MAGLeak: A Learning-Based Side-Channel Attack for Password Recognition With Multiple Sensors in IIoT Environment

Dajiang Chen, *Member, IEEE*, Zihao Zhao, Xue Qin, Yaohua Luo, Mingsheng Cao, Hua Xu, and Anfeng Liu

***Abstract*—As an emerging technology, industrial Internet of Things (IIoT) connects massive sensors and actuators to empower industrial sectors being smart, autonomous, efficient, and safety. However, due the large number of build-in sensors of IIoT smart devices, the IIoT systems are vulnerable to side-channel attack. In this article, a novel side-channel-based passwords cracking system, namely MAGLeak, is proposed to recognize the victim's passwords by leveraging accelerometer, gyroscope, and magnetometer of IIoT touch-screen smart device. Specifically, an event-driven data collection method is proposed to ensure that the user's keystroke behavior can be reflected accurately by the obtained measurements of three sensors. Moreover, random forest algorithm is leveraged for the recognition module, followed by a data preprocessing process. Extensive experimental results demonstrate that MAGLeak achieves a high recognition accuracy under small training dataset, e.g., achieving recognition accuracy 98% of each single key for 2000 training samples.**

***Index Terms*—Accelerometer, gyroscope, industrial Internet of Things (IIoT), magnetometer, password cracking, random forest, side-channel attack.**

Manuscript received May 9, 2020; revised July 17, 2020, September 30, 2020, and November 1, 2020; accepted December 1, 2020. Date of publication December 18, 2020; date of current version September 29, 2021. This work was supported in part by NSFC under Grant 61 872 059, Grant 41 930 112, Grant 61 771 417, and Grant 61 502 085 and in part by the project "The Verification Platform of Multi-tier Coverage Communication Network for oceans" under Grant LZC0020. Paper no. TII-20-2342. *(Dajiang Chen and Zihao Zhao contributed equally to this article.) (Corresponding authors: Yaohua Luo; Mingsheng Cao.)*

Dajiang Chen is with the Network and Data Security Key Laboratory of Sichuan Province, UESTC, Chengdu 611 731, China, and also with the Peng Cheng Laboratory, Shenzhen 518 055, China (e-mail: djchen@uestc.edu.cn).

Zihao Zhao and Mingsheng Cao are with the Network and Data Security Key Laboratory of Sichuan Province, University of Electronic Science and Technology of China, Chengdu 611 731, China (e-mail: kevinzhaozh1998@gmail.com; cms@uestc.edu.cn).

Xue Qin is with the Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON N9B 3P4, Canada (e-mail: xqin@islander.tamucc.edu).

Yaohua Luo is with the State Key Laboratory of Geohazard Prevention and Geoenvironment Protection, Shenzhen 518 055, China, and also with the College of Information Science and Technology, Chengdu University of Technology, Chengdu 610 059, China (e-mail: lyh@cdut.edu.cn).

Hua Xu is with the School of Physics and Electronic Engineering, Yancheng Teachers University, Yancheng 224 007, China (e-mail: xuhua@yctu.edu.cn).

Anfeng Liu is with the School of Information Science and Engineering, Central South University, ChangSha 410 083, China (e-mail: afengliu@mail.csu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2020.3045161.

Digital Object Identifier 10.1109/TII.2020.3045161

I. Introduction

As THE third wave of the world's information industry after computers and the internet, industrial Internet of Things (IIoT) can empower many industrial sectors to be smart, autonomous, efficient, and safe by using massive sensors and actuators [1]. As IIoT has been widely used in urban critical infrastructure [2]–[4], such as, water or natural gas networks, smart grids, and transportation systems, it is crucial to protect the security of IIoT [5], [6]. However, due to the complex and heterogeneous topology, and massive nodes with huge differences in capabilities (e.g., computing power, storage capacity, and energy), the industrial networks are vulnerable to many attacks, such as, node impersonation [7], [8], message tampering [9], and information eavesdropping [10].

Among the large number of attacks, one type of attack is extremely harmful: personal identification number (PIN) or passwords cracking. If the PIN or passwords of a legitimate user is cracked, the adversary can impersonate the victim to perform any authorized operation, resulting in serious damages, such as, leakage of industrial secrets, shutdown of a production line, and IIoT system paralysis. One common type of passwords cracking attack is based on side-channel. Actually, the states of smart device build-in sensors (e.g., gyroscope, accelerometer, magnetometer, and microphone) and/or system components (e.g., memory and CPU) are usually probed by adversary to be regard as a side-channel to infer the victimized user's privacy, such as, passwords, gender, and other attributes [11], [12]. The related works can be traced back to Shukal *et al.* [18], in which, a side-channel attack was first proposed to break the PIN with camera. Since then, side-channel-based PIN or passwords cracking on smart devices has been extensively studied, such as, side-channel attack using WiFi signals [15], audio signals [16], [17], and build-in accelerometer of smart devices [21], [22].

Considering that a large number of IIoT systems use touch screen smart devices to achieve human–system interaction, and passwords are commonly used to authenticate users and manage devices, in this article, a side-channel-based passwords cracking system, namely MAGLeak, is designed by using multiple build-in sensors (i.e., accelerometer, gyroscope, and magnetometer) under the IIoT scenario. Specifically, an adversary aims to recognize the passwords of a victim when the victim enters passwords on the numeric soft keyboard of a IIoT touch-screen smart device.

In MAGLeak, an event-driven data collection method is first proposed to obtain the measurements of three sensors when a victim presses the numeric keyboard and lifts the fingers from the numeric keyboard. Note that, the press (resp. release) action can be captured by the *OnPress* (resp. *OnUp*) function of Android system. In this way, the obtained measurements can accurately describe the number typing behavior. Then, a data preprocessing process is presented, which includes a dimensionality reduction, samples inserting, and data regularization. The dimensionality reduction with principal component analysis (PCA) algorithm [32] and data regularization with ridge regularization are leveraged to prevent the overfitting of original training data, as well as facilitate later data training and recognition. The samples inserting with SMOTE algorithm [28] are used to prevent the imbalance of original training data. Finally, we compare the performance of several classic machine learning algorithms, i.e., the random forest [29], support vector machine (SVM) [30], multiple layer perceptron artificial neural network (ANN) [13], and K-means clustering algorithm (K-Means) [14], in training dataset with different sizes, and the best performing random forest algorithm is used for our recognition module.

We evaluate MAGLeak via extensive experiments in datasets collected with several mobile phones. The results demonstrate that, 1) the performance of side-channel attack with multiple sensors is much better than that with single sensor, especially for small training datasets; 2) compared with the existing works, MAGLeak has a higher recognition accuracy of single keystroke and a lower requirements of training dataset size, e.g., achieving 92% accuracy for 200 training samples, 97% for 1000, and 98% for 2000.

The main contributions of this article is summarized as follows.

