IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 15, 2020, accepted December 24, 2020, date of publication January 8, 2021, date of current version January 20, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3050239

Offensive Security of Keyboard Data Using Machine Learning for Password Authentication in IoT

KYUNGROUL LEE[1], JAEHYUK LEE[1], CHANG CHOI[2], (Senior Member, IEEE), AND KANGBIN YIM[3,4]

[1]School of Computer Software, Daegu Catholic University, Gyeongsan 38430, South Korea
[2]Department of Computer Engineering, Gachon University, Seongnam 13120, South Korea
[3]Department of Information Security Engineering, Soonchunhyang University, Asan 31538, South Korea
[4]Department of Software Convergence, Graduate School of Soonchunhyang University, Asan 31538, South Korea

Corresponding author: Kangbin Yim (yim@sch.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea Government [Ministry of Science and ICT (MSIT)] under Grant 2018R1A4A1025632, and in part by the Soonchunhyang University Research Fund.

ABSTRACT In this paper, to increase the attack success rate, we analyzed the distribution of all collected keyboard data based on the distance of time and keyboard scancode data, which presents the crucial data from the previous study. To achieve this, we derived time-distance based features that have higher attack success rates than in previous studies. The proposed attack method defines 6 features, and evaluates the performance based on 18 datasets. For performance evaluation, the accuracy, precision, recall, F1-score, and AUC of Datasets (1 to 3) were compared, and two experiments showed improved overall performance by at least 10.6 % and up to 16.1 % compared to previous studies in terms of the performance evaluation for each feature, comparison of variations in maximum performance, comparison of variations in performance of each feature, and comparison of variations in overall performance. Moreover, the best accuracy, which represents the probability of password exposure, was 96.7 %, which suggests that our proposed attack method has a higher accuracy than the previous study (96.2 %). In conclusion, we demonstrated that password authentication is neutralized by stealing the user password more effectively. For future research, we will focus on improving the attack success rate with respect to accuracy and overall performance numbers, using not only machine learning, but also deep learning.

INDEX TERMS Offensive security, vulnerability analysis, password authentication, user authentication, machine learning.

I. INTRODUCTION

IoT refers to a technology in which various things with sensors are connected to the Internet or network, and the data collected by the sensors are shared and utilized via the connected Internet or network [21]. With the advent of the fourth industrial revolution, the importance of IoT has triggered an exponential increase in the number of IoT devices, and with it, security problems have also emerged [22]. These problems caused by exposure and manipulation of data collected from sensors with regards to integrity, confidentiality of data transmitted via the network, and security of sensor data and privacy information that is stored in a centralized server [23].

Since the data transmitted from users and sensors is stored in the server, its security is particularly important. Among the security technologies in the server, the user authentication technology is crucial, since both the sensor and the user require authentication in order to access the server. The major user authentication technologies include; password authentication method, a one-time password (OTP) [24], and biometric authentication using iris and fingerprint [25]. Among them, the password authentication is most widely used due to its ease of deployment [3]. However, when authentication information such as ID and password are exposed to an attacker, the attacker takes over the server, which in turn poses a threat to the entire IoT [4]. In other words, authentication information such as ID and password is input from a keyboard device, and hence when the keyboard data is stolen by the

The associate editor coordinating the review of this manuscript and approving it for publication was Patrick Hung.

attacker, the authentication information is also stolen with it [1].

Consequently, an attacker can take over the IoT server using the stolen authentication information and control various IoT applications and devices that communicate with the server. For instance, the attacker can use the stolen authentication information to connect to the IoT server of the victim's smart home, and then manipulate all home devices connected to the server in order to cause damage [26]. Specifically, the attacker connects to the victim's smart home IoT server with stolen authentication information and access a smart home CCTV connected to the server to pry into the victim's privacy. Particularly, this attack poses a security threat that exposes the privacy.

There are various user authentication methods, such as password and OTP, in which sensitive information is input from the keyboard [1]. Although password authentication is an old method, it is widely used, due to its easy-to-deploy advantages [2]. In IoT, most PC platforms, such as servers, use password authentication [3]. For this reason, if a user password is exposed, core systems in IoT are taken over by an adversary, causing a serious problem that may lead to malfunction of control systems or sensors [4]. In other words, when a password, which is data input from the keyboard device, is exposed, password-based user authentication is neutralized.

From an attacker's point of view, neutralizing user authentication is exceedingly attractive, and various attack techniques have been designed to steal keyboard data. Keyboard data attack technologies have emerged for the PS/2 interface keyboard, such as attack using keyloggers [5], direct polling attack [4], C/D bit exploitation vulnerability attack [6], and RESEND command utilization attack [7]. Keylogger is keyboard data recording tool that is easily available in the internet. The direct polling attack preempts the acquisition of keyboard data by periodically checking whether keyboard data is input or not. C/D bit exploitation vulnerability attack uses the C/D bit, which is the information for recognizing random keyboard data generated by a defense tool, to steal the real keyboard data input from the user [6]. Finally, RESEND command utilization attack is a replay attack that requests the keyboard data last sent from the keyboard, and is the most powerful attack technique [7].

The direct polling attack, C/D bit exploitation vulnerability attack, and RESEND command utilization attack are categorized as hardware-level attack techniques. Recent state-of-the-art attack technologies have resulted into attacks caused by vulnerabilities in hardware, such as hardware architecture, i.e. meltdown [16], [17]. This study is also based on a vulnerability in which an attacker preempts the keyboard data input from a user as a result of structural vulnerability in the keyboard controller as a hardware-level attack. In particular, we proposed an attack technique that utilizes the latest attack technology, hardware attack and machine learning [18]–[20].

The attack techniques described above steal user passwords by perfectly preempting keyboard data. However, attack tools used to carry out these attacks cause an overload on the system, such as CPU usage [8], i.e., abnormal behavior caused by attack tools can be detected by the defender. Moreover, a keyboard data security technique that prevents the disclosure of user-input real keyboard data by generating fake keyboard data has been proposed [8]. Therefore, the attacker needed improved attack techniques to steal keyboard data input from the user by classifying the fake keyboard data without causing abnormal behavior; accordingly, a machine learning based keyboard data attack technique was designed [8]. This attack technique classifies the two data using machine learning models based on the collected fake keyboard data and real keyboard data, and the features used in this technique are the elapsed time and scancode. Namely, the keyboard data is classified based on the difference between the elapsed time by the user inputs, and the elapsed time generated by the defender, and the best accuracy was 96.2 %.

To effectively increase the attack success rate, we analyzed the distribution of keyboard data based on the distance with respect to time and keyboard scancode. As a result, we derived that the real keyboard data has a clustering feature according to the distance between time-scancodes. Therefore, in this paper, we defined the distance according to time-keyboard data for each feature, and features were derived with a higher attack success rate than in the previous study that defined only elapsed time and scancode as features [8]. Finally, we analyzed the distance between time and scancode by analyzing the distribution of data from the previous study [8], which extends the knowledge of the previous study to enhancing the classification performance of real keyboard data from fake keyboard data. Therefore, we expected that the performance would be improved if features that apply the algorithm for measuring distance were defined. The experiment results showed that the features defined in this paper have higher accuracy than the existing attack techniques.

The contributions of this study are as follows:

- The proposed attack method has a higher attack success rate than the existing attack method. Having a high attack success rate means effectively stealing keyboard data input from a user, i.e., the proposed method can neutralize authentication by effectively stealing the user password in password authentication.
- We analyzed the distribution of collected data to derive the characteristics of fake keyboard data generated by the defense tool and real keyboard data input from the user, and found that there is a new feature that clusters according to distance between time difference and scancodes.
- Using the features defined in this paper, the proposed attack technique has a higher accuracy (96.7%) than the existing attack techniques (96.2%).
- When comparing the average of all performance evaluation results, such as accuracy, precision, recall, F1-score, and AUC, between the existing attack technique and the proposed attack technique, the overall performance

numbers of the two features increased by at least 10.6 %, and up to 16.1 %. In other words, this paper is novel, in that it proposed an improved attack technique.

This paper is organized as follows. Section 2 describes the configuration of the attack system and the defined features, and Section 3 describes the dataset configuration and the experiment results for each defined feature. Finally, Section 4 concludes the paper.

II. SYSTEM CONFIGURATION AND FEATURE DEFINITION

This section presents the overall configuration of the attack system applying the proposed attack technique, and describes the features defined to effectively classify fake data and real data.

A. ATTACK SYSTEM CONFIGURATION

Fig. 1 shows an attack scenario using the proposed attack technology. The system for the attack scenario consists of an attack server, victim home environment, and victim terminal. The attack server provides features such as malicious code that an attacker installs in the victim's terminal, a server that connects to install malicious code, and a mail server used for sending spam mails. Through these features, the attacker attempts to penetrate the victim's terminal. The victim's smart home environment and terminal consists of a smart home environment that the victim connects to and uses a terminal associated with the environment. The victim's smart home environment is composed of home IoT devices, such as CCTV cameras, air conditioner, heating device, and a central server that connects and manage all the devices. The victim terminal is a device associated with the victim's smart home IoT server, and its role in this paper is to input the authentication information.

The attack scenario was derived as in the environment configured above, and it is shown in Fig. 1. The attack scenario is a smart home attack scenario that deploys the proposed attack technology, and consists of system penetration and stages including; the attack tool installation stage, stealing information stage, and information abuse stage.

Step 1. System penetration and attack tool installation

We assumed that the attacker prepares an attack tool in advance in order to steal the victim's keyboard data input, as in the proposed method. The attacker establishes an attack environment that use phishing sites or spam mails to installs an attack tool in the victim's terminal in order to penetrate the victim's system. Consequently, when the victim connects to the phishing site created by the attacker or opens the spam mail, the attack tool to steal keyboard data is installed on the victim's terminal or system. This process consists the system penetration and attack tool installation stage.

