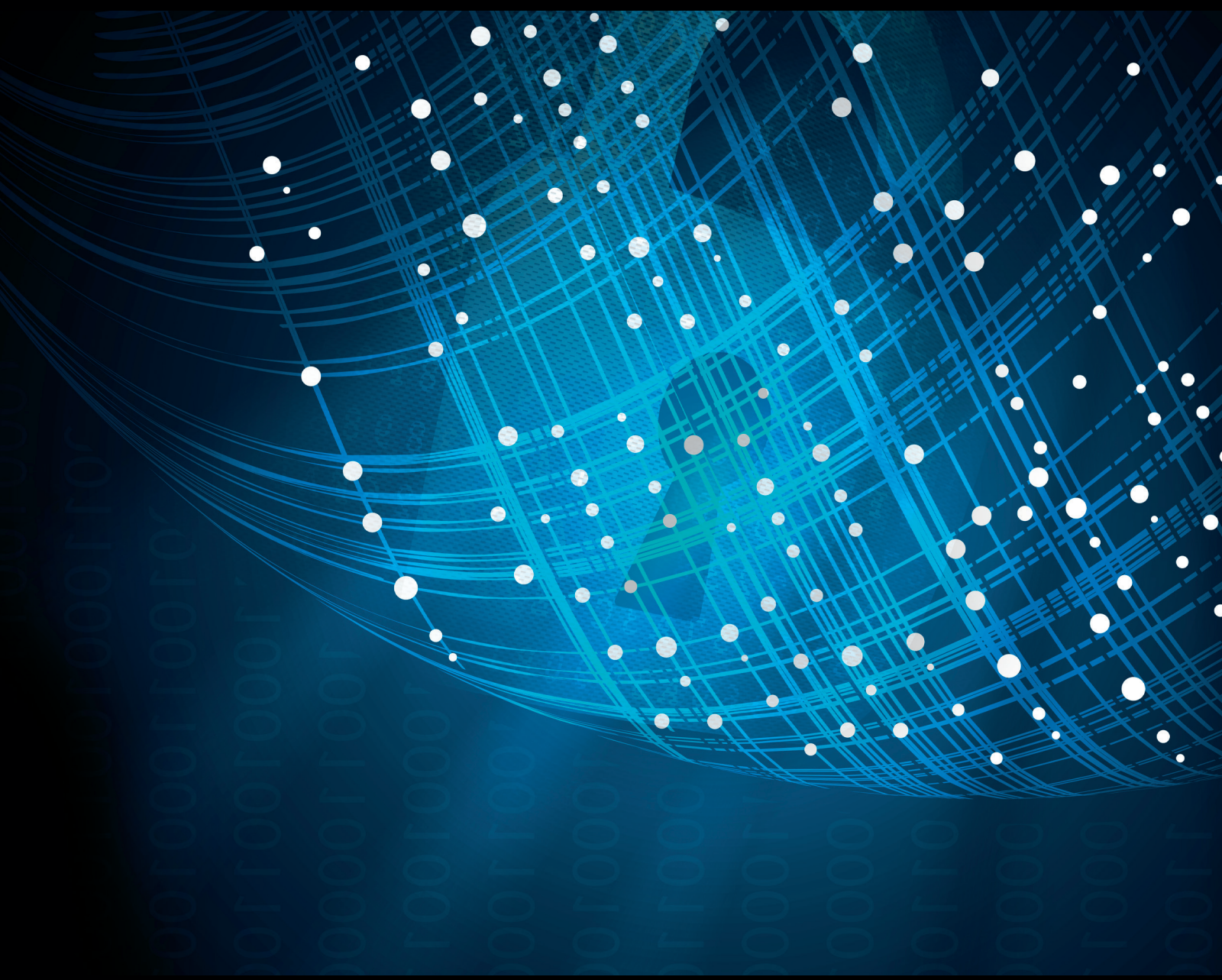
Security and Communication Networks

Security and Privacy for Smart, Connected, and Mobile IoT Devices and Platforms

Lead Guest Editor: Karl Andersson
Guest Editors: Ilsun You and Francesco Palmieri



Security and Privacy for Smart, Connected, and Mobile IoT Devices and Platforms

Security and Communication Networks

Security and Privacy for Smart, Connected, and Mobile IoT Devices and Platforms

Lead Guest Editor: Karl Andersson
Guest Editors: Ilsun You and Francesco Palmieri

Editorial Board

Contents

Security and Privacy for Smart, Connected, and Mobile IoT Devices and Platforms
Karl Andersson, Ilsun You, and Francesco Palmieri
Editorial (2 pages), Article ID 5346596, Volume 2018 (2018)

An Artificial Intelligence Approach to Financial Fraud Detection under IoT Environment: A Survey and Implementation
Dahee Choi and Kyungho Lee
Research Article (15 pages), Article ID 5483472, Volume 2018 (2018)

Analysis and Evaluation of SafeDroid v2.0, a Framework for Detecting Malicious Android Applications
Marios Argyriou, Nicola Dragoni, and Angelo Spognardi
Research Article (15 pages), Article ID 4672072, Volume 2018 (2018)

Detecting Potential Insider Threat: Analyzing Insiders' Sentiment Exposed in Social Media
Won Park, Youngin You, and Kyungho Lee
Research Article (8 pages), Article ID 7243296, Volume 2018 (2018)

A Cascaded Algorithm for Image Quality Assessment and Image Denoising Based on CNN for Image Security and Authorization
Jianjun Li, Jie Yu, Lanlan Xu, Xinying Xue, Chin-Chen Chang, Xiaoyang Mao, and Junfeng Hu
Research Article (13 pages), Article ID 8176984, Volume 2018 (2018)

Design and Analysis of Push Notification-Based Malware on Android
Sangwon Hyun, Junsung Cho, Geumhwan Cho, and Hyoungshick Kim
Research Article (12 pages), Article ID 8510256, Volume 2018 (2018)

A Secure Incentive Scheme for Vehicular Delay Tolerant Networks Using Cryptocurrency
Youngho Park, Chul Sur, and Kyung-Hyune Rhee
Research Article (13 pages), Article ID 5932183, Volume 2018 (2018)

Security Evaluation Framework for Military IoT Devices
Sungyong Cha, Seungsoo Baek, Sooyoung Kang, and Seungjoo Kim
Research Article (12 pages), Article ID 6135845, Volume 2018 (2018)

A Secure and Privacy-Aware Smart Health System with Secret Key Leakage Resilience
Yinghui Zhang, Pengzhen Lang, Dong Zheng, Menglei Yang, and Rui Guo
Research Article (13 pages), Article ID 7202598, Volume 2018 (2018)

Static and Dynamic Analysis of Android Malware and Goodware Written with Unity Framework
Jaewoo Shim, Kyeonghwan Lim, Seong-je Cho, Sangchul Han, and Minkyu Park
Research Article (12 pages), Article ID 6280768, Volume 2018 (2018)

A Homomorphic Network Coding Signature Scheme for Multiple Sources and its Application in IoT
Tong Li, Wenbin Chen, Yi Tang, and Hongyang Yan
Research Article (6 pages), Article ID 9641273, Volume 2018 (2018)

New Certificateless Aggregate Signature Scheme for Healthcare Multimedia Social Network on Cloud Environment
Libing Wu, Zhiyan Xu, Debiao He, and Xianmin Wang
Research Article (13 pages), Article ID 2595273, Volume 2018 (2018)

Survey of Authentication and Authorization for the Internet of Things
Michal Trnka, Tomas Cerny, and Nathaniel Stickney
Review Article (17 pages), Article ID 4351603, Volume 2018 (2018)

A Secure Ciphertext Retrieval Scheme against Insider KGAs for Mobile Devices in Cloud Storage
Run Xie, Chanlian He, Dongqing Xie, Chongzhi Gao, and Xiaojun Zhang
Research Article (7 pages), Article ID 7254305, Volume 2018 (2018)

A High-Security and Smart Interaction System Based on Hand Gesture Recognition for Internet of Things
Jun Xu, Xiong Zhang, and Meng Zhou
Research Article (11 pages), Article ID 4879496, Volume 2018 (2018)