1) A side-channel-based passwords cracking system MAGLeak is developed by using machine learning techniques, which leverages the build-in sensors accelerometer, gyroscope, and magnetometer of a IIoT touch-screen smart device to recognize the passwords of a victim when the victim enters passwords on the device.
2) In the proposed system, an event-driven data collection method is proposed to record the sensors' measurements only when a victim presses the numeric keyboard and lifts the fingers from the numeric keyboard. Moreover, random forest algorithm combined with PCA algorithm, ridge regularization, and SMOTE algorithm are used for our recognition module.
3) Extensive experiments are conducted on datasets collected with several mobile phones to evaluate the performance of the proposed system. It is demonstrated that the proposed system has a higher recognition accuracy of single keystroke and a lower requirements of training dataset size than existing schemes.

II. Related Works

There are many research works on side-channel-based keystroke inference attacks, which can be roughly divided into two categories: Sensor-based attacks and signal-based attacks. Ali *et al.* [15] proposed a keystroke inference system by leveraging the unique pattern in the time-series of channel state information (CSI) when the hands and fingers of a user move in a unique formation and direction while typing a secret key. A side-channel attack was presented to crack pattern lock by analyzing imperceptible acoustic signals [16]. In [17], a keystroke snooping system was proposed by exploiting the audio signals to distinguish mm-level position difference, which can recover 94% of keystrokes.

In sensor-based attacks, an adversary leverages build-in sensors of mobile devices in combination with machine learning to infer user's typed inputs. In [18], a side-channel attack was first introduced to break the user's PIN by using camera to recognize the pattern of spatio-temporal dynamics of the hands during PIN typing. Then, an attack method was designed in [19] to analyze the shadow formation around the fingertip by leveraging the optical flow, deformable part-based model, and k-means clustering to automatically locate the touched points. In [20], an attack method was presented with camera to infer a tablet user's typed inputs with a high probability of success. However, calling the camera requires the user's permission request, which leads to the result that the attacks above are easily detected and prevented.

Ravi *et al.* [21] proposed a side-channel attack with accelerometer to recognize user activity. A side-channel attack was proposed in [22] by leveraging accelerometer's measurements to extract entered text on a smart-phone touch screen keyboard. Liu *et al.* [23] used built-in accelerometer of a smart-watch to track user's hand movements, which makes inferring user inputs on keyboards possible. Besides, Maiti *et al.* [24] introduced a side-channel keystroke inference attack with smartwatches, which employs supervised learning techniques to map the uniqueness in the captured wrist movements to each individual keystroke. Subsequently, a side-channel attack was proposed in [25] to infer mechanical lock combinations with wrist-wearable devices by capturing the movements of wrist during the unlocking process. Recently, a keystrokes inferring attack touchLogger was presented in [26] by extracting features from the motion sensor's measurements. Subsequently, a keystroke recognition system ClickLeak was explored in [27] by using the commodity WiFi devices.

Different from the above related works, we leverage multiple sensors, e.g., accelerometer, gyroscope, and magnetometer, to recognize the passwords entered on a numeric soft keyboard of a IIoT device with high efficiency and accuracy.

III. Problem Formulation

In this article, we consider the problem of launching a side-channel attack to recognize the passwords that entered by a victim on the numeric soft keyboard of a IIoT touch-screen smart device by leveraging the measured data from built-in

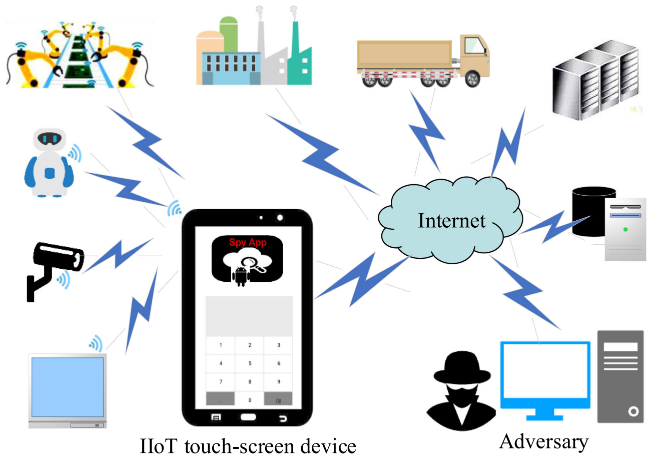

Fig. 1. Overview of MAGLeak.

TABLE I
Summary of Important Notations

Symbol	Definition
$\mathcal{V}$	The data set that we collect
$\mathbf{A}$	Input vector
$\mathbf{X}$	Input vector after dimensionality reduction
$\mathbf{Y}$	Input vector after sample inserting
$\mathbf{C}$	Covariance matrix
λ	Eigenvalues of covariance matrix
ξ	Eigenvectors of covariance matrix
ω	A k-dimensional parameter vector for data regularization
α	Regularization coefficient

TABLE II
Performance of Different Algorithms

Algorithm	Samples Num.	Time(s)	RAM(GB)	Accuracy
Random Forest	200	82.81	0.58	92.27%
	2000	200.46	1.17	97.17%
	6000	318.27	2.35	98.11%
SVM	200	88.13	0.74	80.11%
	2000	233.46	1.49	86.43%
	6000	323.23	2.51	87.86%
ANN	200	97.90	0.98	88.43%
	2000	287.45	1.84	90.47%
	6000	378.88	3.12	92.23%
K-Means	200	75.60	0.31	79.63%
	2000	155.83	0.99	83.37%
	6000	223.23	2.11	84.02%

accelerometer, gyroscope, and magnetometer. The overview of the proposed attack is shown in Fig. 1. The summary of important nations used in this article is shown in Table I.

A. Basic Requirements

It is assumed that Android system version 7.0 or above is installed on a IIoT touch-screen smart device, in which, the IIoT device is equipped with multiple sensors as follows.

1) *Accelerometer*: Accelerometer is used to capture the acceleration of the device with three sensing dimensions.
2) *Gyroscope*: Gyroscope can measure the angular rate of the device with three axes by using the Coriolis force.
3) *Magnetometer*: Magnetometer is used to capture the strength and direction of the magnetic field of the device with three dimensions.

It is also assumed that a victim of the IIoT device enters the passwords through the numeric keyboard to realize identity authentication. This assumption is reasonable in real life, as most of IIoT touch-screen smart devices are equipped with those sensors and a large number of users tend to use the numeric soft keyboard to enter passwords.

B. Adversary Model

It is assumed that the adversary controls a spy application (App) installed on a victimized smart device, who aims to obtain the victim's password. The spy App can be any App running on the smart device, since accelerometer, gyroscope, and magnetic sensor do not require any permission in most Android system. Moreover, the spy App has the capabilities as follows. First, the spy App can continuously collects the measurements from accelerometer, gyroscope, and magnetometer when the victim types the passwords by using a numeric soft keyboard. Second, when the smart device is not connected to the Internet, the spy App first stores the three sensors' measurements, and then sends them to the adversary's computer whenever the device is connected to the Internet.

In order to learn the characteristics of each number on the numeric keyboard of the IIoT device, a small dataset are required for training in the proposed side-channel attack. The minimum requirement for the size of training data is 20 simples for each number key and only a total of 100 keystrokes (as shown in Section V), in which, 2 simples can be obtained for each keystroke. Actually, there are several methods for adversary to obtain a small training data. For instance, the adversary may be an employee within an IIoT company, whose purpose is to obtain the passwords of senior managers to achieve higher operating authority of the IIoT system. Hence, he/she can touch the IIoT device physically and type a number sequence regularly to get training data.