Step 2. Stealing information

An attacker can access the victim's terminal through the system penetration process highlighted in Step 1 and take over the system. While that attacker takes control of the system, malicious actions, and the attack scenario are performed

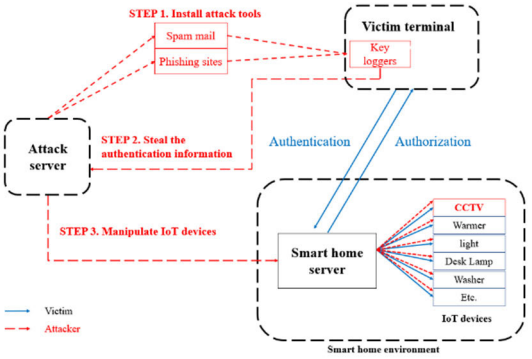

FIGURE 1. Smart home attack scenario composition deployed attack technology.

as highlighted in this paper and can be used to steal important information input by the victim. From this important information, authentication information, used by the victim to interact with his or her smart home environment could be contained. Therefore, when the victim accesses the smart home server and devices, all information that is input from the keyboard is transmitted to the attacker. This is because the attacker has already penetrated the victim's system as Step 1 and installed attack tools such as a keylogger to steal the victim's keyboard data input. This process is the stealing information stage.

Step 3. Information abuse

An attacker steals the victim's authentication information, and abuses this information by using it for authentication as a victim in a smart home IoT server or device. The attacker invades the server and the device using the victim's authentication information stolen as in Step 2, which means that he/she can access all smart home devices connected to the server. Finally, the attacker invades the server and the devices using the victim's authentication information, posing a security threat since he/she can perform malicious actions such as controlling or manipulating the victim's smart home environment through the server with these privileges. This process is termed as the information abuse stage.

Explaining the security threat related to CCTV specifically, an attacker connects to the victim's smart home IoT server and then connects to the victim's smart home IoT server, thereby the attacker could perform malicious action, such as triggering a fire by forcing an air conditioner to extremely run or prying into the victim's privacy by stealing images transferred from CCTV cameras. Finally, when the attacker steals the victim's authentication information, all smart home applications or devices that are connected to the smart home server are exposed, resulting in a chain of security threats.

Fig. 2 shows the configuration of the proposed attack technique. The system consists of data acquisition phase, feature extraction phase, pre-processing phase, dataset configuration phase, machine learning phase, and classification phase. In the data acquisition phase, both fake keyboard data and real keyboard data are collected in a situation where keyboard data protection technology by generating fake keyboard data is applied in the real-world. In the feature

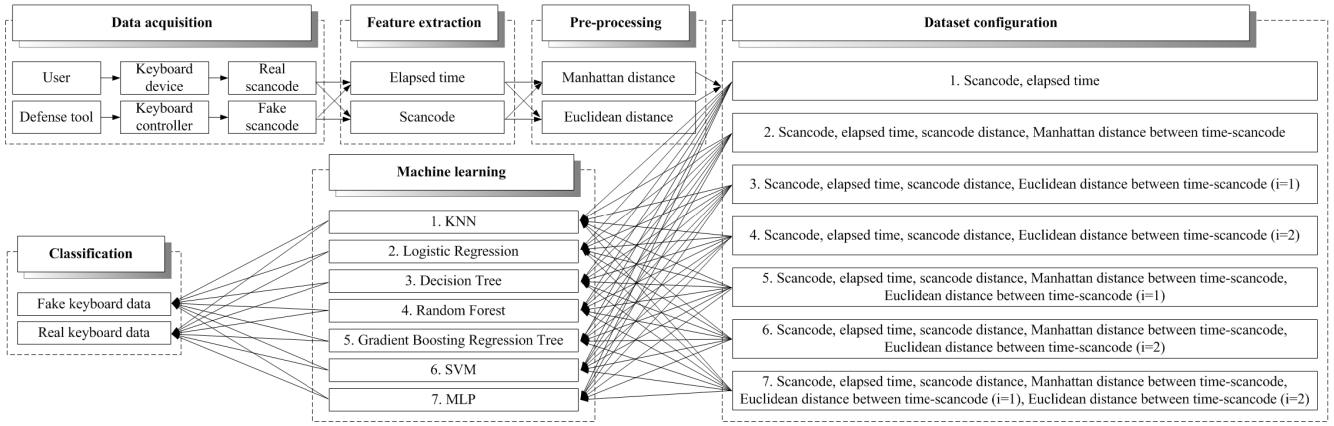

FIGURE 2. Configuration of the whole attack system applying the proposed attack technique.

extraction phase, features are extracted to construct a dataset for learning in machine learning models for data analysis. In the pre-processing phase, distance is calculated based on elapsed time and scancode data to use as features. The dataset configuration phase is a step for constructing datasets for machine learning by defining features based on the results of the pre-processing phase. The machine learning phase is a step for learning data using various machine learning models based on the configured datasets. Finally, the classification phase is to determine fake keyboard data and real keyboard data, by classifying collected keyboard data based on the learned results.

The experimental design is represented by the attack system constructed as shown in Fig. 2, and we verified the possibility of classifying the keyboard data input by the user using machine learning models. To achieve this, the experimental design of the attack system consists of data acquisition phase, extraction phase, pre-processing phase, dataset configuration phase, machine learning phase, and classification phase. Conversely, the factorial design is represented by 7 datasets in dataset configuration. The collected data possessed the characteristics of clustering data according to time and distance. Accordingly, using time and distance as features, various datasets were constructed in order to enhance the attack success rate. Consequently, the defined features are dependent variables of the factorial design, which have a very close relationship with the attack success rate. However, no interaction exists between them.

B. PROCESS OF EACH STEP

1) DATA ACQUISITION

In the data acquisition phase, both fake keyboard data generated by the defense tool, and real keyboard data input from the user are collected for data analysis. The defense tool sends a 0xD2 command to the keyboard controller to generate a scancode, which is the keyboard data, and a scancode to be generated is then passed to the keyboard controller. The keyboard controller that received the command and the scancode passes the scancode to the operating system, where the attack tool collects it (the scancode). On the other hand, the user inputs a key to the keyboard device, and the keyboard device passes the scancode corresponding to the input key to the operating system. The attacker then collects the scancode input from the user. In this phase, both fake keyboard scancode and real keyboard scancode are collected by repeating these collection steps, and we construct a dataset based on the entire collected data.

To collect keyboard data, we constructed an attack system in real-world, collected all data generated and input by implementing attack tool and defense tool, and utilized them as datasets. For example, the attack system collects the scan code input from the keyboard, A1, A2, A3…An. In this process, defense tool periodically generates random scan code, B1, B2…Bn, to deceive attackers. As a result, the attack tool collects both the scancode input from the user and the scan code generated by the defense tool, A1, A2, B1, A3, B2, B3, A4, B4, B5, A5, …, Bn, An. The objective of this paper is to verify the feasibility of stealing keyboard data using machine learning. Namely, the attacker steals a password input from user, which means that the user's consent is not required. Therefore, this paper is based on the attack technique, where the user's consent is not required.

2) FEATURE EXTRACTION

In the feature extraction phase, features are extracted from the keyboard data collected in the data acquisition phase for data analysis. Information that can be used as a feature in the collected data are the elapsed time (collected time) and the scancode (keyboard data). The elapsed time denotes the time when the fake keyboard data is transferred from the keyboard controller, and the time when the user inputs the real keyboard data from the keyboard device. The scancode denotes fake keyboard data and real keyboard data. Finally, we define a binary label for the decision to classify each data.

To analyze the use of elapsed time and scancode as features, we analyzed the characteristics based on the distribution of all data, and Fig. 3 shows the distribution results.

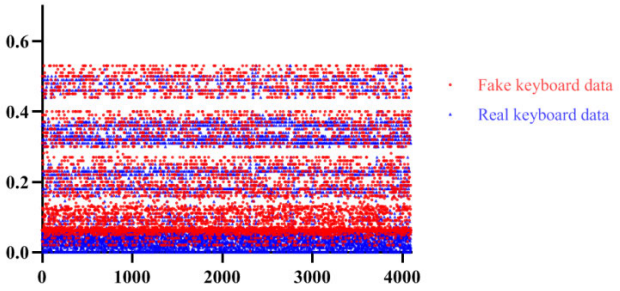

FIGURE 3. Distribution between elapsed time and scancode of fake keyboard data and real keyboard data.

In detail, fake keyboard data has no distribution close to zero, whereas real keyboard data has a distribution close to zero. Consequently, we can measure these distances by converting these values into coordinates, and we hypothesized that it is possible to classify fake data and real data based on distance.

3) PRE-PROCESSING

In the pre-processing phase, distances are measured based on elapsed time and scancode to extend the feature and improve the performance of machine learning classification. In this paper, we used the Manhattan distance and the Euclidean distance to measure the distance. This phase preprocesses the collected data to obtain these distances, which are the distances between the time-scancodes. Namely, we measure the Manhattan distance and the Euclidean distance between time-scancodes.

4) DATA CONFIGURATION

In the dataset configuration phase, datasets are configured for machine learning based on datasets that define the time difference and the scancode as features by extending the features based on the distance obtained from the pre-processing phase, to compare the performance with previous studies. The Manhattan distance involves coordinates according to time and scancode, and the feature defines the Manhattan distance between previous and current coordinates.