Study to Improve Security for IoT Smart Device Controller: Drawbacks and Countermeasures
Xin Su, Ziyu Wang, Xiaofeng Liu, Chang Choi, and Dongmin Choi
Review Article (14 pages), Article ID 4296934, Volume 2018 (2018)

Jammer Localization in Multihop Wireless Networks Based on Gravitational Search
Tongxiang Wang, Xianglin Wei, Jianhua Fan, and Tao Liang
Research Article (11 pages), Article ID 7670939, Volume 2018 (2018)

A Smart Trust Management Method to Detect On-Off Attacks in the Internet of Things
Jean Caminha, Angelo Perkusich, and Mirko Perkusich
Research Article (10 pages), Article ID 6063456, Volume 2018 (2018)

Cryptanalysis of Compact-LWE and Related Lightweight Public Key Encryption
Dianyan Xiao and Yang Yu
Research Article (9 pages), Article ID 4957045, Volume 2018 (2018)

International Network Performance and Security Testing Based on Distributed Abyss Storage Cluster and Draft of Data Lake Framework
ByungRae Cha, Sun Park, JongWon Kim, SungBum Pan, and JuHyun Shin
Research Article (14 pages), Article ID 1746809, Volume 2018 (2018)

Hindawi
Security and Communication Networks
Volume 2018, Article ID 5346596, 2 pages
https://doi.org/10.1155/2018/5346596

Editorial

Security and Privacy for Smart, Connected, and Mobile IoT Devices and Platforms

Karl Andersson[1], **Ilsun You**[2] **and Francesco Palmieri**[3]

[1]*Luleå University of Technology, Skellefteå, Sweden*
[2]*Soonchunhyang University, Chungcheongnam-do, Republic of Korea*
[3]*University of Salerno, Fisciano (SA), Italy*

Correspondence should be addressed to Karl Andersson; karl.andersson@ltu.se

Received 3 September 2018; Accepted 3 September 2018; Published 27 September 2018

1. Introduction

In recent years, with the rapid development of the smart city paradigm, the Internet of Things (IoT) has raised widespread concern in the whole ICT community. IoT refers to linking the sensors, controllers, machines, people, and things together by using local or wide area communication and cloud technologies, with the Internet as the glue, through a new way to build an intelligent things-to-things network. However, in the near future, the large-scale deployment of the IoT also needs to face many challenges, especially in security and privacy issues for smart, connected, and mobile IoT devices and platforms due to the fact that IoT has different characteristics from traditional communication networks, related to its specific features and threats. In particular, the security solutions for IoT must provide IoT nodes (things, users, servers, and objects) with data authenticity, confidentiality, integrity and freshness certification, and authorization. In addition, privacy protection must also be considered. Many IoT services and applications may expose sensitive and personal information which may be abused by attackers. The concept of privacy may be different, but it should protect the user's personal identity information and maintain a certain degree of anonymity, nonlinkability, and data confidentiality. Of course, it is also necessary to strike a balance between the availability and the security and privacy protection for IoT.

This special issue focuses on the IoT devices and platforms with respect to security and privacy preserving technologies for speeding up the technological progress and attracting more researchers' concerns about the development in this field. In addition, this special issue includes extended versions of the best papers presented at the 2nd International Symposium on Mobile Internet Security (MobiSec'17) held on Jeju, Jeju Island, Republic of Korea, on October 19–22, 2017.

For the current issue, we are pleased to introduce a collection of papers covering a range of topics as follows: frameworks for detecting malicious applications; algorithms for image quality assessment and image denoising based for image security and authorization; push notification-based malware; models for detecting potential insider threat; methods for financial fraud detection; security evaluation framework for military IOT devices; certificateless aggregate signature schemes; high-security interaction systems; secure incentive schemes for vehicular delay tolerant networks; ciphertext retrieval schemes; systems with secret key leakage-resilience; solutions for jammer localization in multihop wireless networks; homomorphic network coding signature schemes for multiple sources; cryptanalysis of compact-LWE and related lightweight public key encryption; smart trust management methods to detect on-off attacks; and network performance and security testing.

Conflicts of Interest

The editors declare that they have no conflicts of interest regarding the publication of this special issue.

Acknowledgments

As always, we appreciate the high quality submissions from authors and the support of the community of reviewers.

Karl Andersson
Ilsun You
Francesco Palmieri

Hindawi
Security and Communication Networks
Volume 2018, Article ID 5483472, 15 pages
https://doi.org/10.1155/2018/5483472

Research Article

An Artificial Intelligence Approach to Financial Fraud Detection under IoT Environment: A Survey and Implementation

Dahee Choi and Kyungho Lee

Center for Information Security Technologies (CIST), Korea University, Seoul 02841, Republic of Korea

Correspondence should be addressed to Kyungho Lee; kevinlee@korea.ac.kr

Received 7 March 2018; Revised 8 June 2018; Accepted 25 June 2018; Published 25 September 2018

Academic Editor: Ilsun You

Financial fraud under IoT environment refers to the unauthorized use of mobile transaction using mobile platform through identity theft or credit card stealing to obtain money fraudulently. Financial fraud under IoT environment is the fast-growing issue through the emergence of smartphone and online transition services. In the real world, a highly accurate process of financial fraud detection under IoT environment is needed since financial fraud causes financial loss. Therefore, we have surveyed financial fraud methods using machine learning and deep learning methodology, mainly from 2016 to 2018, and proposed a process for accurate fraud detection based on the advantages and limitations of each research. Moreover, our approach proposed the overall process of detecting financial fraud based on machine learning and compared with artificial neural networks approach to detect fraud and process large amounts of financial data. To detect financial fraud and process large amounts of financial data, our proposed process includes feature selection, sampling, and applying supervised and unsupervised algorithms. The final model was validated by the actual financial transaction data occurring in Korea, 2015.

1. Introduction

Financial fraud under IoT environment is the fast-growing issue since the mobile channel can facilitate nearly any type of payments. Due to the rapid increase in mobile commerce and the expansion of the IoT environment, financial fraud in mobile payment has arisen and becomes more common. More than 87 percentage of merchants support either mobile site or a mobile application for online shopping or both [1]. Supporting for mobile wallets also helps to increase the overall use of financial fraud under IoT environment. As a result, mobile payments under IoT platform have reached $194.1 billion in 2017, and mobile proximity payments also increased to $30.2 billion in 2017, compared to $18.7 billion in 2016 [2]. Financial fraud can occur in several ways, but the most frequent case is an unauthorized use of mobile payment via credit card number or certification number. Financial fraud via credit card can be classified into two main categories based on the presence of a credit card: (1) the physical card and (2) the virtual card. To commit credit card fraud with a physical card offline, an attacker has to steal the credit card to carry out the fraudulent transactions. The online credit card fraud that does not require the presence of a credit card mainly occurs under IoT environment, since the payment under IoT environment does not require the presence of a physical payment tool; instead, it needs some information such as card number, expiration date, card verification code, and pin number to make the fraudulent payment. For this reason, financial fraud, which usually takes place under the IoT environment, is the most frequent type of financial fraud that involves taking or modifying credit card information. To address the problem of rapidly arising fraud under IoT environment, financial institutions employ various fraud prevention tools like real-time credit authorization, address verification systems (AVS), card verification value, positive and negative list, etc. [3].

However, existing detection systems depend on defined criteria or learned records, which makes it difficult to detect new attack patterns. Therefore, various methods using machine learning and artificial neural networks have been attempted to capture new financial fraud.Our contributions are as follows:

(a) Research on the various research papers based on the machine learning and artificial neural network techniques

and review of latest detection techniques mainly from 2016 to 2018

(b) Analysis of advantages and limitations for the latest research paper

(c) Model building based on the implementation of reviewed paper and full process experiment based on actual financial transaction dataset

(d) Deriving the result in specific way through validation on each step in the process

(e) Comparison with traditional machine learning and deep learning based on artificial neural networks for fraud detection.