IV. Architecture and Methodology

In this section, the architecture and methodology of MAGLeak are presented. As shown in Fig. 2, MAGLeak consists of three main parts: Data collection, data preprocessing, and training and recognition.

A. Data Collection

In order to make the measurements of the three sensors accurately describe the behavior of typing numbers on the numeric keyboard, an event-driven data collection method is proposed as follows.

The graphics of the Android system are built on Canvas. Without exception, the numeric keyboard is also built on a separate canvas. As a result, the spy App can capture the touch event by monitoring the canvas through *OnTouchEvent* provided by Android. In this way, the spy App can further monitor the

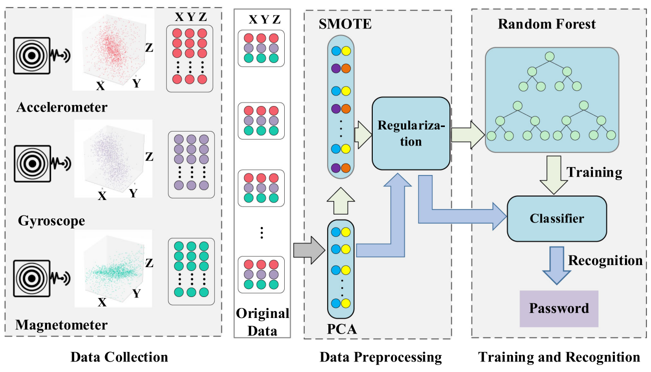

Fig. 2. Architecture of MAGLeak.

user's finger click when the numeric keyboard is opened. Actually, after capturing the numeric keyboard opening event, the spy App overloads *OnPress* and *OnUp* functions to collect the measurements with those three sensors. Specifically, the touch screen press action can be captured by the *OnPress* function when the user presses the screen, and on the other hand, the touch screen release action can be captured by the *OnUp* function when the user lifts the finger from the touch screen. If the touch screen press action (resp. release action) is captured, the sensor measurements will be record and saved to a file.

In this case, two measurements V_1 and V_2 can obtain with the three sensors for key press and release action, respectively, in which, for $i \in \{1, 2\}$,

$$V_i = \left(\overrightarrow{v_A^i}, \overrightarrow{v_G^i}, \overrightarrow{v_M^i}\right) \tag{1}$$

where

$$\begin{cases} \text{Meas. at Acce.:} \overrightarrow{v_A^i} = (V_A^i(x), V_A^i(y), V_A^i(z)) \\ \text{Meas. at Gyro.:} \overrightarrow{v_G^i} = (V_G^i(x), V_G^i(y), V_G^i(z)) \\ \text{Meas. at Magn.:} \overrightarrow{v_M^i} = (V_M^i(x), V_M^i(y), V_M^i(z)) \end{cases} . \tag{2}$$

In training process, it requires that the spy App obtains a set of sensor measurements from typing the number keys at least 100 times to achieve a high probability of successful keystroke recognition. After entering N ($N \geq 100$) numbers on the numeric keyboard by an adversary, the obtained measurements sequence at spy App can be denoted as $\mathcal{V} = (V_1, V_2, \ldots, V_{2N-1}, V_{2N})$. In recognition process, the spy App only needs to record the sensor measurements when the victim enters the passwords on the numeric keyboard. For instance, if the victim's passwords are six digits, then the obtained measurements are $\mathcal{V} = (V_1, V_2, \ldots, V_{11}, V_{12})$.

It is worth pointing out that, the proposed event-driven data collection method ensures that the sensor measurements will be recorded only when the victim enters the password. In this case, no noise data is recorded when the victim does not click on the screen, which reduces the amount of useless data to increase the accuracy and efficiency of the algorithm. Moreover, the Android functions (i.e., *OnTouchEvent OnPress* and *OnUp*) and sensors (i.e., accelerometer, gyroscope, and magnetometer) used in MAGLeak are the ones with the lowest required permissions in the Android system. In other words, we do not need to ask the victim for any additional permissions to launch such an attack. Thus, the victim will hard to be aware of being attacked.

B. Data Preprocessing

With the above data collection procession, spy App obtains the set of measurements $\mathcal{V} = (V_1, V_2, \ldots, V_{2N-1}, V_{2N})$, and then sends them to adversary's computer. After receiving the set of measurements from spy App, the adversary needs to preprocess these data in order to achieve a good performance on classification and recognition.

Dimensionality Reduction: A total of 9-D data is very unfavorable for feature extraction. The PCA algorithm [32] is used to reduce the dimensionality of the original measurements. Taking the $2N \times 9$ matrix $A = (a_{ij})_{ij} = [V_1; V_2; \ldots; V_{2N-1}; V_{2N}]$ as input, the PCA algorithm outputs a $2N \times k$ matrix X, where $k < 9$. Specifically, the mean μ_i and variance σ_i of each row $(a_{i1}, \ldots a_{i9})$ of matrix A are first computed. Secondly, zero-mean of each row of $\mathbf{A}$ are calculated to obtain a new matrix $B = (b_{ij})_{ij}$, i.e., $b_{ij} = (a_ij - \mu_i)/\sigma_i$. Third, it calculates the covariance matrix $\mathbf{C} = 2N * \mathbf{B} * \mathbf{B}^T$, where $(\cdot)^T$ is the transpose of a matrix. Fourth, it calculates the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_\iota$ of the covariance matrix $\mathbf{C}$ and the corresponding eigenvectors $\{\xi : (\mathbf{C} - \boldsymbol{\lambda}_i I)\xi = \mathbf{0}\}$, where I is identity matrix, $\lambda_1 > \lambda_2 > \cdots > \lambda_\iota$, and $i = 1, 2, \ldots, \iota$. Fifth, the feature vectors are arranged into a matrix from top to bottom according to the corresponding eigenvalue, and the first $\mathbf{k}$ rows are taken to form a $k \times 2N$ matrix $\mathbf{P}$. Finally, a $2N \times k$ matrix X is obtained by computing $\mathbf{X} = \mathbf{P} * \mathbf{A}$.

Samples Inserting: Due to the particularity of small dataset for training, it may cause several disadvantages, such as, imbalanced samples, poor model fitting and overfitting. Instead of resampling a few classes directly, SMOTE algorithm [28] can perform interpolation operations in the feature space by inserting new samples into a few samples of similar positions to realize the purpose of data enhancement. SMOTE algorithm does not simply copy the samples according to the random over-sampling method, but add new nonexisting samples, to avoid overfitting of the classifier. Taking $2N \times k$ matrix $\mathbf{X}$ as input, SMOTE algorithm outputs a $S \times k$ matrix $\mathbf{Y}$, where $S > 2N$. Specifically, if i_0th row $\mathbf{X}_{i_0}$ of matrix $\mathbf{X}$ is a minority class simple in $\mathbf{X}$, one can find the $\mathbf{l}$ samples closest to $\mathbf{X}_{i_0}$ according to Euclidean distance between simples; then randomly select a sample from the $\mathbf{l}$ samples, denoted as $\mathbf{X}_{j_0}$; finally, one can synthesize a new positive sample $\mathbf{x}' = \lambda \mathbf{X}_{i_0} + (1 - \lambda)\mathbf{X}_{j_0}$, where λ is chosen from [0,1] uniformly at random. Denoted the synthesized samples as $\mathbf{x}'_1, \ldots, \mathbf{x}'_{S-2N}$, the $S \times k$ matrix $\mathbf{Y} = [\mathbf{X}; \mathbf{x}'_1; \ldots; \mathbf{x}'_{S-2N}]$.