Euclidean distance is calculated in two ways: one is by measuring the distance from coordinate (0, 0) to the current coordinate, and the second one is by measuring the distance between the previous and the current coordinates. Therefore, based on time and scancode, six datasets were constructed by defining six features of Manhattan distance between time-scancodes and Euclidean distance between time-scancodes.

5) MACHINE LEARNING

The machine learning phase learns data to classify fake keyboard data and real keyboard data using machine learning models based on six datasets configured in the dataset configuration phase. However, there are various machine learning models, which are difficult to differentiate in terms of which model has the best performance. Therefore, before the experiments, we pondered on the following question: Which machine learning model has the best performance? To achieve this goal, we used machine learning library known as the scikit-learn library. We used various machine learning models, such as K-Nearest Neighbors (KNN) [9], Logistic Regression [10], Decision Tree [11], Random Forest [12], gradient boosting regression tree [12], Support Vector Machine (SVM) [13], and Multi-Layer Perceptrons (MLP) [14] to derive a machine learning model with the best performance. To learn the data, six datasets were trained in each training set, and performances were verified in each validation set.

6) CLASSIFICATION

In the classification phase, the classification performance of fake keyboard data and real keyboard data are evaluated using test sets of six datasets, based on the results learned in the machine learning phase. The indicators for evaluating the performance are accuracy, precision, recall, F1-score, and AUC; and the performance evaluation results of the six features defined in this study are compared with the elapsed time and scancode defined only as features from previous studies, in terms of the increase and decrease rates. To compare the performance, we derive the feature with the best overall performance based on the totals, and averages of increased and decreased performance indicators.

C. FEATURE DEFINITION

This subsection describes the features defined in this paper. Previous studies have defined only the elapsed time and scancode as features, while we define at least four, and up to six, features. Defined features include the scancode, elapsed time, scancode difference, time-scancode Manhattan distance, time-scancode Euclidean distance (i=1), and time-scancode Euclidean distance (i=2).

1) SCANCODE FEATURE

The scancode feature is all collected data, both fake and real keyboard data, where the collected keyboard data is a hexadecimal 1-byte scancode transmitted from the keyboard device, not letters or numbers. Since the range of single-byte numbers in hexadecimal is from (0 to 255), we preprocess it to have a range between (0 and 1) to facilitate the learning of data in the machine learning models. Real keyboard data input from a user is manually input by the authors. Fake keyboard data generated by the defense tool generates a random 1-byte value corresponding to the keyboard scancode using a random number generation function, and the fake data is generated periodically by a timer that is called every 50 ms.

2) ELAPSED TIME FEATURE

The elapsed time feature is the time difference between when the scancode is collected, and the time when the scancode is acquired, in ns units. Since time has an increasing characteristic, it is not suitable to define a feature as a time value. Therefore, we define the difference between the time when the previous keyboard data was obtained, and the time when

the current keyboard data was obtained, as a feature as shown in (1), and use it as the elapsed time feature. The time difference is in ns units, and this feature has a value between (0 and 1), because fake keyboard data is generated every 50 ms.

$$\text{elapsed time} = T_C - T_P \tag{1}$$

3) SCANCODE DISTANCE FEATURE

The scancode distance feature is the distance between the previously collected scancode and the currently collected scancode. It is obtained by subtracting the previously collected scancode value from the currently collected scancode value. The distance value has a value between (0 and 1) because the scancode value has a value between (0 and 1). The value is expressed as a negative number, therefore, to facilitate the data learning for machine learning models, an absolute value is obtained, as shown in (2), to have a positive number, and used as a scancode distance feature. The reason for using this feature is assumed to be to classify data, because the scancodes in the keyboard layout are sequential.

$$\text{scancode distance} = |S_C - S_P| \tag{2}$$

4) TIME-SCANCODE MANHATTAN DISTANCE FEATURE

The time-scancode Manhattan distance feature is based on characteristics that the scancode generated by the defense tool is relatively fixed at 50 ms, while the scancode input from the user is non-periodic in nature. In other words, the scancode generated by the defense tool has a time range of about 50 ms, unless the user inputs the key, therefore, this feature has a specific range for classifying the scancode input from the user. Moreover, the scancode according to the keyboard layout has a characteristic that has a sequential value. Therefore, if these two features are combined and expressed in one location, it is assumed to have a range for classifying fake keyboard data from the real keyboard data. To do this, the elapsed time and the scancode are expressed as X and Y coordinates, respectively, and the distance between the previous coordinate and the current coordinate is measured as the Manhattan distance, as shown in (3) [15]:

$$\text{Manhattan distance (T, S)} = \sum_{i=1}^{n} |Ti - Si| \tag{3}$$

5) TIME-SCANCODE EUCLIDEAN DISTANCE FEATURE

The time-scancode Euclidean distance feature is similar to the time-scancode Manhattan distance feature, however, uses the Euclidean distance as an algorithm for measuring the distance. To measure the Euclidean distance, we calculated the distance in two ways. One way is by measuring the elapsed time and scancode as X and Y coordinates, and calculating the Euclidean distance (i=1) from (0, 0) to the current coordinate. The other way is by measuring the elapsed time and scancode as X and Y coordinates, and calculating the Euclidean distance (i=2) from the previous coordinate to the current coordinate. Therefore, the distance between the previous coordinate and the current coordinate is expressed as Euclidean distance, as shown in (4) [15]:

$$\text{Euclidean distance (T, S)} = \sqrt{\sum_{i=1}^{n} |Ti - Si|^2} \tag{4}$$

III. EXPERIMENT RESULTS

This section describes the datasets by defining the features described in Section 2, and the experimental results of the performance evaluation by using them for the machine learning models. In order to demonstrate the superiority of the proposed attack method, we compare the performance of the proposed method with the performance of the previous studies based on the domains for performance evaluation such as accuracy, precision, recall, F1-score, and AUC.

A. DATASET CONFIGURATION

We constructed six datasets that defined features for each distance combination based on time difference and scancode difference. The defined features are scancode, time difference, scancode difference, time-scancode Manhattan distance, time-scancode Euclidean distance (i=1), and time-scancode Euclidean distance (i=2). Moreover, we also defined features using the datasets used in [8], which used three datasets.

Dataset 1 collected 3,522 pieces of data; the number of real keyboard data was 392, while the number of fake keyboard data was 3,129. Dataset 2 collected 10,022 data; the number of real keyboard data was 1,422, while the number of fake keyboard data is 8,599. Dataset 3 collected 15,046 data; the number of real keyboard data was 2,281, while the number of fake keyboard data was 12,764. Accordingly, we demonstrated six experiment results except for Experiment 1, which was the previous study that defined only elapsed time and scancode as features, hence, 18 datasets were used for experiments. Table 1 shows all the configured datasets.

B. PERFORMANCE EVALUATION FOR EACH FEATURE

In this subsection, we describe the performance evaluation results (Experiments 2-7) of the datasets consisting of six features we defined, including Experiment 1, which is the result from previous study. For performance evaluation, we compared the results of accuracy, precision, recall, F1-score, and AUC from Datasets (1 to 3). Fig. 4 shows the performance evaluation results for each feature.

Specifically, in Dataset 1, KNN significantly lowered all performance results compared to previous studies, while Decision Tree significantly lowered the performance of AUC only. On the other hand, all other models have either similar or higher performance. The accuracy, precision, F1-score, and AUC have significantly the largest performance improvements with Random Forest, while the recall has significantly the largest performance improvement with Gradient Boosting Regression Tree. The accuracy, precision, F1-score, and AUC have the largest significant performance improvements with Random Forest,