2. Literature Review

We reviewed the latest techniques to detect anomaly and trust relaying in IoT environment. Also, we focused on reviewing the methods and algorithms to detect financial fraud from traditional methods to the latest one. V. Sharma et al. proposed a novel solution in the form of fission computing. The proposed approach relies on the edge-crowd integration for maintenance of trust and preservation of privacy rules in social IoT environment. They performed analysis through numerical simulations by using a safe network and presented a case study on the detection of fake news sources in social IoT environment [4]. Also, a pervasive trust management framework is presented for Pervasive Online Social Networks (POSNs), which is capable of generating high trust value between the users with a lower cost of monitoring [5]. As a solution to identify anomalies in IoT environment, a model was proposed on the basis of cognitive tokens, which provide an Intelligent Sensing Model for Anomalies (ISMA) detection by deliberately inducing faulty data to attract the anomalous users [6]. Van Wyk Hartman suggested automatic network topology detection and fraud detection. If fraud is detected in the distribution network, the system schedules the follow-up and field investigation to investigate and fix the fraud [7]. Also, systems and methods for online fraud detection have been proposed. The front end device generates a first dynamic device identification based on dynamic device characteristics and the back end device generates a second dynamic device identification based on the dynamic device characteristics of the front end device for an authentication [8].

Various learning methods and algorithms have been applied for data analysis and anomaly detection. The learning method of supervised, unsupervised, and artificial neural networks approach has been attempted and a web service-based collaborative scheme for financial fraud detection has been proposed [9, 10]. Also, an efficient fraud detection system which is adaptive to the behavior changes by combining classification and clustering techniques has proposed. The proposed financial fraud detection system is composed of two stages comparing the incoming transaction against the transaction history to identify the anomaly using BOAT algorithm, a scalable algorithm, in the first stage. The false alarm rate suspected anomalies are checked with the fraud history database and decide that the detected anomalies are due to the fraudulent transaction or any short-term change in spending profile in the second stage. BOAT algorithm can also incrementally update a decision tree when the training dataset changes dynamically [11]. The machine learning based research has also been proposed, as a web service-based collaborative scheme for credit card fraud detection [9] . The detection of fraud is based on the genetic algorithm calculation and customer behavior [12], and an efficient financial fraud detection system which is adaptive to the behavior changes by combining classification and clustering techniques, a scalable algorithm named BOAT, has also been proposed [12] . As a traditional method of financial fraud detection, the Dempster–Shafer adder (DSA) based on Dempster–Shafer theory and the use of Bayesian learning research had been proposed. In this research, a transaction conversed with the suspicion score, which can be referred to as the probability of the fraudulent transaction, based on the index in the transaction history database. BLAST and SSAHA algorithm are sequence alignment algorithms and used as the alignment of sequences for an efficient technique to examine the spending behavior of customers [11] . To calculate and predict the probability from the user's existing financial information and to build a multilayer model of program behavior, Hidden Markov Model (HMM) has been proposed. The key idea for applying HMM for anomaly detection is to build a multilayer model of program behaviors using HMMs and various methods [13]. Genetic algorithm calculates and finds critical values which aim to obtain better solutions. During a credit card transaction, the fraud has to be deducted in real time and the number of false alerts is being minimized by using genetic algorithm. The detection of fraud is based on the customer‘s behavior [14]. Artificial neural network (ANN) is applied for detecting fraud, mainly in the context of supervised classification and it can be used in recognition of characteristics timely and make predictions [12] . CARDWATCH is a database mining system used for credit card fraud detection. The system is based on a neural learning module, interfaced with a variety of commercial databases [15]. The module includes Global Constants Module (GCM), Graphical User Interface Module (GUIM), Database Interface Module (DBIM), Learning Algorithms Library (LAL), and Learning Algorithm Interface Module (LAIM). The proposed system is mentioned as easily extensible and able to work directly on a large variety of commercial databases. Fraud detection with Bayesian Belief Network (BBN) has also proceeded [16]. SODRNN is the reverse K-nearest algorithm for data stream outlier detection. Since SODRNN needs only one pass of scan, it is suitable for the credit card fraud detection of massive data processing [17]. Decision trees and Support Vector Machine (SVM) are a kind of supervised learning method detecting normal transaction and fraud by classifier, which can predict whether the transaction is normal or fraud. Decision tree and SVM are to compare the transaction information with historical profile patterns to predict the probability of being fraudulent for a new transaction [18]. There are also other methods for credit card fraud detection such as fuzzy Darwinian detection which comprises Genetic Programming (GP). Syeda et al. in 2002 proposed fuzzy neural networks which run on parallel machines to speed up the rule production for credit

card fraud detection which was customer-specific. In this technique, the Granular Neural Network (GNN) method that uses fuzzy neural network which is based on knowledge discovery was taken to train the network fast and can be processed for fraud detection in parallel [19]. In APATE approach, intrinsic features derived from the characteristics of incoming transactions such as Recency, Frequency, and Monetary (FRM). The customer spending history and network-based features, by exploiting the network of credit card holders and merchants, are deriving a time-dependent suspiciousness score for each network object [20]. A combined method of decision tree, neural networks, and logistic regression [21], decision trees and Support Vector Machine (SVM) [22], a combined method of decision tree, neural networks (NN), and logistic regression [19], Self-Organizing Map (SOM) combined with Gaussian function [22], and fuzzy logic combined with Self-Organizing Map has been introduced for the financial fraud detection method [23]. A combined method of SVM, random forests, logistic regression [24], Self-Organizing Map Neural Network (SOMNN) [25], genetic algorithm with behavior-based technique, and Hidden Markov Model (HMM) has been attempted [26].

We reviewed the latest financial fraud detection methods using machine learning and deep learning methodology. A total of thirteen recent papers published in 2016 and 2017 were reviewed. This paper mainly focused on the papers with experimental results using existing financial datasets and proving the detection efficiency through the dataset. The method and algorithm applied to the dataset are specified and the validation method is also indicated. We also reviewed the advantages and limitations of each paper. A more detailed review of recent financial detection methods is in Table 1.

3. Model and Methodology

Existing detection systems depend on defined criteria or learned records which make it difficult to detect new attack patterns. To discover new patterns and achieve higher detection accuracy, machine learning methods based on supervised learning and unsupervised learning and deep learning using artificial neural networks have been actively studied. Our proposed research analyzes the most recently used methods in financial fraud studies of machine learning and deep learning from 2016 to 2017. Also, our research implemented both machine learning method and deep learning method to compare the efficiency of detecting financial fraud transactions. In the machine learning process, we applied the unsupervised learning method to discover underlying threats and supervised learning for the accurate classification of fraud transactions under IoT environment. The overall processes of detecting financial fraud include sampling process for class imbalance problem and feature selection process for an accurate model. The previous research papers are mainly related to specific approach such as algorithms, which needs a further step for implementation. Moreover, in applying the methods of machine learning, previous research only used one of the learning methods between supervised and unsupervised learning. However, our research has performed the overall process of financial fraud detection in practical perspective based on supervised and unsupervised learning method. Also, we are proposing a practical method by applying sampling process and feature selection process for solving data unbalanced problem and rapid detection in the real world. Furthermore, our research is expected to be useful for practical use since our experiment has a validation score for each process and is based on real transaction data.

3.1. Machine Learning. Machine learning is a field that machines learn concepts using data, using statistical analysis to predict and classify and input data as an output value. The field of machine learning is divided into supervised learning and unsupervised learning depending on the learning method. Supervised learning predicts the value of input data and is classified with the given label. On the other hand, unsupervised learning is performed in a state where data is not labeled and is often called a clustering process.

The proposed model consists of data preprocessing, sampling, feature selection, application of classification, and clustering algorithm based on machine learning. In this paper, the validation step is performed for each step to verify the effectiveness of proposed financial fraud detection model. In the preprocessing process, data correlation analysis and data cleaning process which cleans the noise data are performed. Also, data transformation, integration, and reduction are included in this process. The following process is the sampling process which evaluates dataset with various ratios for verification through random oversampling and undersampling method. Feature selection process has been performed by the filter-based method. After the feature selection process, clustering process with the proposed algorithm performs and this result is used as a training set in the classification process. By applying supervised algorithms to the previous result, which was derived in the clustering process previously, the higher prediction could be achieved. The model validation process is performed with precision and recall rate through F-measure. The overall structure of the proposed fraud detection system model is as Figure 1.