Data Regularization. In order to further prevent overfitting and facilitate later data training and recognition process, data regularization is used as the process of scaling individual samples to have unit norm. Assume that the training data is $D = [(Y_1, y_1), \ldots, (Y_S, y_S)]$, where Y_i is the ith row of $\mathbf{Y}$, and $y_i \in \{0, 1, \ldots, 9\}$ is the corresponding label value of Y_i. There are two ways to be considered: L^1 (Lasso regularization) and L^2

TABLE III
DATASETS

Dataset Name	Device Type	Samples Num.
Dataset A	Samsung GalaxyS10 SM-G9750	2000 $\times$ 3
Dataset B	HuaWei P20 EML-AL00	2000 $\times$ 1
Dataset C	HuaWei mate 30 TAS-AL00	2000 $\times$ 1

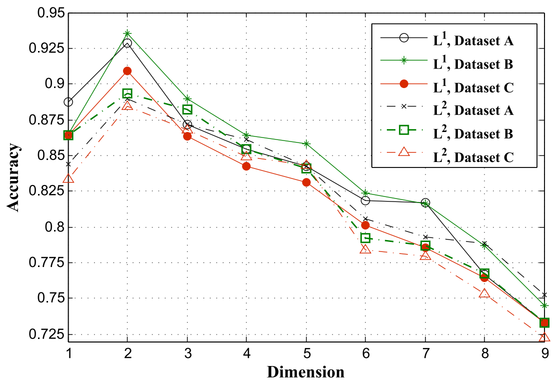

Fig. 3. Performance of MAGLeak under different preprocessing parameters and datasets.

(Ridge regularization) as follows:

$$L^1: \min_{\omega} J(\omega) = \min_{\omega}\left\{\frac{1}{S}\sum_{i=1}^{S}(\omega^T Y_i - y_i)^2 + \alpha\|\omega\|_1\right\} \quad (3)$$

$$L^2: \min_{\omega} J(\omega) = \min_{\omega}\left\{\frac{1}{S}\sum_{i=1}^{S}(\omega^T Y_i - y_i)^2 + \alpha\|\omega\|_2^2\right\} \quad (4)$$

where $\alpha(\in\mathbb{R})$ is a system parameter, $\omega(\in\mathbb{R}^k)$ is a k-dimensional parameter vector for data regularization, $\|\omega\|_1 = \sum_i|\omega_i|$, and $\|\omega\|_2 = \sum_i|\omega_i|^2$. Data regularization is to find the optimal value of ω in order to minimize $J(\omega)$. For each $Y_i = [Y_{i1}, Y_{i2}, \ldots, Y_{ik}]$, the obtained data after normalizing is $Z_i = [\omega_1 Y_{i1}, \omega_2 Y_{i2}, \ldots, \omega_k Y_{ik}]$.

An experiment is conducted to verify the effect of preprocessing parameters on the success rate of MAGLeak under different datasets, in which the random forest [29] is used for model training and numeric key recognition, and each dataset has 2000 samples (the details of datasets please refer to Table III). As shown in Fig. 3, the performance of the proposed side-channel attack is the best when L^1 regularization is adopted and the dimension is reduced to 2 (i.e., $k = 2$). Accordingly, we set the data regularization method as L^1 regularization and $k = 2$ in the proposed side-channel attack.

Another experiment is conducted to compare the classification result between before and after using the SMOTE algorithm. In practice, the collected samples for training many be unbalance due to unpredictable reasons (e.g., data collection process is interfered by others, and the collection has to be stopped) during data collection. Thus, we use SMOTE to make the dataset not seriously unbalanced. In order to show the role of SMOTE in practical situations, we take a set of numeric keys in an unbalance training dataset as follows : 0–7: 5%; 8: 25%; and 9: 35%. The L^1 regularization, dimensionality reduction ($k = 2$) and random forest are used for model training. As shown in Fig. 4, the result of the classifier with using the SMOTE is better than that without using SMOTE. Specifically, in Fig. 4, the yellow and cyan dots represent the majority of samples (that is, 8 and 9), and the dots of other colors represent other samples. In Fig. 4(a), SMOTE is not used, the data is unbalanced, and most classes occupy a large proportion, which leads to the tendency of the model to overfit. In Fig. 4(b), we used SMOTE to generate a random minority sample for the minority points, so that the sample is no longer seriously unbalanced, avoiding overfitting of the model and improving the generalization ability and antinoise ability of the model.

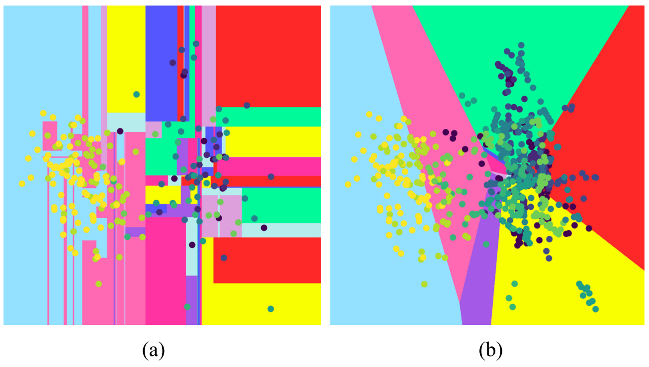

Fig. 4. Comparison the classification result between without and with using the SMOTE algorithm when the samples are imbalanced in training dataset. (a) The result without using SMOTE. (b) The result with using SMOTE.

C. Training and Recognition

Now, we need to choose a machine learning algorithm for model training. In order to ensure the stealth of the attack, only limited training can be obtained at adversary. Thus, the optional algorithms should be suitable for small datasets. Some classic machine learning algorithms, such as, the random forest [29], sequential minimal optimization trained SVM [30], multiple layer perceptron ANN [13], and K-means clustering algorithm (K-Means [14]), are considered in this article. Random forest randomness is a good choice for small dataset as the randomness makes the model have good performance on antinoise, as well as overfitting reduction. Table II compares the performance of those algorithms by taking the original training data (i.e., Dataset A in Table III) as input. As shown in this table, the random forest algorithm yields the best results in efficiency, RAM cost, and recognition accuracy of single key, among a variety of candidates.

The random forest algorithm creates a set of C4.5 decision trees and trains each decision tree using random subsamples of a given data set [31]. By looking at k-dimensional feature vector in a given feature space, the split of each node in the C4.5 tree can be determined. Let $D = [(Z_1, z_1); (Z_2, z_2); \ldots, (Z_S, z_S)]$ be the training dataset after preprocessing, where z_i is the corresponding label value of Z_i, i.e., $z_i \in \{0, 1, \ldots, 9\}$. Let $C_j = \{(Z_i, z_i) : z_i = j\}$ for any $j \in \{0, 1, \ldots, 9\}$, and F_r be the rth attribute for $r \in \{1, 2, \ldots, k\}$.