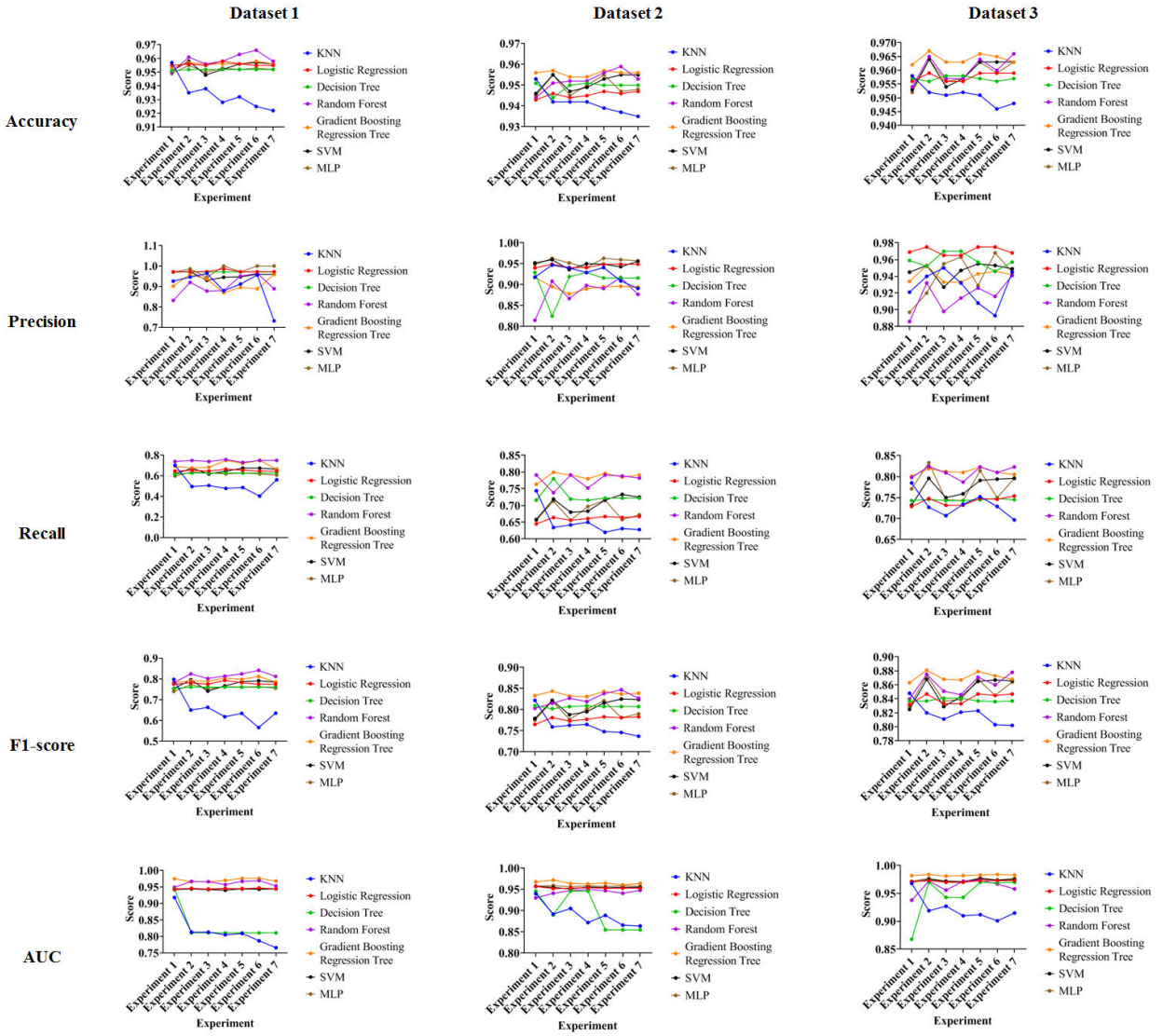

FIGURE 4. Performance evaluation results from Experiments 1 to 7.

while the recall has the largest significant performance with Gradient Boosting Regression Tree. In Dataset 2, KNN had lower accuracy, recall, F1-score, and AUC performances than in the previous studies, except for precision, which significantly decreased in Decision Tree and Gradient Boosting Regression Tree. On the other hand, accuracy significantly improved in Logistic Regression, Random Forest, and SVM, while precision significantly improved in Random Forest. Recall significantly improved in Logistic Regression, Decision Tree, Gradient Boosting Regression Tree, SVM, and MLP, while F1-score significantly improved in Logistic Regression, Random Forest, Gradient Boosting Regression Tree, SVM, and MLP. AUC, the last performance indicator of Dataset 2, shows the biggest improvement in random forest performance. In Dataset 3, KNN has lower accuracy, recall, F1-score, and AUC performance than in the previous studies, except for precision, which improved in all performances compared with previous studies. Accuracy significantly improved in Random Forest, Gradient Boosting Regression Tree, SVM, and MLP, and precision greatly improved in Decision Tree, Random Forest, Gradient Boosting Regression Tree, SVM, and MLP. Recall and F1-score significantly improved in Logistic Regression, Random Forest, Gradient Boosting Regression Tree, and SVM, and AUC greatly improved in Decision Tree and Random Forest.

To prove the superiority of the proposed method, we compared the performance of Experiment 1, experiment from previous study, with the performance of proposed method, based on the increase and decrease rates of best performance. Table 2 shows the comparison results in Dataset 1. The best performance of Experiment 1 in Dataset 1 was 0.957 for

TABLE 1. Configured datasets for the experiments.

Experiment	Dataset	Feature
Exp. 1	• Dataset 1 (3,522, 392/3,129) • Dataset 2 (10,022, 1,422/8,599) • Dataset 3 (15,046, 2,281/12,764)	• Scancode • Elapsed time
Exp. 2	• Dataset 1 (3,522, 392/3,129) • Dataset 2 (10,022, 1,422/8,599) • Dataset 3 (15,046, 2,281/12,764)	• Scancode • Elapsed time • Scancode distance • Manhattan distance between time-scancodes
Exp. 3	• Dataset 1 (3,522, 392/3,129) • Dataset 2 (10,022, 1,422/8,599) • Dataset 3 (15,046, 2,281/12,764)	• Scancode • Elapsed time • Scancode distance • Euclidean distance between time-scancodes (i=1)
Exp. 4	• Dataset 1 (3,522, 392/3,129) • Dataset 2 (10,022, 1,422/8,599) • Dataset 3 (15,046, 2,281/12,764)	• Scancode • Elapsed time • Scancode distance • Euclidean distance between time-scancodes (i=2)
Exp. 5	• Dataset 1 (3,522, 392/3,129) • Dataset 2 (10,022, 1,422/8,599) • Dataset 3 (15,046, 2,281/12,764)	• Scancode • Elapsed time • Scancode distance • Manhattan distance between time-scancodes • Euclidean distance between time-scancodes (i=1)
Exp. 6	• Dataset 1 (3,522, 392/3,129) • Dataset 2 (10,022, 1,422/8,599) • Dataset 3 (15,046, 2,281/12,764)	• Scancode • Elapsed time • Scancode distance • Manhattan distance between time-scancodes • Euclidean distance between time-scancodes (i=2)
Exp. 7	• Dataset 1 (3,522, 392/3,129) • Dataset 2 (10,022, 1,422/8,599) • Dataset 3 (15,046, 2,281/12,764)	• Scancode • Elapsed time • Scancode distance • Manhattan distance between time-scancodes • Euclidean distance between time-scancodes (i=1) • Euclidean distance between time-scancodes (i=2)

accuracy, 0.972 for precision, 0.738 for recall, 0.798 for F1-score, and 0.975 for AUC.

In detail, the accuracy showed the highest improvement in all the other experiments except Experiment 3, and the performance of Random Forest in Experiment 6 is greatly improved. Precision showed the best performance in all experiments, and the performances of MLP in Experiments 4, 6, and 7 showed the greatest improvement. Recall had the best performance improvement in all the other experiments except Experiment 5, and the performance of Random Forest in experiments showed the best improvement. F1-score had the best performance in all experiments, and the performance of Random Forest in Experiment 6 was greatly improved. Finally, AUC showed decrease in performances in all other experiments except Experiments 5 and 6, and the performances of Gradient Boosting Regression Tree in Experiments 5 and 6 showed the best performance improvements. Overall, the ratios of the Random Forest model are the highest in accuracy, recall, and F1-score, while precision and AUC have high ratios of MLP and Gradient Boosting Regression Tree models, respectively.

Table 3 shows the comparison results in Dataset 2. The best performance of experiment 1 in Dataset 2 has an accuracy of 0.956, precision of 0.952, recall of 0.791, F1-score of 0.833, and AUC of 0.968.

Specifically, accuracy has the best performance improvements in all of the other experiments except Experiments 2 and 3, and the performance of Random Forest in Experiment 6 was greatly improved. Precision showed the best performance in all the other experiments except Experiment 4, and the performances of MLP in Experiments 2 and 5 showed significant improvements. Recall had the best performance improvements in all the other experiments, except Experiments 4 and 6, and the performance of Gradient Boosting Regression Tree in Experiment 2 was greatly improved.

F1-score had the highest performance improvement in all other experiments, except Experiments 3 and 4, and the performance of Random Forest in Experiment 6 had the biggest improvement. Finally, AUC showed the highest performance decrease in all other experiments, except Experiment 2, and the performance of Gradient Boosting Regression Tree in Experiment 2 was greatly improved. Overall, the ratios of Gradient Boosting Regression Tree are the highest in accuracy, recall, F1-score, and AUC, and precision has high ratios of MLP model.

Table 4 show the comparison results in Dataset 3. The best performance of Experiment 1 in Dataset 3 had an accuracy of 0.962, precision of 0.969, recall of 0.801, F1-score of 0.863, and AUC of 0.982:

In detail, accuracy showed the best performances in all the experiments, and the performance of Gradient Boosting Regression Tree in Experiment 2 showed the greatest improvement. Precision had the highest performance improvements in all the other experiments, except Experiment 7, and the performances of Logistic Regression in Experiments 2, 5, and 6 show the greatest performance improvements. Recall and F1-score had the best performance improvements in all experiments, and the performance of MLP in Experiment 2 showed great improvement. Finally, AUC showed decrease in performances in all other experiments, except Experiment 3, and the performances of Gradient Boosting Regression Tree in Experiments 2 and 6 had the biggest improvements. Overall, the ratios of Gradient Boosting Regression Tree were highest in accuracy, recall, F1-score, and AUC; and precision had a high ratio in Logistic Regression.

C. PERFORMANCE COMPARISON OF EACH FEATURE ACCORDING TO INCREASE AND DECREASE

In this subsection, we describe the performance comparison of the datasets with six defined features based on

TABLE 2. Comparison of the best increase and decrease performances of previous research and the proposed method (Dataset 1).

Experiment	Accuracy			Precision			Recall			F1-score			AUC		
	M	B	+/−	M	B	+/−	M	B	+/−	M	B	+/−	M	B	+/−
Exp. 2	R	0.961	0.004	M	0.986	0.014	R	0.748	0.01	R	0.825	0.027	R	0.967	−0.008
Exp. 3	R, G	0.956	−0.001	L	0.972	0	R	0.738	0	R	0.802	0.004	R, G	0.966	−0.009
Exp. 4	L, R	0.958	0.001	M	1	0.028	R	0.757	0.019	R	0.814	0.016	G	0.97	−0.005
Exp. 5	R	0.963	0.006	L	0.972	0	R	0.729	−0.009	R	0.825	0.027	G	0.976	0.001
Exp. 6	R	0.966	0.009	M	1	0.028	R, G	0.748	0.01	R	0.842	0.044	G	0.976	0.001
Exp. 7	R	0.958	0.001	M	1	0.028	R	0.748	0.01	R	0.812	0.014	G	0.968	−0.007

Ex: Experiment; M: Model; B: Best score; +/−: Increase or decrease score; K: KNN; L: Logistic Regression; D: Decision Tree; R: Random Forest; G: Gradient Boosting Regression Tree; S: SVM; M: MLP

TABLE 3. Comparison of the best increase and decrease of performances of the previous research and the proposed method (Dataset 2).