3.2. Sampling. Imbalanced problem in the data could mislead the detection process to the misclassifying problem and a real transaction dataset of financial transaction usually contains a data imbalanced problem. Previous research has proposed a random minority oversampling method and clustering-based undersampling approach to select the representative data as training data to improve the classification accuracy for minority class [40].

To solve this class imbalanced problem, our research generates various datasets using Synthetic Minority Oversampling Technique (SMOTE) and Random Undersampling (RUS) for the more accurate experiment. SMOTE is an oversampling technique that uses a method of generating arbitrary examples rather than simply oversampling through duplication or replacement [41]. RUS was applied for downsizing the normal transactions by extracting sample data randomly for the class imbalance problem. Since the low ratio of anomalous data might lead to less precise results, our

Table 1: Review of financial fraud detection methods.

Cited Used	Data Description	Applied Method	Validation Method	Advantage	Limitation
[27]	Credit and debit card transactions of the Spanish bank BBVA, from January 2011 to December 2012	Multilayer perceptron (MLP)	True Positive Rate (TPR), Receiver Operating Characteris-tic (ROC curves)	The improvement of accuracy in detecting results by using parenclitic networks reconstruction for feature extraction	Requires the cases where the features are not correlated or not extracted by parenclitic networks
[28]	Transactions of National banking group of Italy, from April 2013 to August 2013	Multi-objective genetic algorithm	TPR, ROC curves	Provide feature selection process via the auto-tuning method	Should be applied to cases other than Banksealer
[29]	UCSD Data Mining Contest 2009 data	Deep neural network (DNN)	Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Errors (MAE), Root Mean Squared Log Error (RMSLE)	Study on the importance of features based on the deep learning method	Does not have accurate experimental explanation process and validation
[30]	Dataset achieved from the second robotic & artificial intelligence festival of Amirkabir University	Decision trees	F-Measure	Ensemble classification is performed using cost-sensitive decision trees in a decision forest framework	Having a class imbalance problem
[31]	German dataset (which has been used in KDD99 competition), Australian credit cards' open dataset	Particle swarm optimization (PSO), Teaching-learning-based optimization (TLBO)	Confusion Matrix (True positive, True negative, False positive, False negative)	Experiment with various datasets	Detection accuracy is relatively low
[32]	Not specific	Linear Regression, Artificial Neural Networks (ANN), K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Decision Stump, M5P Tree, Decision Table	Normalized Root Mean Squared Error (NRMSE), TPR, F-Measure	A comparative study using various algorithms	Detection accuracy should be increased
[33]	UCI German credit card dataset	SVM	K-fold Cross validation	As Gaussian kernels including RBF are with appropriate regularization, it guarantees a globally optimal predictor which minimizes both the estimation and approximation errors of a classifier	There is no comparison with other algorithms and there is no explanation to verify SVM algorithms superiority to others
[34]	Credit card transaction data from commercial bank in China	Convolutional Neural Networks (CNN), K-Means	F-Measure	Designing a feature called trading entropy based on the latest consumption preferences for each customer and generating synthetic fraudulent samples from real frauds by a cost-based sampling method	Detection accuracy should be increased

Table 1: Continued.

Cited Used	Data Description	Applied Method	Validation Method	Advantage	Limitation
[35]	Banking transaction dataset in Iran	KNN	Accuracy, Re-call, Precision	A novel approach combining K-nearest neighbor, association rules like Apriori algorithm	The validation is not specific and it is difficult to compare the proposed results with other algorithms
[36]	Open dataset: ccFraud	NN, PSO, Auto-associative neural network (AANN), Particle swarm optimization auto-associative neural network (PSOAANN)	MSE, Classification Rate (CR)	Combined parallelization of the auto-associative neural network in the hybrid architecture	Dataset is highly unbalanced and detection accuracy should be increased
[37]	Open dataset: MIT Human Dynamics Lab	SVM, Fuzzy clustering	TPR, FPR, ROC curves	Divide the fraud detection system into two principal modules	Would be better to compare it with more diverse algorithms
[38]	Transactions from a large national bank in Italy, collected from December 2012 to August 2013	Principal component analysis (PCA), DBSCAN	ROC curves	Operate in online processing	Accuracy by validation is not constant
[39]	Not specific	Self- organizing map (SOM)	TPR, FPR	Division of transactions to form an input matrix and ability to be applied to a large complex set	Only have compared to one algorithm

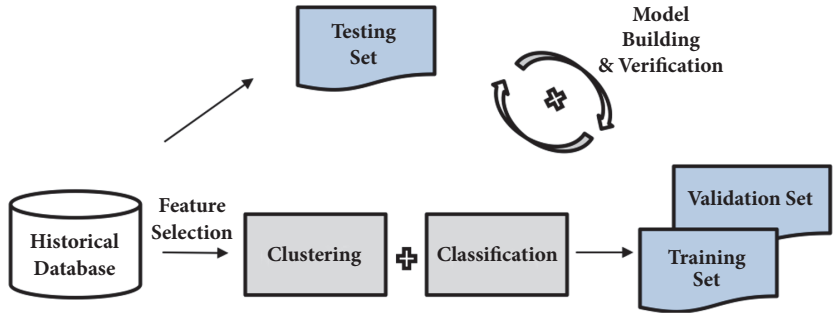


Figure 1: The overall detection process of the proposed process.

research applied both SMOTE and RUS for generating the different ratio of sampling dataset to increase the reliability and accuracy of our proposed research.

3.3. Feature Selection. Feature selection has been proven to be effective and efficient for machine learning problems. The objectives of feature selection include building simpler and more comprehensible models, improving data mining performance such as predictive accuracy and comprehensibility. Also it includes preparing to remove redundancy and irrelevancy for understandable data [42]. Feature selection can be divided into wrapper and filter method. The wrapper method relies on the predictive performance of a predefined learning algorithm to evaluate the selected features. It repeats the searching step and evaluating criteria until desired learning performance is obtained. The drawback of wrapper method is that the search space could be vast and it is relatively more expensive than other methods. Filter method is independent of any learning algorithms and relies on certain characteristics of data to assess the importance of features. Features are scored based on the scores according to the evaluation criteria, and the lowest scored features are removed [43]. For this reason, we applied filter-based feature selection algorithms for feature selection method, which is the fastest and also suitable for practical use. Feature selection based on filter method can be categorized into ranker and the subset selector [44].

In the proposed research, we selected eight subset feature selection algorithms and six ranked feature selection algorithms to select features among existing features. Also, we

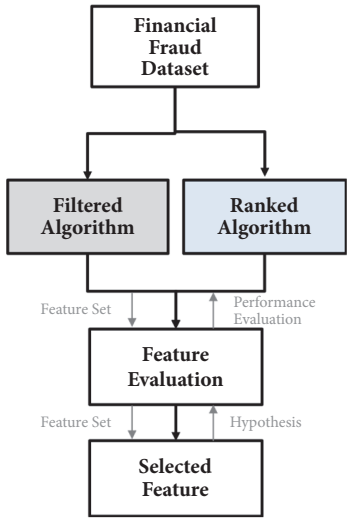


Figure 2: Flowchart of feature selection process.

assigned the score to evaluate features based on the frequency. The ranker algorithms are calculated by weighting the higher ranks. The results of two feature selection algorithms are combined to prioritize the features by selecting features which exceed the parameter in frequency and ranking. Figure 2 is the flowchart of feature selection process proposed in our research.