In the construction of C4.5 decision trees, some basic information theory measures are used as follows. Information entropy of dataset D can be denoted as

$$H(D) = -\sum_{j=0}^{9} \frac{|C_j|}{|D|} \log\left(\frac{|C_j|}{|D|}\right). \tag{5}$$

For each attribute F_r, let $FV_r = \{a_{r1}, a_{r2}, \ldots, a_{rn_r}\}$ be the set of attribute values of F_r in training dataset. Then, we denote the set of median of adjacent eigenvalues as

$$T_r = \left\{\frac{a_{ri} + a_{r(i+1)}}{2} : 1 \le i \le n_r - 1\right\}. \tag{6}$$

For each $t \in T_r$, we divide the training dataset into two parts: $D_t^- = \{(Z_i, z_i) : Z_i(r) \leqslant t\}$, and $D_t^+ = \{(Z_i, z_i) : Z_i(r) > t\}$. The conditional information entropy of D given F_r and $T_r = t$ can be denoted as

$$H(D|F_r, t) = -\sum_{\lambda\in\{-,+\}} \frac{|D_t^\lambda|}{|D|} \sum_{j=0}^{9} \frac{|D_t^\lambda(j)|}{|D_t^\lambda|} \log\left(\frac{|D_t^\lambda(j)|}{|D_t^\lambda|}\right) \tag{7}$$

where $D_t^\lambda(j) = D_t^\lambda \bigcap C_j$ for $\lambda \in \{-,+\}$. The information gain of $F_r, T_r = t$ can be denoted as

$$\text{Gain}(F_r, t) = H(D) - H(D|F_r, t). \tag{8}$$

Then, the split entropy is used to normalize the information gain, in which, the split entropy is as follows:

$$\text{SplitEntr}(F_r) = -\sum_{\lambda\in\{-,+\}} \frac{|D_t^\lambda|}{|D|} \times \log_2\left(\frac{|D_t^\lambda|}{|D|}\right). \tag{9}$$

Finally, the normalized information gain (i.e., gain rate) of attribute F_r at $T_r = t$ for dataset D is shown as follows:

$$\text{GainRatio}_D(F_r, t) = \frac{\text{Gain}(F_r, t)}{\text{SplitEntr}(F_r)}. \tag{10}$$

The maximum information gain rate of attribute is used to split the attribute in random forest algorithm.

Taking training dataset D, feature set $\mathcal{F} = \{F_r\}_r$, training round (i.e., the number of trees in the forest) TN, the maximum depth of each tree $MaxD$, the minimum number of samples required to split an internal node $MinS$, and threshold Th as inputs, the random forest algorithm outputs a classifier $H(\cdot)$.

Specifically, for each $b \in \{1, 2, \ldots, TN\}$, a set of samples DR_b with size θ is chosen from D uniformly at random. Then, a random-forest tree T_b can be grow by recursively repeating the following steps for each terminal node of the tree. Step (1): Let D^* be the current dataset ($D^* = DR_b$ at the beginning). If the depth of the current tree is less than $MaxD$, $|D^*| \geq MinS$, and maximum information gain rate for D^* is larger than Th, pick the best (F_{r^*}, t^*) split-point according to the gain rate for current dataset D^*, i.e.,

$$(F_{r^*}, t^*) = arg \max_{(F_r,t)} \text{GainRatio}_{D^*}(F_r, t). \tag{11}$$

Step (2): Let $D_{t^*}^-$ (resp. $D_{t^*}^+$) be the subset of D^* such that the value of r^*th feature in each element is less (resp. larger) than t^*. For each $\lambda \in \{-,+\}$, split the node (F_{r^*}, t^*) into to two daughter nodes with step (1) by taking $D^* = D_{t^*}^\lambda$. Finally, outputs a tree $\mathbf{T}_b$ for randomly chosen dataset DR_b. After generating the set of trees $\{\mathbf{T}_b\}_b^{TN}$, a final classifier $H(\cdot)$ can be obtained.

For keystroke recognition, the classification of a given keystroke instance is by pushing the instance (after data preprocessing) down into each tree in the forest and counting the tags of training instances that match the test instance as "votes" to represent specific trees. Then, it only needs to count the votes and the most frequent votes in the trees is treated as a prediction of the forest.

V. Experiment

In this section, we first evaluate the performance of the proposed side-channel attack MAGLeak on the real datasets collected from multiple mobile phones. Then, we discuss how to prevent this type of side-channel attack.

Experimental Settings: As shown in Table III, in our experiment, the dataset are collected from several mobile phones under different collected angles. Hence, our attack method can be applied to different situations in the real IIoT. The 10-fold cross validation is used in our experiments to obtain the accuracy of MAGLeak. Specifically, the initial dataset are first divided into ten subsets evenly; then each subset is used for testing and the rest for training; finally, the average results of ten runs is calculated as the final result. All experimental evaluations are conducted on Windows 10 operating system under Intel(R) Core(IM) i7-8750H CPU @2.20GHZ and 16 GB RAM.

A. Impact of System Parameters

We first consider the effect of the system parameters on the proposed attack. Actually, in a random forest, there are several important parameters that need to be set: the number of trees in the forest, TN; the maximum depth of each tree, $MaxD$; and the minimum number of samples required to split an internal node, $MinS$. In this experiment, we always consider all features used for generated trees instead of a random subset of all features. The dataset used in this experiment is Dataset A, in which the first 2000 samples are included.

Fig. 5(a) shows the numeric key recognition success rate (i.e., accuracy) at adversary with the change of the number TN of trees in the forest. The result shows that the accuracy increases with increasing the number TN of trees. However, the larger the value of TN is, the longer the running time of the algorithm. When $TN = 100$, the curve tends to be flat, which means a low time cost and high accuracy. Accordingly, we set $TN = 100$ in the proposed side-channel attack.

Fig. 5(b) plots the accuracy of numeric key recognition at adversary with the change of the maximum depth $MaxD$ of each tree. The maximum depth $MaxD$ is used to prevent overfitting. It can be seen that, the accuracy first increases with the increase of $MaxD$, reaches the peak value at $MaxD = 14$, and then decreases with the increase of $MaxD$. The reason is that the deeper the decision tree is, the stronger correlation between obtained trees, which further weakens the generalization ability

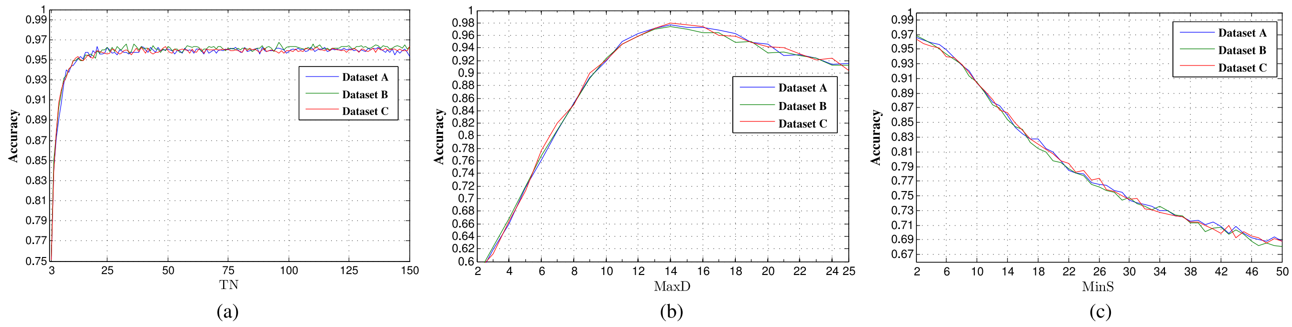

Fig. 5. Effect of the system parameters on MAGLeak. (a) The number of trees in the forest, TN. (b) The maximum depth of each tree, $MaxD$. (c) The minimum number of samples, $MinS$.