Experiment	Accuracy			Precision			Recall			F1-score			AUC		
	M	B	+/−	M	B	+/−	M	B	+/−	M	B	+/−	M	B	+/−
Exp. 2	G	0.957	0.001	M	0.963	0.011	G	0.799	0.008	G	0.844	0.011	G	0.972	0.004
Exp. 3	G	0.954	−0.002	M	0.952	0	R, G	0.791	0	G	0.832	−0.001	G	0.964	−0.004
Exp. 4	G	0.954	−0.002	S	0.95	−0.002	G	0.78	−0.011	G	0.831	−0.002	G	0.963	−0.005
Exp. 5	G	0.957	0.001	M	0.963	0.011	G	0.796	0.005	G	0.843	0.01	G	0.965	−0.003
Exp. 6	R	0.959	0.003	M	0.96	0.008	R	0.788	−0.003	R	0.847	0.014	G	0.961	−0.007
Exp. 7	G	0.956	0	M	0.957	0.005	G	0.791	0	G	0.839	0.006	G	0.964	−0.004

TABLE 4. Comparison of the best increase and decrease performances of the previous research and the proposed method (Dataset 3).

Experiment	Accuracy			Precision			Recall			F1-score			AUC		
	M	B	+/−	M	B	+/−	M	B	+/−	M	B	+/−	M	B	+/−
Exp. 2	G	0.967	0.005	L	0.975	0.006	M	0.833	0.032	G	0.881	0.018	G	0.984	0.002
Exp. 3	G	0.963	0.001	D	0.97	0.001	G	0.812	0.011	G	0.868	0.005	G	0.981	−0.001
Exp. 4	G	0.963	0.001	D	0.97	0.001	G	0.81	0.009	G	0.867	0.004	G	0.982	0
Exp. 5	G	0.966	0.004	L	0.975	0.006	R, G	0.823	0.022	G	0.8789	0.0159	G	0.983	0.001
Exp. 6	G	0.965	0.003	L	0.975	0.006	R, G	0.81	0.009	G	0.873	0.01	G	0.984	0.002
Exp. 7	R	0.966	0.004	L	0.968	−0.001	R, G	0.823	0.022	R	0.878	0.015	G	0.983	0.001

the previous research experiment results, and compare the performance based on the number of increased items in Datasets (1 to 3) based on accuracy, precision, recall, F1-score, and AUC. Fig. 5 shows the results of the performance increase and decrease for each feature.

Specifically, based on the Experiment 1 as the previous study, we marked the increasing items in blue, and the decreasing items in red. Experiment 2 had the greatest number of increased items while Experiment 3 had the greatest number of decreased items. Comparing the performance increase and decrease in accordance with the datasets, except for Experiment 4, all other experiments had the smallest decrease of items and the highest increase of items in Dataset 3. Namely, the proposed method was most effective in Dataset 3. The next best performance increase and decrease dataset is Dataset 1, except for Experiment 4, while the worst performance increase and decrease dataset was Dataset 2. Finally, the most unchanged experiment was Experiment 3; this presents that the number of items, which are marked in yellow, is largest.

For a more detailed comparison of the performance increase and decrease, Table 5 shows the number of items according to the increase and decrease of the performance by datasets consisting of six defined features. Experiment 2 had 74 increased items, and 27 decreased items, which mean that Experiment 2 had the best performance. On the other hand, Experiment 3 had 52 increased items and 37 decreased items, which mean that Experiment 3 showed the worst performance.

TABLE 5. Comparison results of the number of increased and decreased performance items for each experiment.

Experiment	+/−	Dataset 1	Dataset 2	Dataset 3	Total
Exp. 2	Decrease	9	12	6	27
	0	3	0	1	4
	Increase	23	23	28	74
Exp. 3	Decrease	12	15	10	37
	0	9	4	3	16
	Increase	14	16	22	52
Exp. 4	Decrease	10	13	10	33
	0	1	5	3	9
	Increase	24	17	22	63
Exp. 5	Decrease	10	14	6	30
	0	2	1	2	5
	Increase	23	20	27	70
Exp. 6	Decrease	8	15	9	32
	0	5	1	0	6
	Increase	22	19	26	67
Exp. 7	Decrease	10	15	6	31
	0	5	1	2	8
	Increase	20	19	27	66

Performance comparison results between experiment 1 and experiment 2

Model	Dataset 1					Dataset 2					Dataset 3				
	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC
KNN	-0.022	0.02	-0.206	-0.148	-0.105	-0.011	0.029	-0.11	-0.063	-0.048	-0.006	0.019	-0.058	-0.028	-0.049
Linear Regression	0.001	0	0.009	0.007	0.003	0.003	0.009	0.019	0.016	-0.005	0.003	0.006	0.019	0.015	0.002
Decision Tree	0.001	0	0.009	0.007	-0.134	-0.007	-0.104	0.064	-0.007	-0.056	-0.001	-0.007	0.003	0	0.102
Random Forest	0.011	0.088	0.01	0.043	0.018	0.007	0.093	-0.053	0.012	0.011	0.011	0.046	0.026	0.035	0.034
Gradient Boosting	0.004	0.058	-0.019	0.008	-0.009	0.001	-0.022	0.036	0.011	0.004	0.005	0.019	0.018	0.018	0.002
SVM	0.007	0.002	0.056	0.042	0.002	0.009	0.008	0.061	0.043	-0.002	0.011	0.008	0.064	0.043	0.005
MLP	0.006	0.016	0.038	0.033	0	0.01	0.015	0.057	0.045	0.001	0.012	0.023	0.062	0.045	0.005

Performance comparison results between experiment 1 and experiment 3

Model	Dataset 1					Dataset 2					Dataset 3				
	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC
KNN	-0.019	0.038	-0.196	-0.135	-0.105	-0.011	0.022	-0.102	-0.059	-0.035	-0.007	0.029	-0.078	-0.037	-0.041
Linear Regression	0	0	0	0	0	0.001	0.001	0.011	0.008	-0.004	0	-0.004	0.003	0.001	-0.001
Decision Tree	0.001	0	0.009	0.007	-0.134	-0.001	-0.01	0.003	-0.002	0	0.001	0.011	0	0.004	0.075
Random Forest	0.006	0.046	0	0.02	0.017	0.008	0.052	0	0.024	0.017	0.003	0.012	0.011	0.011	0.018
Gradient Boosting	0.003	0.034	-0.01	0.006	-0.009	-0.002	-0.039	0.028	-0.001	-0.004	0.001	-0.001	0.011	0.005	-0.001
SVM	-0.003	-0.041	0	-0.012	0	0.001	-0.016	0.022	0.009	-0.006	0.001	-0.018	0.018	0.004	0.002
MLP	0.001	-0.026	0.028	0.013	-0.001	0	0.004	0	0.002	-0.002	0.004	0.058	-0.026	0.008	0

Performance comparison results between experiment 1 and experiment 4

Model	Dataset 1					Dataset 2					Dataset 3				
	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC
KNN	-0.029	-0.047	-0.224	-0.18	-0.113	-0.011	0.011	-0.094	-0.057	-0.068	-0.006	0.011	-0.051	-0.027	-0.058
Linear Regression	0.003	0.014	0.019	0.018	0.002	0.002	0.001	0.016	0.012	-0.003	0	-0.004	0.003	0.001	-0.001
Decision Tree	0.001	0	0.009	0.007	-0.134	0	0	0	0	0	0.001	0.011	0	0.004	0.075
Random Forest	0.008	0.048	0.019	0.032	0.008	0.008	0.083	-0.039	0.016	0.02	0.003	0.028	-0.011	0.006	0.034
Gradient Boosting	0.003	-0.032	0.056	0.021	-0.005	-0.002	-0.027	0.017	-0.002	-0.005	0.001	-0.001	0.009	0.004	0
SVM	0.001	-0.026	0.028	0.013	-0.002	0.003	-0.002	0.025	0.016	-0.001	0.004	0.002	0.027	0.018	0.001
MLP	0.004	0.03	0.019	0.023	0.001	0.005	-0.007	0.041	0.026	0.001	0.005	0.066	-0.028	0.01	-0.001

Performance comparison results between experiment 1 and experiment 5

Model	Dataset 1					Dataset 2					Dataset 3				
	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC
KNN	-0.025	-0.014	-0.215	-0.164	-0.109	-0.014	0.023	-0.124	-0.074	-0.051	-0.007	-0.013	-0.033	-0.025	-0.056
Linear Regression	0.001	0	0.009	0.007	0.002	0.004	0.009	0.022	0.018	-0.004	0.003	0.006	0.019	0.015	0.002
Decision Tree	0.001	0	0.009	0.007	-0.134	-0.001	-0.013	0.006	-0.002	-0.091	0	-0.002	0.002	0	0.102
Random Forest	0.013	0.119	-0.009	0.043	0.018	0.012	0.076	0	0.035	0.017	0.01	0.04	0.025	0.031	0.036
Gradient Boosting	0.003	-0.007	0.028	0.015	0.001	0.001	-0.022	0.033	0.01	-0.003	0.004	0.009	0.022	0.016	0.001
SVM	0.005	-0.024	0.056	0.033	0.003	0.007	-0.003	0.058	0.037	-0.003	0.01	0.01	0.059	0.04	0.006
MLP	0.003	0.001	0.028	0.021	-0.002	0.01	0.015	0.06	0.046	-0.001	0.011	0.032	0.043	0.039	0.006

