2 \cdot precision_i \cdot recall_i}{recall_i + precision_i}$$

The overall F1-score is the weighted average F1-score of all location labels, based on the number of true instances of each location in the testing data label set L as follows

$$F1_{overall} = \sum_i \frac{|L_i|}{|L|} F1_i$$

A. Overall Performance of Missile System

Table. V shows the performance comparison between decision tree (DT) and random forest (RF), with Euclidean-norm-based (EN) and rotation-matrix-based normalization (RM), namely, DTEN, DTRM, RFEN, RFRM. We observe that all classifiers significantly outperform random guess (one out of 16 choices, 6.25%), which justifies the feasibility of MISSILE. Further, random forest, an ensemble classifier, can achieve an even higher F1-score of 62%. On the other hand, rotation matrix normalization always outperforms Euclidean norm by at least 10%, because it preserves useful information for classification. Anomaly detection by isolation tree has shown moderate improvement of F1-score for all classifiers, among which the classifier with decision tree and Euclidean norm witnesses over 6% improvement. This shows both the ensemble method and rotation matrix normalization are more robust against anomalies. For the rest of experiments, we will mainly report DTRM and RFRM results after isolation tree. In terms of CPU time, classifiers using ensemble method cost significantly 10 times more CPU resources to train the model and 50 times more to make a prediction. Rotation matrix normalization also noticeably increases the training overhead but has less influence for prediction. It suggests that spyware can switch among different classifiers to balance battery condition and desired utility.

B. Impact of Training Data Size

Fig. 8 illustrates the precision for individual sensitive locations. We categorize them into locations with small training data ($\leq$ 50 exemplars) and locations with rich training data ($\geq$ 150 exemplars). We observe that locations in the former category ($L6$ to $L15$) have a higher probability to be misclassified. This effect is more eminent for the decision tree than for the random forest, as the former is a single classifier method and thus more vulnerable to noises and outliers. An ensemble method such as the random forest tends to alleviate the impact of noises and outliers by splitting data into subsets with crossover items, so that they cannot easily dominate the training process. Note that $L16$ (grouped as N) is a location label for all non-sensitive locations. It achieves around 90% accuracy in $RFRM$, which indicates that the system is able to identify most of non-sensitive locations.

C. Impact of BLE Reference Points

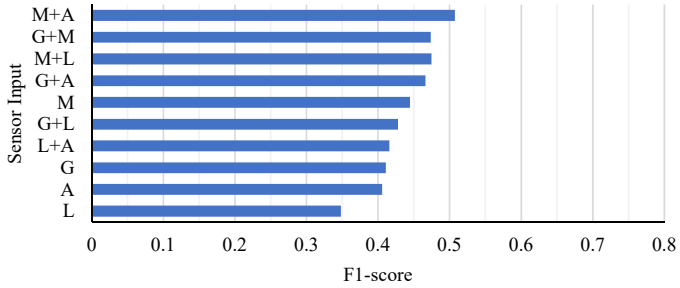
To evaluate the impact of the number of BLE reference points on the annotation of starting timestamps, we collect additional training data from location L1 and L2 using two beacons each and re-train our system. Fig. 9 plots the difference of system precision using one and two reference points. Overall, the system with two reference points always performs better by 3%-4% for DTRM and 1% for RFRM. The small gain might be attributed to the long exemplar length, which is already long enough to include enough sensor patterns to distinguish a sensitive location (more details are discussed in Section IV-F. Since one reference point already leads to satisfactory performance, throughout the experiment we use one reference point for each location and calibrate the starting timestamps of exemplars with the mean of deviation distribution.

D. Impact of Sensor Type

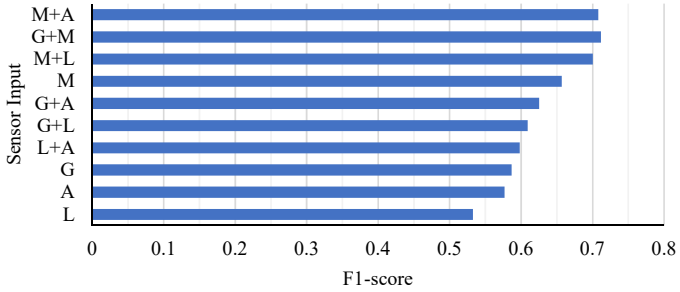
To investigate the contributions of different sensor types in the MISSILE system, we measure the F1-scores using single or a pair of sensors in Fig. 10. We observe that in both DTRM and RFRM, the top F1-score rankings are similar, which means some sensor or sensor combinations are consistently better than the others regardless of the classifiers. In particular, the magnetic field sensor plays a major role, with its F1-score reaching over 60% (alone) and around 70% (pairwise). This indicates that the magnetic distribution caused by geolocation and indoor structure material can constitute a unique signature for inferring sensitive locations. By combining another motion sensor, such a sensor pair can approximate the result of using