3.4. Deep Artificial Neural Networks. Deep learning (DL) is a subfield of the machine learning inspired by the structure and function of the brain called artificial neural networks. An artificial intelligence function imitates the working of the human brain in processing data and creating patterns for use in decision-making area, through the capability of unsupervised learning from data that is unstructured or unlabeled. Artificial Neural Networks (ANN) are called neural networks or multilayer perceptrons. A perceptron is a single neuron model that was a precursor to larger neural networks. In neural networks, the predictive capability comes from the hierarchical or multilayered structure of the networks [45]. Also, multilayer perceptron has a neural network with one or more intermediate layers between the input and output layers. Figure 3 is a simple artificial neural network and the middle layer between the input layer and the output layer is called a hidden layer. The network is connected in the direction of the input layer, the hidden layer, and the output layer and is a feedforward network in which there is no direct connection from the output layer to the input layer in each layer. Most multilayer perceptrons can be learned using backpropagation learning algorithms.

3.5. Validation. In machine learning method, which is based on statistics, F-measure is a well-known measurement of model performance between predicted class and actual class using recall and precision. In our research, the F-measure is used to measure the ratio between the actual value and the value that the algorithm detects and predicts [46] and the confusion matrix used to measure the F-measure value is as in Table 2.

4. Experiment

We validated each step to measure the efficiency of the proposed model. Before the feature selection process, the accuracy of each algorithm with raw dataset was measured. After the previous step, the accuracy of each algorithm using the feature extracted through the proposed feature selection method was measured. We used both supervised learning algorithm and unsupervised learning algorithm. In addition to actual datasets, open data were also applied additionally for more accurate verification of our proposed methods.

4.1. Data Description. Our research was conducted based on the actual payment data under IoT environment occurring in Korea, 2015. With the agreement of a major financial institution, provider collected actual financial transaction data for 6 months from June to November. A total of 270,000 pieces of data were extracted from the September data and used as training data. Among data, 21 characteristics are extracted as features (transaction serial number, transaction type, certification date, authentication time, transaction status, telecommunication company, phone number, transaction amount, corporation ID, shop ID, service ID, email hash, IP information, authenticated client version, etc.). For the protection of personal information, key information has been anonymized and data which can identify an individual has been converted to the hash value.

4.2. Modeling Process. In this paper, we aimed to discover hidden patterns by using unsupervised learning and supervised learning for more accurate classification. To design the detection system as to be useful in the operation in the real environment, we proposed the feature selection method that can be applied to the automation system. Therefore, we constructed the system model process by applying the feature selection method on unsupervised learning algorithm firstly and then applied supervised learning algorithm later for accurate classification based on the above experimental results by open dataset and real dataset. The final model validation was performed based on actual financial transaction data in Korea. Also, we compared the final accuracy of the proposed machine learning based detection model and the detection accuracy of models using artificial deep neural networks.

The proposed machine learning based model includes various proportions of the sampling process for application in the real environment and includes an algorithm based automatic feature selection process. In addition, we apply algorithms based on unsupervised learning using selected features and apply algorithms based on supervised learning for more accurate classification. On the other hand, the deep learning model derives the optimum value through the parameter adjustment of the neural networks. Plot (a) in Figure 4 shows the classification accuracy of the UCI German credit card data, by dividing the data before and after the

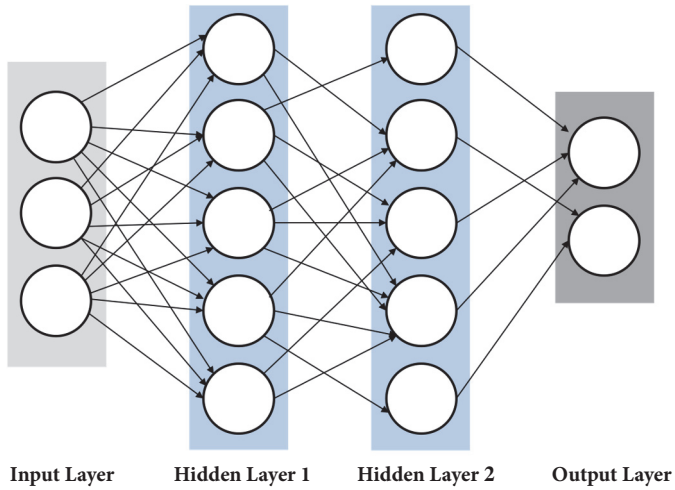


Figure 3: A simple artificial neural network configuration.

Table 2: Confusion matrix.

	Predicted Positive	Predicted Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

feature selection process and applying it to each algorithm to detect abnormal transactions. Applied algorithms are clustering algorithms: EM, simpleK, DensityBased, LVQ, XMeans, FarthestFirst, Hierarchical, and Self-Organizing Map. The purple line indicates the accuracy before the feature selection, and the orange line indicates the accuracy after the feature selection. Plot (b) in Figure 4 shows the F-measure value of classification and also the purple line indicates the value before the feature selection process and the red line indicates the value after the feature selection process.

The results of experiments based on the open dataset show that the F-measure value arises in the majority of algorithms after the proposed feature selection process. The algorithms based on the unsupervised learning have achieved a maximum accuracy improvement of 11.5% and an average of about 11% after the feature selection process in open dataset. Figure 5 is the distribution of accuracy and F-measure value before and after the feature selection process.

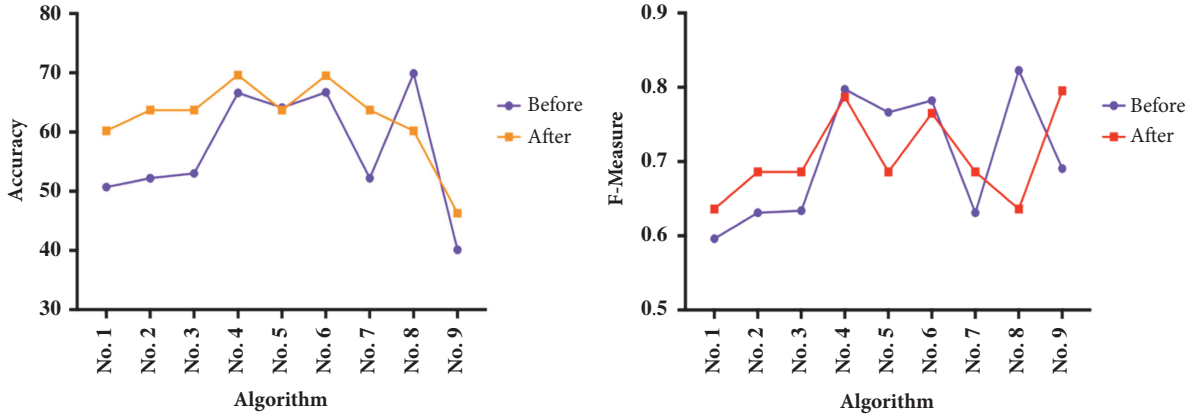
Table 3 shows the detailed results of F-measure value in experiments based on the open dataset before and after the feature selection process. Specifically, the proportion of 90:10 ranked the highest in average F-measure value of 0.7475. The proportion of 95:1 ranked second highest in average value of 0.7387 and 99:1 ranked third in the average F-measure value of 0.7131.

The real dataset is difficult to detect due to highly unbalanced data problem. To find the sampling ratio suitable for the financial dataset, various sampling experiments were conducted. Sampling rates were 50:50, 60:40, 70:30, 80:20, 90:10, 95:5, and 99:1. Plot (a) in Figure 6 shows the average of the detection results in accuracy based on clustering algorithms at various sampling ratios. Also, Plot (b) in Figure 6 shows the average of the F-measure value at various sampling ratios. Results indicate that, at dataset, the 95:5 ratio was the most efficient, followed by the 90:10 ratio.

The five algorithms with good detection efficiency among clustering algorithms were selected and experiments were conducted. The selected algorithms were applied to various ratios as described above, and the proposed model was validated using actual financial transaction dataset occurring in Korea, 2015. Details about accuracy and F-measure in various ratios with real dataset occurring in September are as follows in plot (a) and plot (b) in Figure 7. For more accurate validation, we performed validation process with more dataset. Additionally, accuracy and F-measure in various ratios with real dataset occurring in October and November are in Figures 8 and 9.

The accuracy in detection averages of the clustering algorithms in each dataset is in plot (a) in Figure 10, which includes detection values at various sampling ratios. Plot (b) in Figure 10 shows the average of F-measure via clustering algorithms in various sampling ratios.