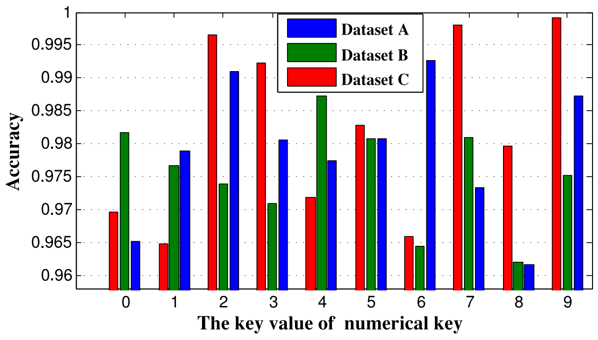

Fig. 6. Recognition accuracy of ten numeric keys under different datasets.

of the forest. Hence, we set $MaxD = 14$ in our system as in this case the highest accuracy can obtained.

Fig. 5(c) plots the accuracy of numeric key recognition at adversary with the change of the minimum number $MinS$ of samples required to split an internal node. This value limits the minimum number of samples of the leaf node. If the number of leaf nodes is less than the number of samples, it will be pruned with the sibling nodes. As shown in this figure, the accuracy decreases with the increase of $MinD$. Hence, we set $MinS = 2$ in our system.

Finally, an experiment is conducted to test the success rate of the proposed side-channel attack under different datasets by using 10-fold cross-validation. Fig. 6 plots the recognition accuracy of ten numeric keys under different datasets. It can be seen that the accuracy of all numeric keys is greater than 96%, and the average success rate is as high as 97.87%. Accordingly, the proposed attack has a good performance in recognition accuracy of single key.

B. Impact of Dataset Size and Number of Sensors

Since the acquisition process of training data may expose the adversary's attack intention, it is necessary to discuss the impact on the attack success rate under the size of the training dataset. Moreover, because of the fact that not all IIoT smart devices are equipped with those three sensors, it is also necessary to study the performance of the proposed attack when a single or two sensors are available.

We first conduct an experiment to observe the effect of parameter changes on the accuracy of numeric key recognition under different dataset size and different number of available sensors. In this experiment, the training dataset size includes 200, 500, 1000, and 2000; and the available sensors can be any single sensor of those three sensors, combination of any two sensors among those three sensors, and all three sensors.

Fig. 7 plots the accuracy of numeric key recognition under different dataset size and different number of available sensors, in which, we set $TN = 100$ and $MinS = 2$, and the training dataset is randomly chosen from Dataset A. The result shows that for different sizes of training dataset and different available sensors, there is a slight difference in the optimal values of $MaxD$ to maximum the accuracy, such as, the optimal values of $MaxD$ is 11 when the training dataset size is 200 and only gyroscope is available, and the optimal values of $MaxD$ is 15 when the training dataset size is 2000 and only accelerometer is available. In order to reduce the time to adjust many parameters to improve the efficiency of the algorithm, the parameter $MaxD$ is fixed at 14 in our system instead of changing with the size of training dataset.

From Fig. 7, it can also be seen that: (1) The performance of the proposed attack by using the combination of any two sensors or all three sensors is better than that by using only one single sensor; (2) for small training dataset (e.g., the training dataset size: 200 and 500), the accuracy of the proposed attack with all three sensors is larger than that with the combination of any two sensors; (3) for a relatively large training dataset (e.g., the training dataset size: 1000 and 2000), the performance of the attack with the combination of any two sensors is almost the same as that with all three sensors.

We conduct another experiment to further verify the above results. As shown in Fig. 8, we discuss the performance of the proposed attack in different dataset under different training dataset size and number of available sensors. The results show that (1) multisensor measurements are much better reflect the behavior of pressing numeric keys than single sensor measurements; (2) the recognition accuracy of three sensor measurements is more stable than that of two sensor measurements, especially when only a small data training dataset can be obtained.

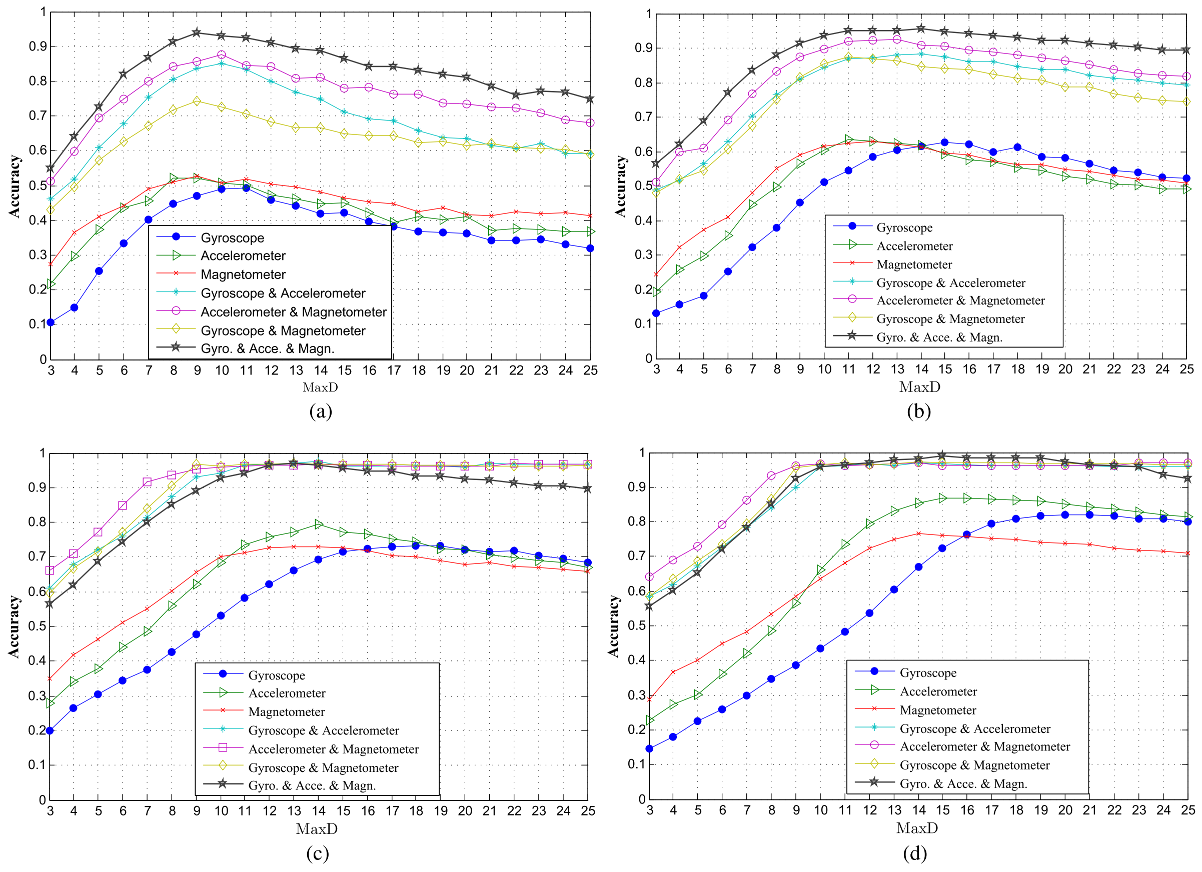

Fig. 7. Effect of the system parameters on MAGLeak. (a) Training dataset size = 200. (b) Training dataset size = 500. (c) Training dataset size = 1000. (d) Training dataset size = 2000