TABLE V
OVERALL PERFORMANCE OF MISSILE SYSTEM

Classifier	Training Time (s)	Inference Time (ms)	F1-Score	F1-Score (after Isolation Forest)
Decision Tree + Euclidean Norm (DTEN)	0.6 - 0.8	0.8 - 1.2	35.14%	42.35%
Decision Tree + Rotation Matrix (DTRM)	1.6 - 2.0	1.5 - 2.0	49.27%	53.13%
Random Forest + Euclidean Norm (RFEN)	9.0 - 10.0	40.0 - 60.0	59.62%	63.14%
Random Forest + Rotation Matrix (RFRM)	15.0 - 16.0	60.0 - 80.0	70.81%	73.79%



(a) Sensor evaluation with DTRM



(b) Sensor evaluation with RFRM

Fig. 10. Performance results in descending order for different sensors: Gyroscope (G), Magnetic Field Sensor (M), Linear Acceleration Sensor (L), Accelerometer (A)

all four sensors. Other three sensors leverage user behavior information and obtain similar results (over 50%).

E. Impact of Location Characteristics

Regardless of the methods used for classification, we observe that the F1-scores in some locations are consistently better than those in the others. For example, locations $L1$ and $L2$ have both high precision (75% and 73%) and high recall (90% and 86% in Fig. 11). From Table III, we learn that $L1$ and $L2$ have 4 and 3 characteristics, respectively, while all other locations have 2 or even fewer. Furthermore, some characteristic has more significant impact than the others. For example, the top-ranked recall locations — $L1$-$L5$, $L13$, $L14$, $L11$, and $L15$ — all share a common characteristic: a fire stop door. Such high recall implies a high tendency to identify locations with door opening event correctly.

F. Impact of System Parameters

1) Impact of Exemplar Length: In previous experiments, we set 15 seconds as the standard time length of an exemplar in the attacking stage. This value is set to generate a sufficiently long signal pattern that captures necessary characteristics of any sensitive location. In this subsection, we vary this length from 3 to 30 seconds and plot the F1-score in Fig. 12(a) under both decision tree and random forest methods.

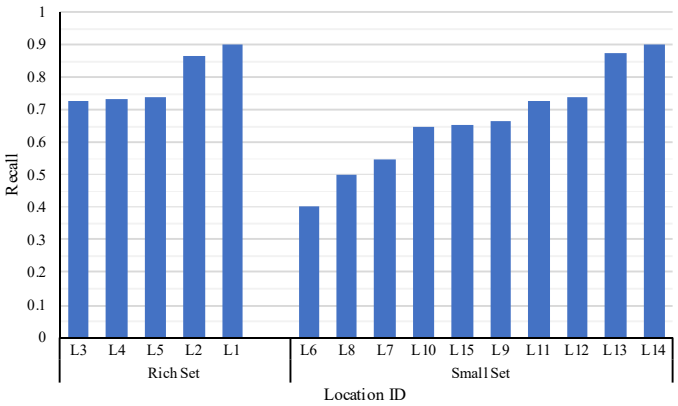


Fig. 11. Recall of RFRM in ascending order for individual location

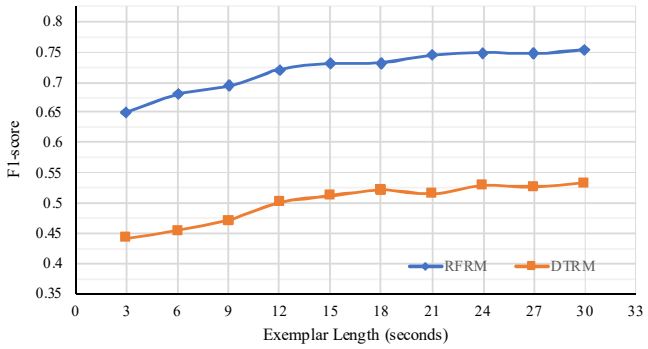
For both methods, we observe a steady increase as the exemplar length increases, which coincides with our reasoning above that a longer exemplar may capture more characteristics of a sensitive location. However, the F1-score starts to saturate after 12 seconds especially for classifier $RFRM$, which indicates that over-extending this length does not significantly help to further improve the classification results as the chance of a sensitive location being covered by two consecutive exemplars is slim.