Performance comparison results between experiment 1 and experiment 6

Model	Dataset 1					Dataset 2					Dataset 3				
	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC
KNN	-0.032	0.03	-0.299	-0.232	-0.131	-0.016	-0.009	-0.113	-0.076	-0.074	-0.012	-0.028	-0.056	-0.045	-0.067
Linear Regression	0	0	0	0	0.004	0.003	0.009	0.019	0.016	-0.004	0.003	0.006	0.017	0.013	0.002
Decision Tree	0.001	0	0.009	0.007	-0.134	-0.001	-0.013	0.006	-0.002	-0.091	-0.001	-0.013	0.005	-0.001	0.101
Random Forest	0.016	0.132	0.01	0.06	0.02	0.015	0.102	-0.003	0.044	0.011	0.006	0.03	0.012	0.02	0.029
Gradient Boosting	0.005	-0.013	0.056	0.029	0.001	0	-0.021	0.022	0.004	-0.007	0.003	0.012	0.009	0.01	0.002
SVM	0.006	-0.011	0.056	0.037	0.001	0.009	-0.009	0.075	0.046	-0.002	0.01	0.008	0.062	0.042	0.004
MLP	0.004	0.03	0.019	0.023	-0.001	0.002	0.012	0.002	0.006	0.001	0.007	0.071	-0.021	0.016	0.002

Performance comparison results between experiment 1 and experiment 7

Model	Dataset 1					Dataset 2					Dataset 3				
	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC	Accuracy	Precision	Recall	F1-score	AUC
KNN	-0.035	-0.194	-0.14	-0.163	-0.152	-0.018	-0.027	-0.116	-0.085	-0.076	-0.01	0.024	-0.088	-0.046	-0.053
Linear Regression	0	0	0	0	0.001	0.004	0.009	0.022	0.018	-0.004	0.003	-0.001	0.025	0.015	0.002
Decision Tree	0.001	0	0.009	0.007	-0.134	-0.001	-0.013	0.006	-0.002	-0.091	0	-0.002	0.002	0	0.102
Random Forest	0.008	0.057	0.01	0.03	0.004	0.009	0.062	-0.009	0.024	0.018	0.012	0.055	0.025	0.038	0.02
Gradient Boosting	0.003	0.057	-0.028	0.002	-0.007	0	-0.023	0.028	0.006	-0.004	0.001	0.008	0.004	0.005	0.001
SVM	0.005	-0.012	0.047	0.031	0.002	0.009	0.004	0.067	0.045	-0.002	0.01	0.004	0.064	0.041	0.005
MLP	0.003	0.03	0.009	0.016	-0.001	0.003	0.009	0.016	0.015	-0.001	0.011	0.048	0.025	0.035	0.005

FIGURE 5. Performance increase and decrease comparison results for each feature.

Comparing the number of increased and decreased items for each dataset, in Dataset 1, Experiments 2 and 6 both had 23 increased items, and 9 and 8 decreased items, respectively, and were the best performances. On the other hand,

Experiment 3 had 14 increased items and 12 decreased items, which was the worst performance. In Dataset 2, Experiment 2 had 23 increased items and 12 decreased items, which was the best performance. On the other hand, Experiments 6 and 7 had 19 increased items and 15 decreased items, which were the worst performance. Finally, in Dataset 3, Experiment 2 had 28 increased items and 6 decreased items, which was the best performance. On the other hand, Experiments 3 and 4 had 22 increased items and 10 decreased items, which were the worst performance.

Evaluating the datasets according to the number of increased and decreased items, Dataset 3 had 152 increased items and 47 reduced items, which was the best performance. On the other hand, Dataset 2 had 114 increased items and 84 decreased items, which was the worst performance.

D. OVERALL PERFORMANCE INCREASE AND DECREASE COMPARISON RESULTS

In this subsection, we compare the overall performance of datasets consisting of six features, based on the experiment results of previous research. The performance was compared based on the sum of the decreased scores, the sum of the increased scores, the sum of the increased average, and the sum of the decreased average. Fig. 6 shows the comparison results of the overall performance increase and decrease according to each experiment.

Specifically, we described the comparison results of the overall performance increase and decrease of the datasets according to each feature. The experiment with the highest decrease in Dataset 1 was Experiment 7, with −0.866. The experiment with the highest decrease in Dataset 2 was Experiment 2, with −0.488. The experiment with the highest decrease in Dataset 3 was Experiment 6, with -0.244. The experiments with the highest average decrease in all datasets were Experiments 6 and 7, both with an average of −0.51267. Thus, Experiments 6 and 7 with the highest decrease average showed the worst performance. The experiment with the lowest decrease in Dataset 1 was Experiment 2, with −0.623. The experiment with the lowest decrease in Dataset 2 was Experiment 3, with -0.294. The experiment with the lowest decrease in Dataset 3 was Experiment 5, with −0.136. The experiment with the lowest average of decrease in all datasets was Experiment 3, with an average of 0.399. Therefore, Experiment 3, which had the lowest average of decrease, showed the best performance.

The experiment with the highest increase in Dataset 1 was Experiment 6, with 0.556. The experiment with the highest increase in Datasets 2 and 3 was Experiment 2, with 0.564 and 0.661, respectively. The experiment with the highest average of increase in all datasets was Experiment 2 with an average of 0.568. Thus, Experiment 2 with the highest average of increase showed the best performance. The experiment with the lowest increase in Dataset 1 was Experiment 7, with 0.332. The experiment with the lowest increase in Datasets 2 and 3 was Experiment 3, with 0.213 and 0.291, respectively. The experiment with the lowest average of increase in all datasets was Experiment 3, with an average of 0.327. Therefore, Experiment 3, which had the lowest average of increase, showed the worst performance.

TABLE 6. Overall increase and decrease IN numbers.

Experiment	Dataset 1	Dataset 2	Dataset 3	Average
Exp. 2	−0.144	0.076	0.512	0.148
Exp. 3	−0.212	−0.081	0.077	−0.072
Exp. 4	−0.405	−0.015	0.136	−0.094
Exp. 5	−0.277	0.093	0.463	0.093
Exp. 6	−0.297	−0.037	0.258	−0.025
Exp. 7	−0.534	−0.098	0.39	−0.08

The experiment with the highest average decrease in Dataset 1 was Experiment 6, with an average of −0.106. The experiment with the highest average decrease in Dataset 2 was Experiment 2, with an average of −0.04. The experiment with the highest average decrease in Dataset 3 was Experiment 7, with an average of −0.033. The experiment with the highest average decrease in all datasets was Experiment 6, with an average of −0.054. Thus, Experiment 6 showed the worst performance. The experiment with the lowest average decrease in Dataset 1 was Experiment 3, with an average of −0.057. The experiment with the lowest average decrease in Dataset 2 was Experiment 3, with an average of −0.019. The experiment with the lowest average decrease in Dataset 3 was Experiment 4, with an average of −0.018. The experiment with the lowest average decrease in all datasets was Experiment 3, with an average of −0.032. Therefore, Experiment 3 showed the best performance.

The experiment with the highest average increase in Dataset 1 was Experiment 6, with an average of 0.025. The experiment with the highest average increase in Dataset 2 was Experiment 5, with an average of 0.024. The experiment with the highest average increase in Dataset 3 was Experiment 2, with an average of 0.023. The experiment with the highest average increase in all datasets was Experiment 2, with an average of 0.022. Thus, Experiment 2 showed the best performance. The experiment with the lowest average increase in Dataset 1 was Experiment 4, with an average of 0.016. The experiment with the lowest average increase in Datasets 2 and 3 was Experiment 3, with an average of 0.013. The experiment with the lowest average of increase in all datasets was Experiment 3, with an average of 0.014. Therefore, Experiment 3 showed the worst performance.

Fig. 7 and Table 6 compare the results of the total increase and decrease numbers based on the sum of the increased scores, the sum of the increased average, and the sum of the decreased average.

Finally, we derived the feature with the best performance by comparing the overall increased and decreased numbers of datasets according to each feature. The experiment with the highest total decreased numbers in Dataset 1 was Experiment 7, with −0.534. The experiment with the highest total decreased numbers in Dataset 2 was Experiment 7, with −0.098. The experiment with the highest total decreased