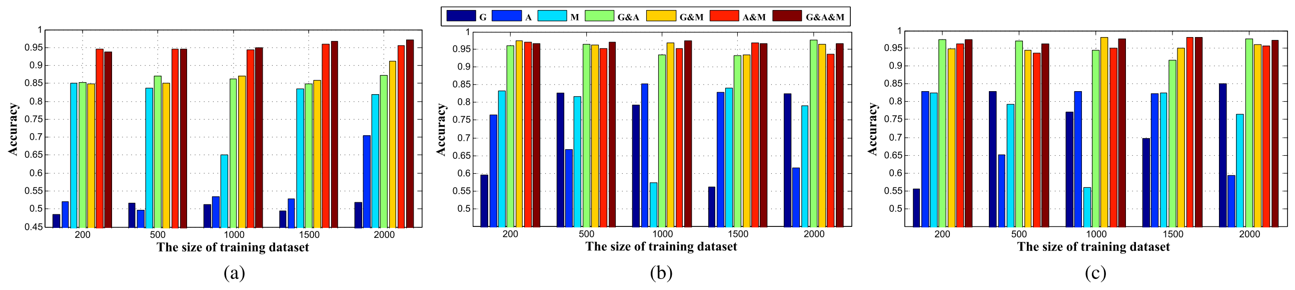

Fig. 8. Performance of MAGLeak under different training dataset size and number of available sensors.(a) Dataset A. (b) Dataset B. (c) Dataset C.

C. Comparison and Defence

We now compare the existing keystroke inference works (i.e., TouchLogger [26], ClickLeak [27], WiKey [15]) with MAGLeak. As shown in Table IV, TouchLogger achieves 72% accuracy of single keystroke by using motion sensor to measure 449 samples; ClickLeak achieves 83% accuracy of single keystroke by leveraging WiFi signals with the help of build-in acceleration and microphone sensors of mobile phone; and WiKey achieves 96% accuracy of single keystroke by utilizing the CSI stream of continuous WiFi signals. Compared with the above works, MAGLeak have a higher accuracy of single keystroke and a lower requirements of training dataset size. The performance

TABLE IV
COMPARISON WITH EXISTING WORKS

Schemes	Measurements	Samp. Num.	Accuracy
TouchLogger [26]	Motion Sensor	449	72%
ClickLeak [27]	WiFi/Microphone/Acce.	5000	83%
WiKey [15]	WiFi Signal	12950	96%
MAGLeak	Gyro./Acce./Magn.	200	92%
		1000	97%
		2000	98%

of WiKey is close to MAGLeak. However, the WiFi signal is required in MAGLeak, which will be greatly affected by signal interference in IIoT environment.

Then, we discuss possible strategies to defend against the proposed side-channel attack. Since current Android operating system does not ask users about access rights for accelerometer, gyroscope, and magnetometer, a malicious applications can obtain the measurements from these three sensors without any restrictions. It suggests that introducing new permissions into the Android OS to control the access rights so that the user can choose whether to grant the permission to an App. Moreover, given a specific type and model of IIoT smart device, the numeric keypad for entering passwords usually has a fixed design pattern. In other words, the numerical arrangement order and position of the numeric keyboard does not change with time. Therefore, the randomization of user interface attributes (e.g., the position of the keyboard, the relative position of each key on the keyboard, etc.) can be leveraged to defend against such kind of side-channel attacks.

VI. Conclusion

In this article, a novel side-channel attack MAGLeak was presented to infer user's passwords by leveraging multiple build-in sensors (i.e., accelerometer, gyroscope, and magnetometer) of IIoT touch-screen device. In order that the measurements of three sensors accurately reflect the user's keystroke behavior, an event-driven data collection method was proposed, which only records the measurements when the press action or release action of touch screen happened. In MAGLeak, the random forest algorithm was used for model training and passwords recognition. Extensive experiments were conducted on real dataset obtained from several mobile phones. The experimental results show that the proposed learning model has a better performance than the existing works. For the future work, we will focus on keystroke inference for alphabet keyboard by using multiple build-in sensors of smart devices.

References

[1] S. Vitturi, C. Zunino, and T. Sauter, "Industrial communication systems and their future challenges: Next-generation ethernet, IIoT, and 5G," *IEEE Proc. IRE*, vol. 107, no. 6, pp. 944–961, May 2019.

[2] F. Al-Turjman and S. Alturjman, "Context-sensitive access in industrial Internet of Things (IIoT) healthcare applications," *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2736–2744, Jun. 2018.

[3] N. Zhang, S. Zhang, P. Yang, O. Alhussein, W. Zhuang, and X. S. Shen, "Software defined space-air-ground integrated vehicular networks: Challenges and solutions," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 101–109, Jul. 2017.

[4] J. Wang, C. Jiang, K. Zhang, X. Hou, Y. Ren, and Y. Qian, "Distributed q-learning aided heterogeneous network association for energy-efficient IIoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2756–2764, Apr. 2020.

[5] M. Hassan, A. Gumaei, S. Huda, and A. Almogren, "Increasing the trustworthiness in the industrial IoT networks through a reliable cyber-attack detection model," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6154–6162, Sep. 2020.

[6] R. Chaudhary *et al.*, "SDN-enabled multi-attribute-based secure communication for smart grid in IIoT environment," *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2629–2640, Jan. 2018.

[7] S. Chen, Z. Pang, H. Wen, K. Yu, T. Zhang, and Y. Lu, "Automated labeling and learning for physical layer authentication against clone node and sybil attacks in industrial wireless edge networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 2041–2051, Mar. 2021.

[8] A. Makkar, S. Garg, N. Kumar, M. S. Hossain, A. Ghoneim, and M. Alrashoud, "An efficient spam detection technique for IoT devices using machine learning," *IEEE Trans. Ind. Informat.*, vol. 17, no. 2, pp. 903–912, Feb. 2021

[9] D. Chen, N. Zhang, N. Cheng, K. Zhang, Z. Qin, and X. Shen, "Physical layer based message authentication with secure channel codes," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 5, pp. 1079–1093, Sep./Oct. 2020.