2) Impact of Random Forest Setting: We vary the number of decision trees for the ensemble method (i.e., the random forest) and plot the F1-score for both Euclidean norm and rotation matrix normalization in Fig. 12(b). We observe that both methods reach a saturation point after 60-100, which means the random forest is robust under this parameter.

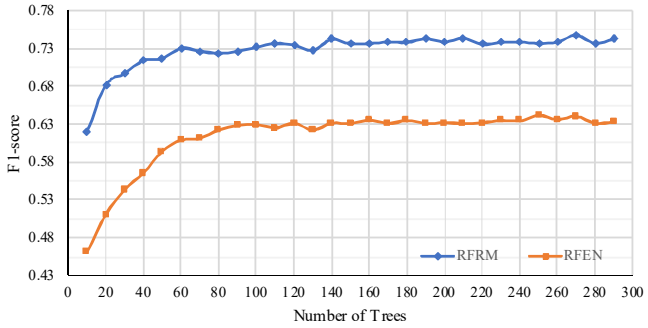
3) Impact of Feature Setting: Top-12 feature selection is adopted during the automatic feature extraction in all the previous experiments. To examine the impact of this setting, we measure and plot the F1-score change of classifier $RFRM$ with feature sets generated under different top-r settings in Fig. 12(c). We observe that top-1 feature in classifier $RFRM$ can reach an F1-score of 48% alone. The performance of classifier grows steadily until this setting reaches top-6 and F1-score saturates at around 73%. This indicates that a minimum of top-6 feature setting is required for classifier $RFRM$ to achieve its best performance. Since the ranking is decided by p-values shown in Table. II, it is obvious that features containing the unique information of sensitive location are highly associated with low p-values.

G. Power Consumption

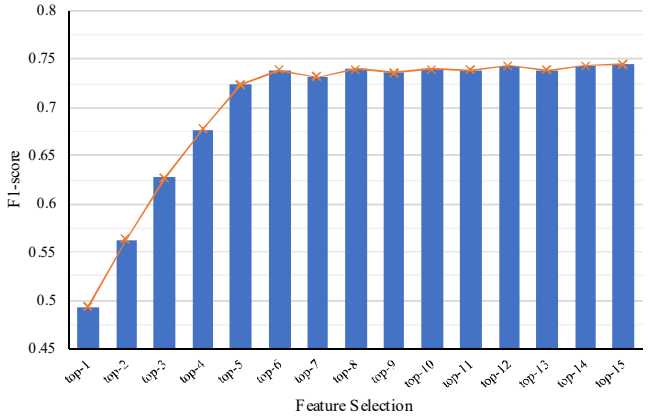
The spyware installed by MISSILE continually samples multiple mobile sensors. To reduce power consumption, we implement both opportunistic WiFi activation and motion ac-



(a) System performance under different exemplar lengths



(b) System performance under different number of trees in random forest



(c) System performance of RFRM under different top-r settings

Fig. 12. Impact of system parameters over performance

TABLE VI
SPYWARE POWER CONSUMPTION ON MONITORING SENSORS (50HZ)

Model	Capacity	Spyware-On	Idle	Usage
HTC U Ultra	3000mAh	1.323%	0.611%	0.712%
Samsung Galaxy S8	3000mAh	1.529%	0.801%	0.728%
Google Pixel	2770mAh	1.317%	0.507%	0.810%

tivation as in Section III. To further evaluate the power impact of continuously accessing sensors, we activate the spyware in the background to sample sensors with the screen off and measure the power usage per hour of various smartphones. The result is presented in Table. VI, which shows a moderate consumption of around 0.7% - 0.8% extra battery per hour.

V. EXTENSION AND COUNTERMEASURE

In this section, we will discuss the potential extension of MISSILE and countermeasures.

In the experiment, the sensitive locations data are collected by attackers manually, which limits the scalability of this attack. To acquire a large-scale and more diversified dataset, this process can be enhanced by automation or crowdsensing. The former, such as IndoorAtlas [24], can provide efficient sensory measurement of indoor location. The latter delegates the task of sensing and labeling location data to a crowdsourcing platform.