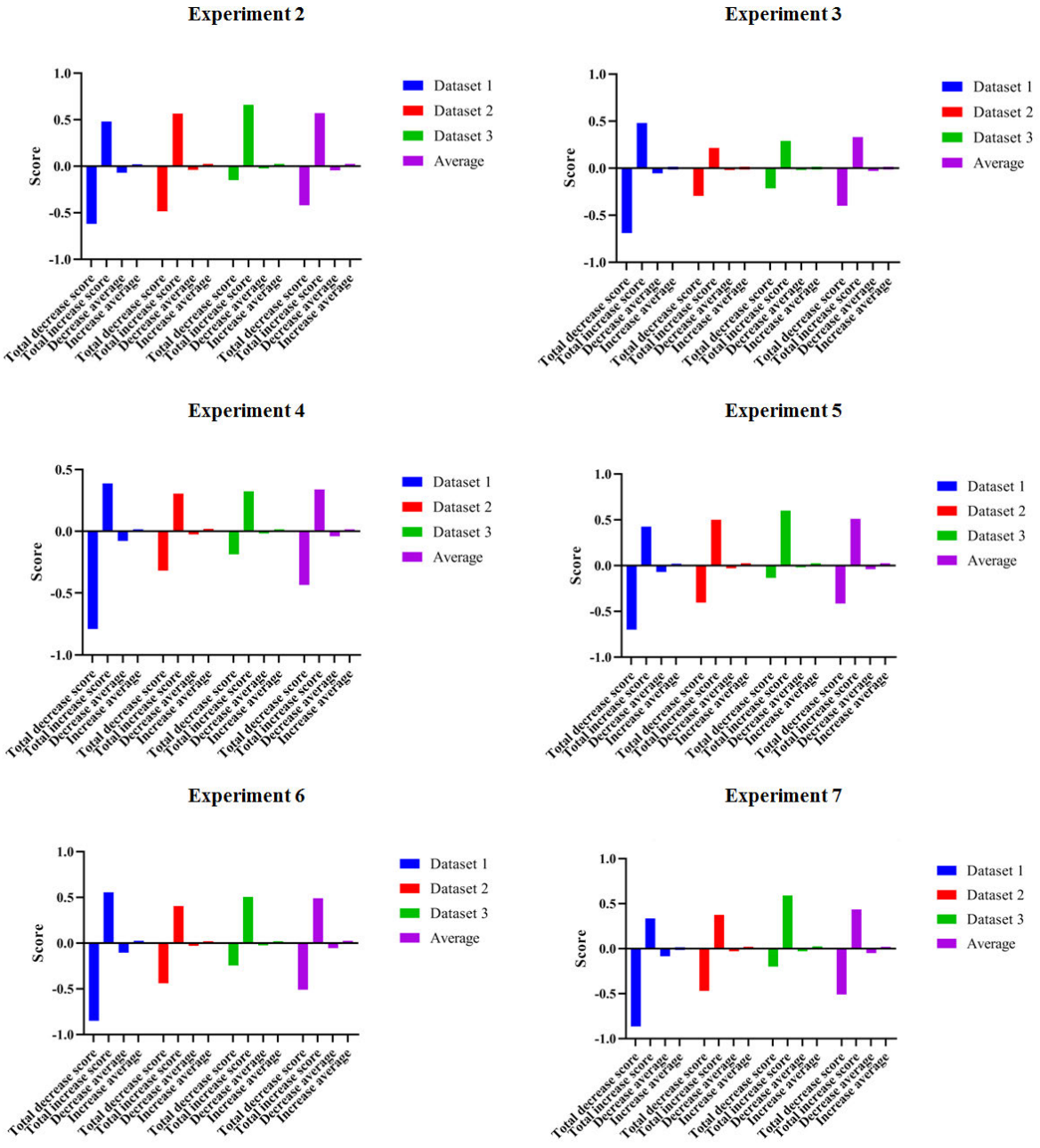

FIGURE 6. Comparison results of the overall performance increase and decrease according to each experiment.

numbers in Dataset 3 was Experiment 3, with −0.077. The experiment with the highest decreased average in all datasets was Experiment 7, with an average of −0.08. Therefore, Experiment 7, which defined all the six features, namely, scancode, elapsed time, scancode distance, Manhattan distance between time-scancode, Euclidean distance between time-scancode (i=1), and Euclidean distance between time-scancode (i=2), showed the worst performance.

On the other hand, the experiment with the highest total increase in Dataset 1 was Experiment 2, with -0.144. The experiment with the highest total increase in Dataset 2 was Experiment 5, with 0.093. The experiment with the highest total increase in Dataset 3 was Experiment 2, with 0.512. The experiment with the highest increase average in all datasets was Experiment 2, with an average of 0.148.

Therefore, Experiment 2, which defined four features, namely, scancode, elapsed time, scancode distance, and Manhattan distance between time-scancode, showed the best performance. The second-best experiment was Experiment 5, with an increase average of 0.093. Experiment 5, which defined five features, namely, scancode, elapsed time, scancode distance, Manhattan distance between time-scancode,

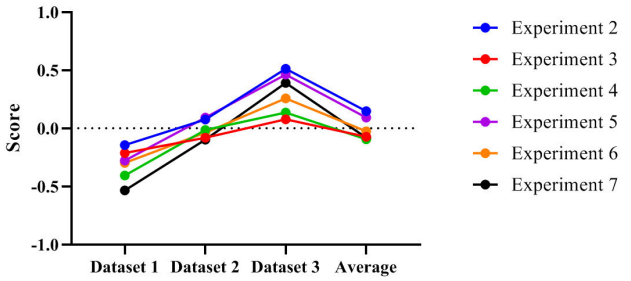

FIGURE 7. Comparison results according to the total increase and decrease in numbers.

and Euclidean distance between time-scancode (i=1), had improved overall performance, compared to previous studies. In conclusion, only two of the six features defined in this paper increase the overall performance. This means that using our proposed attack technique, we can more effectively steal user passwords.

The limitations of the proposed attack technique are as follows. In terms of internal validity, the proposed attack technique needs to access the keyboard controller as hardware to collect keyboard data, and also, a device driver running in kernel mode is required for this access. In terms of external validity, an attacker must transmit data to the outside in order to learn the collected data.

Finally, we compare and evaluate the performance of the proposed attack method with the latest attack technologies using hardware attack and machine learning. One of the recent studies that compares to the proposed attack goal is acoustic side channel attack, one of the side-channel attacks. This attack technique is used when the victim inputs a key using a keyboard device, and then an attacker guesses the pressed key based on sound information. One of the methods for collecting sound information is through a microphone of attached devices, such as PC, laptop, and smartphone, while using VoIP such as Skype [28] or Hangout. Various machine learning models are used to enhance the accuracy of pressed key information based on the collected sound information. Therefore, we showed the results of the performance comparison between the proposed attack technology and the latest attack technologies as in Table 7. The following indicators are presented for purposes of performance comparison; accuracy, accuracy increase/decrease (+/−), and the required conditions and the attacker's knowledge.

Accuracy, which is a performance evaluation indicator, refers to the best accuracy for each research, targeting the lowercase alphabetic key (a to z) for all existing attack technologies, and the increase or decrease (+/−) refers to the increase or decrease of the proposed method, and condition and knowledge mean the attacker's knowledge such as typing style.

In [27], it is based the accuracy using the time-frequency technique according to the typing styles. The highest accuracy of the hint and peck typing style is up to 64%, and the best accuracy of the touch-typing style is up to 40%. Both figures are less accurate than the proposed attack method.

TABLE 7. Performance comparison and evaluation results with the latest keyboard data stealing researches (Alphabet keys (a-z)), Single key classification).

	Technique (Algorithms)		Accuracy	+/-	Condition and knowledge)
[27]	time–frequency technique		64% 40%	+32.70% +56.70%	the victim's typing style (hunt and peck)
[28]	Logistic regression		91.7%	+5.00%	the victim's typing style (touch typing)
[29]	time-domain and frequency-domain distance estimates	Frequency Time	67.30%	+29.40%	
	using FFT coefficients	Simple Logistic Regression	39.54%	+57.16%	the victim's typing style (hunt and peck)
		Multinomial Logistic Regression	38.89%	+57.81%	
	using MFCC	SMO	74.33%	+22.37%	
		Simple Logistic Regression	73.17%	+23.53%	
Proposed	Time-distance based features		96.7%	-	All keys, no typing style

In [28], only Logistic Regression, which is the model with highest accuracy among various machine learning models such as Logistic Regression, Linear Discriminant Analysis, Support Vector Machine, and Random Forest, is shown, to have the highest accuracy of about 91.7%. This accuracy is not significantly different from the proposed attack method, but there are requirements for this attack method. [28] assumes that the attacker knows all of the victim's typing styles and keyboard models, and used Top-5 Accuracy as a performance evaluation indicator. Top-5 Accuracy is mainly used for image data, and is not suitable in a case of keyboard data, since each key is important for keyboard information.

[29] is based on the transformed audio information using three signal processing technologies, namely, time-domain and frequency -domain distance estimates, FFT coefficients, and MFCC, and the accuracy is derived using machine learning models such as J48, Random Forest, Linear Nearest Neighbor Search, SMO, Simple Logistic Regression, and Multinomial Logistic Regression. The best accuracy for each signal processing technology is about 67.3% for time-domain and frequency-domain distance estimates, 39.54% for FFT coefficients, and 74.33% for MFCC. Among the three technologies, MFCC has the best performance, but the accuracy of all three technologies is lower compared to that of the proposed attack method.

Finally, in order to evaluate the performance of the proposed attack method and the latest attack technologies, the increase and decrease values were compared based on the highest accuracy of the proposed method. Compared to the study in [27], the proposed method improved the performance of 32.70%, and 56.70%, respectively, and the average of 44.7% improved the accuracy performance. Compared to

the [29], the proposed method improved the performance of 29.40%, 57.16%, 57.81%, 22.37%, and 23.53%, respectively, with the average improvement being 38.05%. Overall, the attack method proposed in this paper has an average accuracy of about 29.75%, which is an improvement from the existing acoustic side channel attacks. When comprehensively comparing and evaluating all technologies, the results show a performance improvement of at least 5.45% and a maximum of 148.65%, which means that the proposed attack method steals keyboard data more effectively compared to latest keyboard attack technologies. Moreover, latest keyboard attack technologies have limitations in terms of their attack success due to various conditions. In the case of acoustic side channel attacks, physical access is required, in addition to the numerous assumptions made, such as the victim's typing style and keyboard product model. In addition, since all technologies use machine learning models, datasets make for important researches, but there are also limitations in dataset configuration due to the variety of the attack target's type styles and keyboard product models.