The final detection is performed through classification algorithms in the process of sampling, feature selection, and clustering. We aimed to discover hidden patterns by using both unsupervised learning and supervised learning. Therefore, we constructed the system model process by applying feature selection and clustering algorithm firstly and then apply classification algorithm later for accurate classification based on the above experimental results. Six types of classification algorithms were used and we divide the results with sampling ratio for more detailed information. For each ratio, the detection rate of the classification algorithm was measured based on the average detection rate of the previous clustering algorithms. Final detection results of classification in the various ratio are as in Figure 11.



(a) (b)

Figure 4: (a) Accuracy before and after the feature selection process. (b) F-measure before and after the feature selection process.

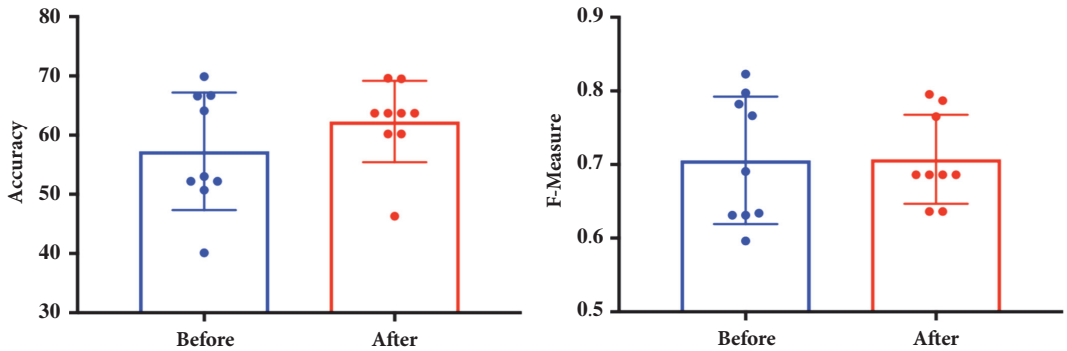


Figure 5: Distribution of accuracy and F-measure value before and after the feature selection process.

Table 4 is the F-measure value of final detection performed through classification algorithms in various sampling ratios.

5. Results

We performed the validation based on the identical actual financial transaction data for machine learning method and artificial neural network. In conclusion, the well-known machine learning method has a higher fraud detection rate than the artificial neural network. By integrating the ratios, the maximum detection rate of the machine learning method was 1, the lowest detection rate was 0.736, and the average detection rate was 0.98618 when all of the algorithms were utilized. The maximum detection rate in all ratios of the artificial neural network was 0.914, the lowest detection rate was 0.651, and the average detection rate was 0.77228. Specific numerical values for each method are shown below.

Results in Figure 12 show the F-measure value of the artificial neural network for detecting financial fraud in various ratios. The ANN achieved an average detection rate of 0.77228 at various ratios; however, it reached a detection rate of 0.914 for each of the 95: 1 and 99: 1 ratios as in Figure 12.

In machine learning model, the experiments were performed from clustering processes such as EM, simple, FarthestFirst, XMeans, and DensityBased algorithms. Classification algorithms such as NaiveBayes, SVM, Regression, OneR, C4.5, and RandomForest were performed and the final result was measured.

In clustering algorithms, EM algorithm reached an average of 0.99862 in fraud detection. DensityBased algorithm ranked second top in fraud detection and reached 0.98788. More details are in Table 5 and Figure 13.

In classification algorithms, Regression reached an average of 0.99971 in fraud detection. Also, RandomForest reached an average of 0.99969 and second top in fraud detection. C4.5 ranked the third tier by reaching 0.99943. More details are in Table 6 and Figure 14.

Comparison with machine learning based classification algorithms and artificial neural networks for accuracy in financial fraud detection is as follows in Figure 15. Six classification algorithms were used for the final classification and compared with the artificial neural network algorithm. We measured the result of the modeling process with F-measure via real dataset. The result classified by well-known machine learning algorithm and the artificial neural network algorithm in various sampling ratios is shown in Figure 15.

Table 3: F-measure of clustering algorithms before and after the feature selection process.

Ratio	Stage	EM	simpleK	FarthestFirst	XMeans	DensityBased
50:50	Before	0.5297	0.4815	0.2692	0.4815	0.4806
	After	0.6134	0.6443	0.6647	0.6443	0.6492
60:40	Before	0.5787	0.5186	0.6869	0.5186	0.5179
	After	0.6319	0.6820	0.7473	0.6820	0.6835
70:30	Before	0.6222	0.6159	0.7527	0.6159	0.6163
	After	0.6453	0.7112	0.8202	0.5400	0.7121
80:20	Before	0.6575	0.6540	0.8085	0.6540	0.6541
	After	0.6540	0.6268	0.8849	0.7178	0.5859
90:10	Before	0.6841	0.6892	0.8590	0.6892	0.6932
	After	0.6610	0.6402	0.9426	0.8126	0.6813
95:1	Before	0.6896	0.7053	0.9028	0.8926	0.7076
	After	0.6732	0.6732	0.9130	0.7608	0.6733
99:1	Before	0.7574	0.7212	0.8328	0.7212	0.7296
	After	0.6829	0.7373	0.7143	0.7538	0.6775
AVG	Before	0.6456	0.6265	0.7307	0.6532	0.6284
	After	0.6516	0.6735	0.8124	0.7016	0.6661

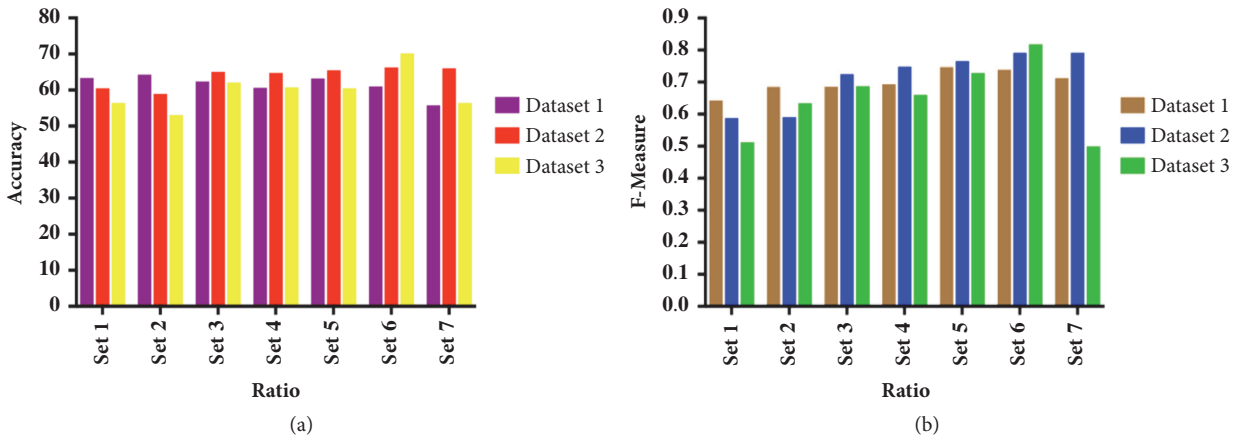


Figure 6: (a) Average detection accuracy in various sampling ratios. (b) Average F-measure in various sampling ratios.