Currently, the MISSILE system only considers stateless recognition, which does not take the relationship between sensitive locations into consideration. Inspired by dead reckoning for indoor positioning [25], we can improve the location inference performance by extracting detailed context such as walking distance, turning angle and pushing motion.

As for countermeasures for indoor sensitive location inference attack of MISSILE, we propose two methods below.

Access Control: Permission mechanism is the first line of defense. We suggest that no request from mobile application for statistics or raw sensory information should bypass the permission mechanism. In addition, since high-resolution sensor data can be exploited by attackers who take advantage of subtle change [26], we suggest replacing them with feature-level data access, which also significantly reduces computational cost.

Data Manipulation: Noise injection is an alternative countermeasure. Software level noise injection has already been applied to GPS data in the geo-social network. With the same rationale, noise can be injected into sensor readings to avoid highly accurate location inference. If the operating system cannot be trusted to perform this injection, we recommend employing hardware noise injection, for example, enabling the vibrator of a mobile device.

VI. RELATED WORK

In this section, we review the related literature on mobile side-channel attacks using inertial sensors and indoor positioning.

A. Side-channel Attacks on Mobile Devices

Side-channel attacks on mobile devices have evolved drastically. Since smart devices constantly sense the environmental information with their embedded sensors, external influence such as temperature, air pressure, noise, and body movement may create unique patterns on sensory data. For motion-related sensors, Owusu *et al*. [20] have eavesdropped users' passwords using only accelerometer to detect the acceleration caused by different digits. Mehrnezhad *et al*. [27] further improve this approach by a website that can smuggle sensor readings with JavaScript. For radio frequency related sensors, Li *et al*. [28] propose WindTalker to infer sensitive keystrokes on mobile devices with side-channel information from wireless network. As for other environmental sensing modules (e.g. barometer, magnetometer, microphone, and camera), similar approach has been applied to the inference of identity [29] and behavior [30].

Recently side-channel attacks have exploited the inertial sensors in Android and iOS to infer a user's outdoor location and even trace him/her. While we pay close attention to the

indoor scenario with pedestrian where the estimation has to deal with limited data and volatile movement, most of related works focus on outdoor inference where multiple resources are available (e.g. GPS, street map and real-time public transport database). Users' driving routes have been successfully tracked in [31] and [6] by motion sensory information from their mobile devices. The former work is based on accelerometer and gyroscope, which leverages a dead reckoning technique with probability mapping algorithm while the latter one uses fine-grained gyroscope data and the graph of road information. Other than motion sensors, ambient sensors have also been investigated in driving route inference attack. Won *et al*. [12] have proposed to use the latent relation between barometer readings and the geolocation to track drivers. As for public transportation, Hua *et al*. [7] have shown that they can reveal users' daily metro schedule by monitoring accelerometers whose data are significantly affected by the route of metro. Watanabe *et al*. [32] have demonstrated how to infer users' train schedule by matching user motions with the public railway database.

Other location attacks leverage non-sensory or active information. Mosenia *et al*. [33] can track users on train, plane or outdoor walking using hybrid sources. Michalevsky *et al*. [4] have designed a location inference attack by profiling power consumption during commutes as cellular signal strength varies. Gao *et al*. [34] have shown that usage-based automotive insurance can expose a driver's route through the recorded driving speed while Zhou *et al*. [35] enhanced the inference performance using real-time traffic and proposed defense framework with privacy-preserving scoring and audition. Kenneth *et al*. [36] develop an active location attack using signal transmitted from low power magnetic coil and received by mobile magnetometer. Cellular network based localization has also shown to be effective by observing the pattern of data transmission [5] or listening to GSM broadcast channel [37]. Li *et al.* [38] and Ometov *et al*. [39] have proposed new methods on location tracking through social network footprints. Arp *et al*. [40] have used the ultrasonic wave to infer a user's current location.

B. Indoor Positioning

Mobile devices have played a major role in indoor positioning system (IPS) over the past decade. Significant approaches in IPS include dead reckoning, magnetic field fingerprinting, wireless multilateration and fingerprinting.