[10] G. Falco, C. Caldera, and H. Shrobe, "IIoT cybersecurity risk modeling for scada systems," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4486–4495, Dec. 2018.

[11] J. Lee, M. Tehranipoor, C. Patel, and J. Plusquellic, "Securing designs against scan-based side-channel attacks," *IEEE Trans. Dependable Secure Comput.*, vol. 4, no. 4, pp. 325–336, Oct.–Dec. 2007.

[12] M. N. N. Abrishamchi, A. H. Abdullah, A. D. Cheok, and K. S. Bielawski, "Side channel attacks on smart home systems: A short overview," in *Proc. IEEE IECON*, 2017, pp. 8144–8149.

[13] N. Talebi, A. M. Nasrabadi, and I. Mohammad-Rezazadeh, "Estimation of effective connectivity using multi-layer perceptron artificial neural network," *Cogn. Neurodyn.*, vol. 12, no. 1, pp. 21–42, 2018.

[14] U. R. Raval and C. Jani, "Implementing & improvisation of k-means clustering algorithm," *Int. J. Comput. Sci. Mobile Comput.*, vol. 5, no. 5, pp. 191–203, 2016.

[15] K. Ali, A. X. Liu, W. Wang, and M. Shahzad, "Keystroke recognition using WiFi signals," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw.*, 2015, pp. 90–102.

[16] M. Zhou *et al.*, "Patternlistener: Cracking android pattern lock using acoustic signals," in *Proc. 2018 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 1775–1787.

[17] J. Liu *et al.*, "Snooping keystrokes with mm-level audio ranging on a single phone," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw.*, 2015, pp. 142–154.

[18] D. Shukla *et al.*, "Beware, your hands reveal your secrets," in *Proc. ACM SIGSAC*, 2014, pp. 904–917.

[19] Q. Yue *et al.*, "Blind recognition of touched keys on mobile devices," in *Proc. ACM SIGSAC*, 2014, pp. 1403–1414.

[20] J. Sun *et al.*, "Visible: Video-assisted keystroke inference from tablet backside motion," in *Proc. NDSS*, 2016, pp. 1–15.

[21] N. Ravi *et al.* "Activity recognition from accelerometer data," in *Proc. AAAI*, 2015, pp. 1541–1546.

[22] E. Owusu *et al.*, "Accessory: Password inference using accelerometers on smartphones," in *Proc. 12th Workshop Mobile Comput. Syst. Appl.*, 2012, pp. 1–6.

[23] X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang, "When good becomes evil: Keystroke inference with smartwatch," in *Proc. ACM SIGSAC*, 2015, pp. 1273–1285.

[24] A. Maiti, M. S. Jadliwala, J. He, and I. Bilogrevic, "(Smart) watch your taps: Side-channel keystroke inference attacks using smartwatches," in *Proc. ACM Int. Symp. Wearable Comput.*, 2015, pp. 27–30.

[25] A. Maiti, R. Heard, M. Sabra, and M. Jadliwala, "Towards inferring mechanical lock combinations using wrist-wearables as a side-channel," in *Proc. 11th ACM Conf. Secur. Privacy Wireless Mobile Netw.*, 2018, pp. 111–122.

[26] L. Cai and H. Chen, " Touchlogger: Inferring keystrokes on touch screen from smartphone motion." in *Proc. HotSec*, 2011, pp. 1–9.

[27] F. Li, X. Wang, H. Chen, K. Sharif, and Y. Wang, "ClickLeak: Keystroke leaks through multimodal sensors in cyber-physical social networks," *IEEE Access*, vol. 5, pp. 27311–27321, Nov. 2017.

[28] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, 2002.

[29] M. Pal, "Random forest classifier for remote sensing classification intern," *J. Remote Sens.*, vol. 26, no. 1, pp. 217–222, 2005.

[30] J. Platt, "Sequential minimal optimization: A Fast algorithm for training support vector machines," Tech. Rep. MSR-TR-98-14, pp. 1–21, 1998.

[31] Q. J. Ross, *C4. 5: Programs for Machine Learning*. San Mateo, CA, USA: Elsevier, 1993.

[32] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics Intell. Lab. Syst.*, vol. 2, nos. 1–3, pp. 37–52, 1987.

Dajiang Chen (Member, IEEE) received the Ph.D. degree in information and communication engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2014.

He was an Associate Professor with the School of Information and Software Engineering, University of Electronic Science and Technology of China. He was a Postdoctoral Fellow with the BBCR group, Department of ECE, University of Waterloo, Waterloo, ON, Canada, from 2015 to 2017. His current research interest includes physical layer security, secure channel coding, and machine learning and its applications in wireless security.

Dr. Chen was the workshop Chair for BDEC-SmartCity'19 in conjunction with IEEE WiMob 2019 and the organizer for IoT track in conjunction with EAI CollaborateCom 2020. He also serves/served as a Technical Program Committee Member for IEEE Globecom, IEEE ICC, IEEE VTC, IEEE WPMC, and IEEE WF-5G.

Zihao Zhao is currently a undergraduate student majoring with Cyber Security with the School of Information and Software Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu, China. His current research interests include wireless security, machine learning, federated learning, etc.

Xue Qin received the B.S. degree in telecommunication engineering from North University of China, Taiyuan, China, in 2008.

From 2016 to 2017, she was a Postgradaute Student with Conestoga College, Kitchener, ON, Canada, where she graduated in 2017. She is now a graduate student with the Department of Electrical and Computer Engineering, University of Windsor, Canada. Her research interests include communication networks, security, and machine learning.

Yaohua Luo received the Ph.D. degree in geodetection and information technology from the Chengdu University of Technology, Chengdu, China, in 2013.

He is currently an Associate Professor with the School of Information Science and Technology, Chengdu University of Technology, Chengdu, China. He was a Post-doctor with the Institute for Mineral and Energy Resources of University of Adelaide, Adelaide, Australia. His research interests include network security, machine learning, optimization methods, and geophysics.

Mingsheng Cao received the B.Sc. and M.Sc. degrees in computer science and software and theory from the University of Electronic Science and Technology of China, 2008 and 2011, respectively, and the Ph.D. degrees in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2019.

Currently, he is a Lecturer with the Network and Data Security Key Laboratory of Sichuan Province, University of Electronic Science and Technology of China, Chengdu, China. His research interests include network security and pervasive computing.

Hua Xu received the M.S. and Ph.D. degrees in information and communications engineering from Soochow University, Nanjing University of Posts and Telecommunications, Nanjing, China, in 2003 and 2007, respectively.

He is currently a Professor with the School of Physics and Electronic Engineering, Yancheng Teachers University, Yancheng, China. He was a Visiting Scholar with the BCWS Centre of Carleton University, Canada, from August 2010 to February 2011. His research interests include LDPC codes design, compressed sensing and magnetic induction communication.

Anfeng Liu received the M.Sc. and Ph.D degrees in computer science from Central South University, Changsha, China, in 2002 and 2005, respectively.

He is currently a Professor with the School of Computer Science and Engineering, Central South University, Changsha, China. He is also a Member (E200012141 M) of the China Computer Federation (CCF). His major research interest include wireless sensor networks, Internet of Things, information security, edge computing, and crowdsourcing.