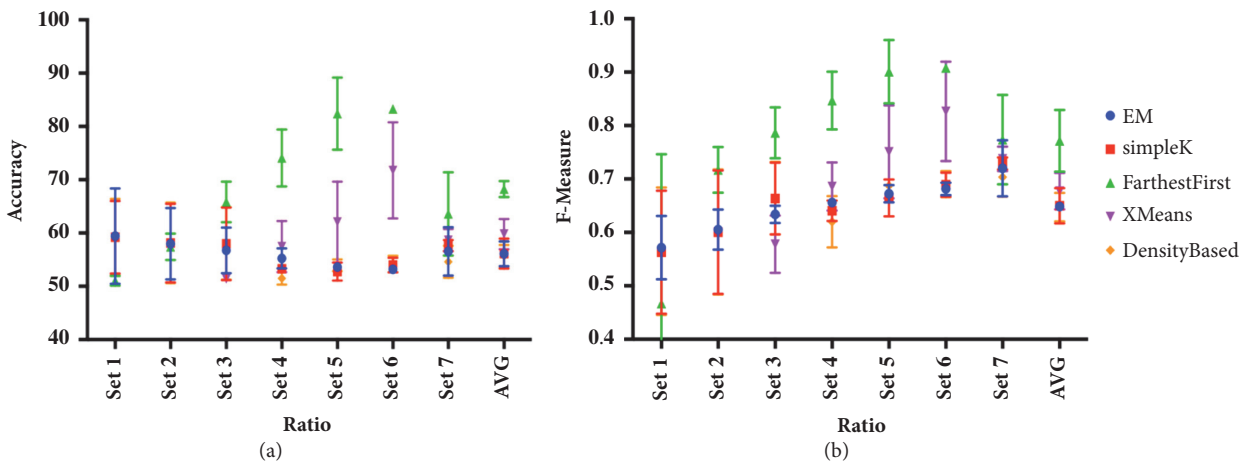


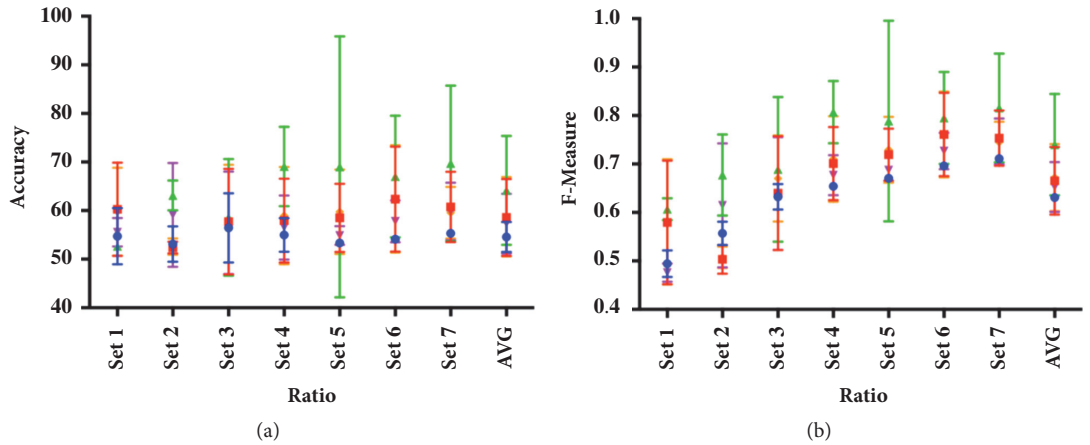
Figure 7: (a) Accuracy in various ratios with real dataset occurring in September. (b) F-measure in various ratios with real dataset occurring in September.

Table 4: F-measure of final classification in various ratios.

	Ratio	Naïve Bayes	SVM	Regression	OneR	C4.5	Random Forest
EM	50:50	1	0.988	0.999	1	1	1
	60:40	1	0.989	0.999	1	1	1
	70:30	0.999	0.999	1	0.983	0.999	1
	80:20	1	0.994	1	1	1	1
	90:10	1	1	1	1	1	1
	95:5	1	1	1	1	1	1
	99:1	0.993	1	1	1	1	1
simpleK	50:50	0.949	0.995	1	0.912	0.997	0.998
	60:40	0.952	0.996	1	0.91	0.998	0.999
	70:30	0.952	0.996	1	0.912	0.999	0.999
	80:20	0.908	0.999	1	0.906	0.999	0.999
	90:10	0.908	1	1	0.898	0.999	1
	95:5	1	1	1	1	1	1
	99:1	0.959	1	1	0.929	1	1
Farthest First	50:50	0.998	0.999	1	0.983	1	1
	60:40	0.998	0.999	1	0.983	0.999	1
	70:30	0.999	1	1	0.983	0.999	1
	80:20	0.999	1	1	0.983	1	1
	90:10	0.999	1	1	0.984	1	1
	95:5	0.995	0.999	1	0.828	1	1
	99:1	0.979	1	1	0.736	1	1
XMeans	50:50	0.949	0.995	1	0.912	0.997	0.998
	60:40	0.952	0.996	1	0.910	0.998	0.999
	70:30	0.996	1	1	0.988	1	1
	80:20	0.949	0.997	1	0.915	1	1
	90:10	0.959	1	1	0.880	1	1
	95:5	0.947	0.999	1	0.913	1	1
	99:1	0.999	1	1	0.908	1	1
Density Based	50:50	0.956	0.989	0.997	0.927	0.998	0.999
	60:40	0.956	0.992	0.998	0.918	0.999	0.999
	70:30	0.955	0.993	0.998	0.919	0.999	0.999
	80:20	0.975	1	1	0.977	1	1
	90:10	0.974	1	0.999	0.975	1	1
	95:5	1	1	1	1	1	1
	99:1	1	1	1	1	1	1

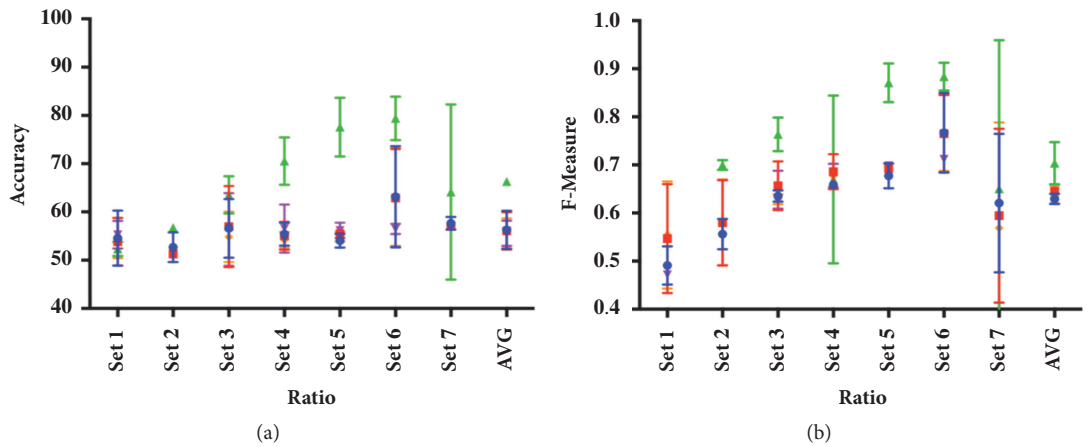
Table 5: Average of clustering algorithms in fraud detection.

	Naïve Bayes	SVM	Regression	OneR	C4.5	Random Forest	AVG
EM	0.99886	0.99571	0.99971	0.99757	0.99986	1	0.99862
simpleK	0.94686	0.99800	1	0.92386	0.99886	0.99929	0.97781
Farthest_F	0.99529	0.99957	1	0.92571	0.99971	1	0.98671
XMeans	0.96443	0.99629	1	0.91800	0.99929	0.99957	0.97990
Density_B	0.97371	0.99754	0.99886	0.95943	0.99943	0.99957	0.98788



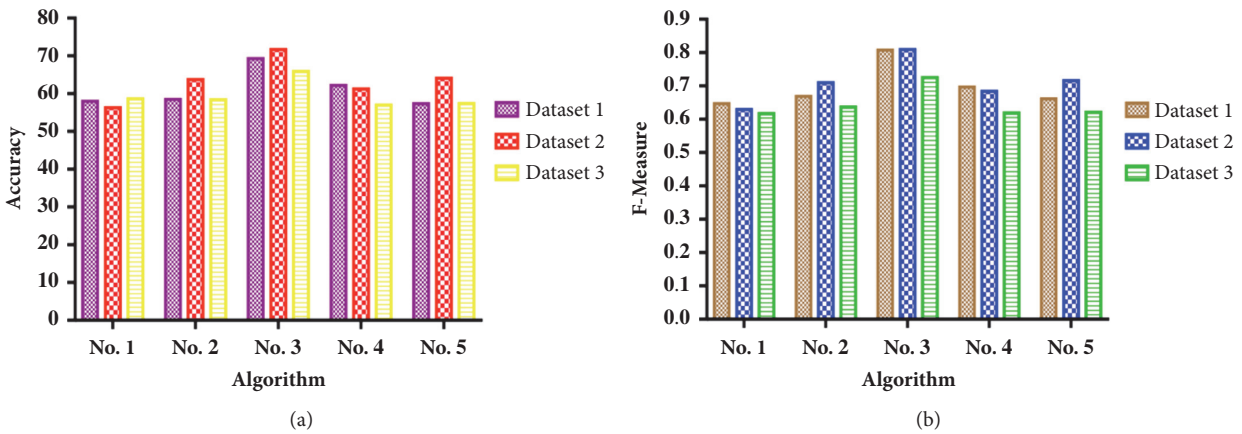
(a) (b)

Figure 8: (a) Accuracy in various ratios with real dataset occurring in October 2015. (b) F-measure in various ratios with real dataset occurring in October 2015.