Among these approaches, dead reckoning extensively utilizes inertial motion sensors for displacement and activity analysis. It predicts user's indoor route from an initial position with continuously measured speed and direction. Murata *et al.* [41] improve the performance of basic indoor dead reckoning with human activity knowledge such as age and environment when estimating step length. While Kang *et al.* [25] develop a fine-grained system to derive accurate walking parameters from inertial sensor data. But in dead reckoning, small error may easily accumulate and lead to erroneous results [42]. An alternative indoor positioning technique using inertial sensor is to leverage magnetic filed information. Magnetic field fingerprinting requires an offline mapping of magnetic intensity [43]. Wireless multilateration is another more stable positioning solution which derives time of arrival or direction of arrival information from external reference devices to pinpoint current location [42] while wireless fingerprinting captures the distribution of wireless signal strength [44].

Modern State-of-Art IPS usually incorporates various radio frequency signals (WLAN, RFID, Bluetooth, etc.) with context information from the floor plan, ambient sound/light sensing, magnetic field map to provide reliable positioning [45]. Under this legitimate scenario, unconstrained permission is granted to access privileged radio sensors, pedestrian initial state, computation power and floor plan data. However, such resources are unavailable for a stealthy attacker, who also has little domain knowledge of mobility analysis or magnetic map construction.

VII. Conclusion

In this paper, we investigate a side-channel attack that can eavesdrop user's sensitive locations using unprivileged sensory information. This attack is modeled as a classification problem of various sensory data collected from different locations. The classifier is built from supervised learning of training data prepared by automatic labeling mechanism, effective processing and optimal feature extraction. Real experiments are conducted on 15 indoor locations inside a university campus. The classifier using modeling with anomaly calibration can reach around 73% F1-score, which is significantly higher than random guess. As for future work, we plan to implement the improved version using multiple reference points for labeling, stateful routes and other side-channels (e.g., JavaScript in mobile browser). We also plan to investigate countermeasures against such attack, evaluate and compare them on various metrics, such as time complexity, accuracy and utility.

Acknowledgment

This work was supported by National Natural Science Foundation of China (Grant No: 61572413, U1636205), the Research Grants Council, Hong Kong SAR, China (Grant No: 15238116, 15222118 and C1008-16G), and a research grant from Huawei Technologies.

References

[1] "Privacy & data security update," Accessed: Nov. 16, 2018. [Online]. Available: http://www.ftc.gov/reports/

[2] "Data protection in the eu," Accessed: Nov. 19, 2018. [Online]. Available: https://ec.europa.eu/info/law/law-topic/data-protection/data-protection-eu

[3] Q. A. Chen, Z. Qian, and Z. M. Mao, "Peeking into your app without actually seeing it: UI state inference and novel android attacks," in *Proc. USENIX Secur. Symp.*, 2014, pp. 1037–1052.

[4] Y. Michalevsky, A. Schulman, G. A. Veerapandian, D. Boneh, and G. Nakibly, "Powerspy: Location tracking using mobile device power analysis," in *Proc. USENIX Secur. Symp.*, 2015, pp. 785–800.

[5] H. Soroush, K. Sung, E. G. Learned-Miller, B. N. Levine, and M. Liberatore, "Turning off GPS is not enough: Cellular location leaks over the internet," in *Proc. Priva. Enhancing Technol. Symp. (PETs)*, 2013, pp. 103–122.

[6] S. Narain, T. D. Vo-Huu, K. Block, and G. Noubir, "Inferring user routes and locations using zero-permission mobile sensors," in *IEEE Symp. Secur. and Priva. (S&P)*, 2016, pp. 397–413.

[7] J. Hua, Z. Shen, and S. Zhong, "We can track you if you take the metro: Tracking metro riders using accelerometers on smartphones," *IEEE Trans. Inf. Forensics Secur.*, vol. 12, no. 2, pp. 286–297, 2017.

[8] M. Mehrnezhad, E. Toreini, S. F. Shahandashti, and F. Hao, "Touchsignatures: Identification of user touch actions based on mobile sensors via javascript," in *Proc. ACM ASIA Conf. Comput. Commun. Secur. (AsiaCCS)*, 2015, p. 673.

[9] L. Grustniy, "What's wrong with 'legal' commercial spyware." 2018. [Online]. Available: https://www.kaspersky.com/blog/stalkerware-spouseware/26292/

[10] K. O'Flaherty, "Whatsapp users targeted by spyware – here's what you need to know." 2019. [Online]. Available: https://www.forbes.com/sites/kateoflahertyuk/2019/05/14/whatsapp-users-targeted-with-israeli-spyware-heres-what-you-need-to-know/

[11] B. Li, T. Gallagher, A. G. Dempster, and C. Rizos, "How feasible is the use of magnetic field alone for indoor positioning?" in *Int. Conf. Indoor Positioning and Indoor Navigation (IPIN)*, 2012, pp. 1–9.