IV. CONCLUSION

In this study, we evaluated the security of keyboard data, which is authentication information in password authentication, by defining features to improve the accuracy of existing attack techniques using machine learning models. Previous attack technology effectively classified fake keyboard data generated by the defender and real keyboard data input from the user by defining the elapsed time and scancode as features, with a best accuracy of 96.2 %. In this study, we analyzed the distribution of data based on the distance of time and scancode, which are core data of previous study, to increase the attack success rate more effectively, and derived features based on time-scancode distance with higher attack success rate than previous studies. The proposed attack method defined six features, and evaluated the performance based 18 datasets.

We evaluated the performance by comparing the performance of each feature, the best performance increase and decrease comparison, the performance comparison of each feature according to increase and decrease, and the overall performance increase and decrease comparison. The performance evaluation for each feature and the best performance increase and decrease comparison result presented improvements of performance, and the best performances such as accuracy, precision, recall, F1-score, and AUC, for all features defined in this paper.

The performance comparison result of each feature according to the increase and decrease showed Experiment 2 with the most increased items, and Experiment 3 with the most decreased items. Experiment 2 had 74 increased items and 27 decreased items, which was the best performance. On the other hand, Experiment 3 had 52 increase items and 37 decreased items, which was the worst performance. Overall performance increase and decrease comparison results showed that Experiment 2 had the best increase average in all datasets, with an average of 0.148. Therefore, in our approaches, Experiment 2, which defined four features, namely, such as scancode, elapsed time, scancode distance, and Manhattan distance between time-scancode, had the best performance.

Moreover, compared with the previous studies, two of the six features defined in this paper increased the overall performance. This means that we can more effectively steal password input from users by using our proposed attack method in terms of offensive security. Consequently, we consider that this paper has the novelty of improving the performance of the attack success rates.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

REFERENCES

[1] A. Salem and M. S. Obaidat, "A novel security scheme for Behavioral authentication systems based on keystroke dynamics," *Secur. Privacy*, vol. 2, no. 2, p. e64, Mar. 2019.
[2] S. ShanmugaPriya, A. Valarmathi, and D. Yuvaraj, "The personal authentication service and security enhancement for optimal strong password," *Concurrency Comput., Pract. Exper.*, vol. 31, no. 14, Jul. 2019, Art. no. e5009.
[3] D. Patel, H. Jiang, J. Patel, and B. Kaminska, "Versatile authentication using an identifier based on optical variable nanostructures," *IEEE Trans. Nanotechnol.*, vol. 18, pp. 1079–1088, Sep. 2019.
[4] K. Lee and K. Yim, "Keyboard security: A technological review," in *Proc. 5th Int. Conf. Innov. Mobile Internet Services Ubiquitous Comput.*, Jun. 2011, pp. 9–15.
[5] S. Lee, K. Lee, and K. Yim, "Security assessment of keyboard data: Based on Kaspersky product," in *Proc. BWCCA*, Nov. 2016, pp. 395–400.
[6] K. Lee, K. Bae, and K. Yim, "Hardware approach to solving password exposure problem through keyboard sniff," *Int. J. Electr. Comput. Eng.*, vol. 3, no. 8, pp. 1501–1503, Aug. 2009.
[7] K. Lee, Y. Choi, H. Yeuk, and K. Yim, "Password sniff by forcing the keyboard to replay scan codes," in *Proc. JWIS*, Guangzhou, China, Aug. 2010, pp. 9–11.
[8] K. Lee and K. Yim, "Cybersecurity threats based on machine learning-based offensive technique for password authentication," *Appl. Sci.*, vol. 10, no. 4, p. 1286, Feb. 2020.
[9] M.-L. Zhang and Z.-H. Zhou, "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognit.*, vol. 40, no. 7, pp. 2038–2048, Jul. 2007.
[10] W. Cheng and E. Hüllermeier, "Combining instance-based learning and logistic regression for multilabel classification," *Mach. Learn.*, vol. 76, nos. 2–3, pp. 211–225, Jul. 2009.
[11] C. Sinclair, L. Pierce, and S. Matzner, "An application of machine learning to network intrusion detection," in *Proc. 15th Annu. Comput. Secur. Appl. Conf. (ACSAC)*, Dec. 1999, pp. 6–10.
[12] R. E. Banfield, L. O. Lawrence, K. W. Bowyer, and W. P. Kegelmeyer, "A comparison of decision tree ensemble creation techniqu," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 173–180, Nov. 2006.
[13] S. U. Jan, Y.-D. Lee, J. Shin, and I. Koo, "Sensor fault classification based on support vector machine and statistical time-domain features," *IEEE Access*, vol. 5, pp. 8682–8690, 2017.
[14] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.
[15] Y. Huang, W. Jin, B. Li, P. Ge, and Y. Wu, "Automatic modulation recognition of radar signals based on manhattan distance-based features," *IEEE Access*, vol. 7, pp. 41193–41204, 2019.
[16] T. Kim and Y. Shin, "Reinforcing meltdown attack by using a return stack buffer," *IEEE Access*, vol. 7, pp. 186065–186077, 2019.
[17] M. D. Hill, J. Masters, P. Ranganathan, P. Turner, and J. L. Hennessy, "On the spectre and meltdown processor security vulnerabilities," *IEEE Micro*, vol. 39, no. 2, pp. 9–19, Mar. 2019.

[18] K. Lee, C. Esposito, and S.-Y. Lee, "Vulnerability analysis challenges of the mouse data based on machine learning for image-based user authentication," *IEEE Access*, vol. 7, pp. 177241–177253, 2019.
[19] K. Lee and S. Lee, "Improved practical vulnerability analysis of mouse data according to offensive security based on machine learning in image-based user authentication," *Entropy*, vol. 22, no. 3, pp. 1–16, Mar. 2020.
[20] D. Liu and J.-H. Lee, "CNN based malicious website detection by invalidating multiple Web spams," *IEEE Access*, vol. 8, pp. 97258–97266, 2020.
[21] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.
[22] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "IoT security: Ongoing challenges and research opportunities," in *Proc. IEEE 7th Int. Conf. Service-Oriented Comput. Appl.*, Nov. 2014, pp. 230–234.
[23] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: The road ahead," *Comput. Netw.*, vol. 76, pp. 146–164, Jan. 2015.
[24] N. M. Haller, *The S/Key (Tm) One-Time Password System*, document RFC 1760, Feb. 1995.
[25] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 1, pp. 4–20, Jan. 2004.
[26] E. Fernandes, J. Jung, and A. Prakash, "Security analysis of emerging smart home applications," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2016, pp. 636–654.
[27] T. Halevi and N. Saxena, "Keyboard acoustic side channel attacks: Exploring realistic and security-sensitive scenarios," *Int. J. Inf. Secur.*, vol. 14, no. 5, pp. 443–456, Sep. 2014.
[28] S. Cecconello, A. Compagno, M. Conti, D. Lain, and G. Tsudik, "Skype & type: Keyboard eavesdropping in voice-over-IP," *ACM Trans. Privacy Secur.*, vol. 22, no. 4, pp. 1–34, Dec. 2019.
[29] S. A. Anand and N. Saxena, "Keyboard emanations in remote voice calls: Password leakage and Noise(less) masking defenses," in *Proc. 8th ACM Conf. Data Appl. Secur. Privacy*, Mar. 2018, pp. 103–110.

KYUNGROUL LEE received the B.S., M.S., and Ph.D. degrees from Soonchunhyang University, South Korea, in 2008, 2010, and 2015, respectively. Since 2020, he has been an Assistant Professor with Daegu Catholic University. He also has experience serving or is currently serving on the organizing or program committees of the international conferences and workshops. He has also served as an Editorial Board Member of IJITST, a Topic Editor Member of *Electronics* journal, and a Reviewer Board Member, such as Sensors, Information, and Entropy. His research interests include security protocols, vulnerability analysis, hardware security, reverse engineering, platform security, and offensive security.

JAEHYUK LEE is currently pursuing the bachelor's degree with the School of Computer Software, Daegu Catholic University. His research interests include vulnerability analysis, reverse engineering, and offensive security

CHANG CHOI (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computer engineering from Chosun University, in 2005, 2007, and 2012, respectively. Since 2020, he has been an Assistant Professor with Gachon University. He has authored over 50 publications, including papers in prestigious journal/conferences such as the *IEEE Communications Magazine*, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, the IEEE INTERNET OF THINGS JOURNAL, *Information Sciences*, *Future Generation Computer Systems*, and so on. His research interests include intelligent information processing, semantic web, smart IoT systems, and intelligent system security. He was awarded the academic awards from the Graduate School, Chosun University, in 2012. He also received a Korean government scholarship for graduate students (Ph.D. course), in 2008. He has served or is currently serving on the organizing or program committees of international conferences and workshops, such as ACM RACS, EAI BDTA, IE, ACM SAC, and IEEE CCNC/SeCHID. He has also served as a Guest Editor of high-profile journals, such as the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, *Future Generation Computer Systems*, *Applied Soft Computing*, *Multimedia Tools and Applications*, the *Journal of Ambient Intelligence and Humanized Computing*, *Concurrency and Computation: Practice and Experience*, *Sensors*, and *Autosoft*.

KANGBIN YIM received the B.S., M.S., and Ph.D. degrees from the Department of Electronics Engineering, Ajou University, South Korea, in 1992, 1994, and 2001, respectively.

He is currently a Full Professor with the Department of Information Security Engineering, Soonchunhyang University. His research interests include vulnerability assessment, malware analysis, leakage prevention, secure architecture, especially on mobile network, industrial control systems, and vehicular platforms.

Dr. Yim has served as an Executive Board Member of the Korea Institute of Information Security and Cryptology, the Korean Society for Internet Information, and the Institute of Electronics Engineers of Korea. He also has served as the committee chair of the international conferences and workshops and the Guest Editor of the journals, such as *JIT*, *MIS*, and *JCPS*.