[12] M. Won, A. Mishra, and S. H. Son, "Hybridbaro: Mining driving routes using barometer sensor of smartphone," *IEEE Sensors Journal*, vol. 17, no. 19, pp. 6397–6408, 2017.

[13] K. Cho, W. Park, M. Hong, G. Park, W. Cho, J. Seo, and K. Han, "Analysis of latency performance of bluetooth low energy (BLE) networks," *Sensors*, vol. 15, no. 1, pp. 59–78, 2015. [Online]. Available: https://doi.org/10.3390/s150100059

[14] S. S. Wilks, "The large-sample distribution of the likelihood ratio for testing composite hypotheses," *The Annals of Mathematical Statistics*, vol. 9, no. 1, pp. 60–62, 1938.

[15] J. Goslinski, M. Nowicki, and P. Skrzypczynski, "Performance comparison of ekf-based algorithms for orientation estimation on android platform," *IEEE Sensors Journal*, vol. 15, no. 7, pp. 3781–3792, 2015.

[16] M. Christ, A. W. Kempa-Liehr, and M. Feindt, "Distributed and parallel time series feature extraction for industrial big data applications," *CoRR*, vol. abs/1610.07717, 2016.

[17] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth, 1984.

[18] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[19] K. M. Ting, F. T. Liu, and Z. Zhou, "Isolation forest," in *IEEE Int. Conf. Data Mining (ICDM)*, vol. 00, 2008, pp. 413–422.

[20] E. Owusu, J. Han, S. Das, A. Perrig, and J. Zhang, "Accessory: password inference using accelerometers on smartphones," in *ACM Workshop on Mobile Comput. Syst. and Appl. (HotMobile)*, 2012, pp. 9:1–9:6.

[21] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An online algorithm for segmenting time series," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, 2001, pp. 289–296.

[22] J.-l. Shen, J.-w. Hung, and L.-s. Lee, "Robust entropy-based endpoint detection for speech recognition in noisy environments," in *Proc. Int. Conf. on Spoken Language Processing*, 1998.

[23] N. Chinchor, "MUC-4 evaluation metrics," in *Proc. Conf. Message Understanding (MUC)*, 1992, pp. 22–29.

[24] "Indooratlas," Accessed: Nov 20, 2018. [Online]. Available: http://www.indooratlas.com/

[25] W. Kang and Y. Han, "Smartpdr: Smartphone-based pedestrian dead reckoning for indoor localization," *IEEE Sensors Journal*, vol. 15, no. 5, pp. 2906–2916, 2015.

[26] Y. Michalevsky, D. Boneh, and G. Nakibly, "Gyrophone: Recognizing speech from gyroscope signals," in *Proc. USENIX Secur. Symp.*, 2014, pp. 1053–1067.

[27] M. Mehrnezhad, E. Toreini, S. F. Shahandashti, and F. Hao, "Touchsignatures: Identification of user touch actions and pins based on mobile sensor data via javascript," *Journal of Info. Secur. and Appl.*, vol. 26, pp. 23 – 38, 2016.

[28] M. Li, Y. Meng, J. Liu, H. Zhu, X. Liang, Y. Liu, and N. Ruan, "When csi meets public wifi: Inferring your mobile phone password via wifi signals," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2016, pp. 1068–1079.

[29] X. Zhou, S. Demetriou, D. He, M. Naveed, X. Pan, X. Wang, C. A. Gunter, and K. Nahrstedt, "Identity, location, disease and more: inferring your secrets from android public resources," in *Proc. ACM Conf. Comput. Commun. Secur. (CCS)*, 2013, pp. 1017–1028.

[30] S. Hemminki, P. Nurmi, and S. Tarkoma, "Accelerometer-based transportation mode detection on smartphones," in *Proc. ACM Conf. Embedded Netw. Sens. Syst. (SenSys)*, 2013, pp. 13:1–13:14.

[31] J. Han, E. Owusu, L. T. Nguyen, A. Perrig, and J. Zhang, "Accomplice: Location inference using accelerometers on smartphones," in *Int. Conf. Commun. Syst. Netw. (COMSNETS)*, 2012, pp. 1–9.

[32] T. Watanabe, M. Akiyama, and T. Mori, "Routedetector: Sensor-based positioning system that exploits spatio-temporal regularity of human mobility," in *USENIX Workshop on Offensive Technol. (WOOT)*, 2015.

[33] A. Mosenia, X. Dai, P. Mittal, and N. K. Jha, "Pinme: Tracking a smartphone user around the world," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 4, no. 3, pp. 420–435, 2018.

[34] X. Gao, B. Firner, S. Sugrim, V. Kaiser-Pendergrast, Y. Yang, and J. Lindqvist, "Elastic pathing: Your speed is enough to track you," in *Proc. Int. Joint Conf. Pervas. and Ubiquitous Comput. (UbiComp)*, 2014, pp. 975–986.

[35] L. Zhou, S. Du, H. Zhu, C. Chen, K. Ota, and M. Dong, "Location privacy in usage-based automotive insurance: Attacks and countermeasures," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 1, pp. 196–211, Jan 2019.

[36] K. Block and G. Noubir, "My magnetometer is telling you where i've been?: A mobile device permissionless location attack," in *Proc. ACM Conf. Secur. Priva. Wireless Mobile Netw. (WiSec)*, 2018, pp. 260–270.

[37] D. F. Kune, J. Kölndorfer, N. Hopper, and Y. Kim, "Location leaks over the GSM air interface," in *Proc. Annu. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, 2012.

[38] H. Li, H. Zhu, S. Du, X. Liang, and X. . Shen, "Privacy leakage of location sharing in mobile social networks: Attacks and defense," *IEEE Trans. Dependable and Security Comput.*, vol. 15, no. 4, pp. 646–660, 2018.

[39] A. Ometov, A. Levina, P. Borisenko, R. Mostovoy, A. Orsino, and S. Andreev, "Mobile social networking under side-channel attacks: Practical security challenges," *IEEE Access*, vol. 5, pp. 2591–2601, 2017.

[40] D. Arp, E. Quiring, C. Wressnegger, and K. Rieck, "Privacy threats through ultrasonic side channels on mobile devices," in *IEEE European Symp. Secur. Priva. (EuroS&P)*, 2017, pp. 35–47.

[41] Y. Murata, K. Kaji, K. Hiroi, and N. Kawaguchi, "Pedestrian dead reckoning based on human activity sensing knowledge," in *Proc. Int. Joint Conf. Pervas. and Ubiquitous Comput. (UbiComp)*, 2014, pp. 797–806.

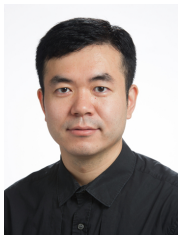
[42] R. F. Brena, J. García-Vázquez, C. E. Galván-Tejada, D. M. Rodríguez, C. V. Rosales, and J. F. Jr., "Evolution of indoor positioning technologies: A survey," *Journal of Sensors*, vol. 2017, pp. 2 630 413:1–2 630 413:21, 2017.

[43] B. Gozick, K. P. Subbu, R. Dantu, and T. Maeshiro, "Magnetic maps for indoor navigation," *IEEE Trans. on Instrumentation and Measurement*, vol. 60, no. 12, pp. 3883–3891, 2011.

[44] A. Khalajmehrabadi, N. Gatsis, and D. Akopian, "Modern wlan fingerprinting indoor positioning methods and deployment challenges," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1974–2002, thirdquarter 2017.

[45] P. Davidson and R. Piché, "A survey of selected indoor positioning methods for smartphones," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 2, pp. 1347–1370, 2017.

Huadi Zheng receives the BEng degree from the School of Data and Computer Science, Sun Yat-sen University, China, in 2012. Currently he is pursuing a PhD degree in the Department of Electronic and Information Engineering, Hong Kong Polytechnic University. His research interests include mobile side-channel security, data privacy and machine learning.



Haibo Hu is an associate professor in the Department of Electronic and Information Engineering, Hong Kong Polytechnic University. His research interests include cybersecurity, data privacy, internet of things, and machine learning. He has published over 70 research papers in refereed journals, international conferences, and book chapters. As principal investigator, he has received over 10 million HK dollars of external research grants from Hong Kong and mainland China. He is the recipient of a number of titles and awards, including IEEE MDM 2019 Best Paper Award, WAIM Distinguished Young Lecturer, VLDB Distinguished Reviewer, ACM-HK Best PhD Paper, Microsoft Imagine Cup, and GS1 Internet of Things Award.