(a) (b)

Figure 9: (a) Accuracy in various ratios with real dataset occurring in November 2015. (b) F-measure in various ratios with real dataset occurring in November 2015.



(a) (b)

Figure 10: (a) Average of detection accuracy via clustering algorithms in various sampling ratios. (b) Average of F-measure via clustering algorithms in various sampling ratios.

Table 6: Average of classification algorithms in fraud detection.

	EM	simpleK	FarthestFirst	XMeans	DensityBased	AVG
NaiveBayes	0.99886	0.94686	0.99529	0.96443	0.97371	0.97583
SVM	0.99571	0.99800	0.99957	0.99814	0.99629	0.99754
Regression	0.99971	1	1	1	0.99886	0.99971
OneR	0.99757	0.92386	0.92571	0.91800	0.95943	0.94491
C4.5	0.99986	0.99886	0.99971	0.99929	0.99943	0.99943
RandomForest	1	0.99929	1	0.99957	0.99957	0.99969

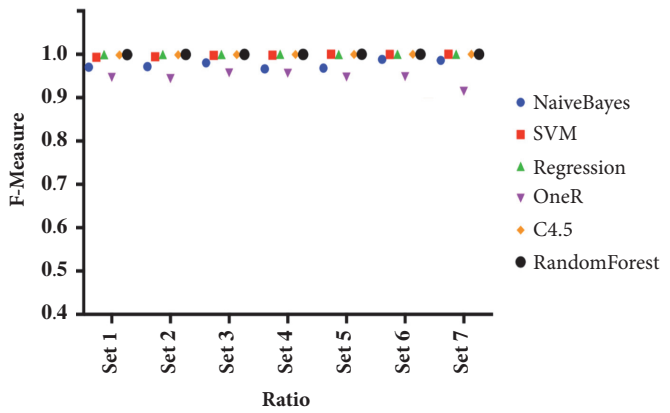


Figure 11: Final detection results of classification algorithms in various ratios.

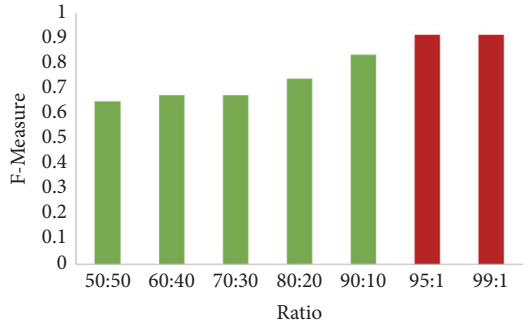


Figure 12: Results of artificial neural network in various ratios for detecting financial fraud.

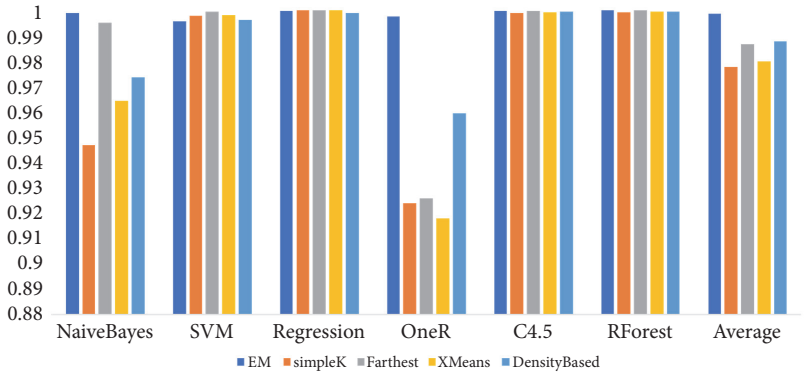


Figure 13: Detection average of clustering algorithms in F-measure.

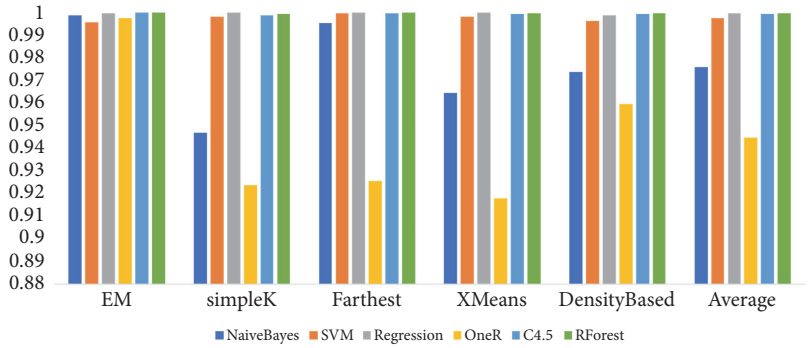


Figure 14: Detection average of classification algorithms in F-measure.

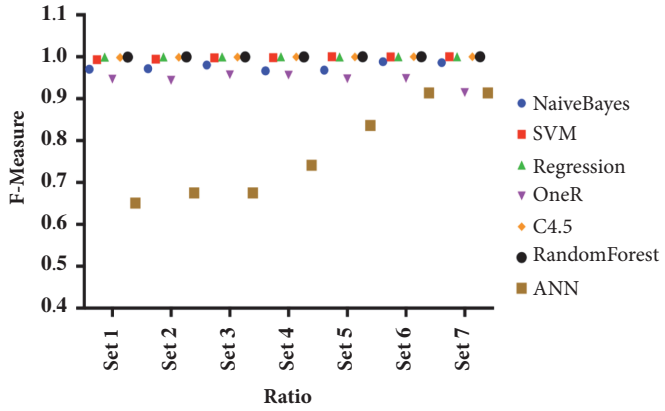


Figure 15: Final financial fraud detection of various algorithms and artificial neural network in various ratios.

6. Conclusions

In this paper, we reviewed the latest financial fraud detection technique using machine learning and artificial neural networks and implemented the experiment based on the real financial data in Korea. The process based on the machine learning method consists of the feature selection process based on the filter method, the clustering process, and the classification process. Experimental results show that machine learning based method has higher detection efficiency than neural networks at various ratios; however, the feature selection process must be performed according to input data. Also, machine learning based process has to verify the optimal combination of clustering algorithms and classification algorithms. Validation of various financial data sets will be performed in the future work. Neural networks reached a particularly high detection accuracy at 95: 1 and 99: 1 ratios, which is nearly similar to the actual ratio in the real world. However, the process takes relatively longer than the machine learning process. In the future work, we aim to improve the accuracy and processing time of the financial fraud process in real time combined with both machine learning based process and deep artificial neural networks.

Data Availability

Our research was conducted based on the actual payment data under IoT environment occurring in Korea, 2016. Since security agreement for the data has been written, it cannot be provided online. The authors are sincerely very sorry for not providing the data online.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] k. corp, "Mobile payments fraud survey report," 2016.

[2] "Javelin strategy and research," 2016.

[3] S. Panigrahi, A. Kundu, S. Sural, and A. K. Majumdar, "Credit card fraud detection: a fusion approach using dempstershafer theory and bayesian learning," *Information Fusion*, vol. 10, no. 4, pp. 354–363, 2009.

[4] V. Sharma et al., "Cooperative trust relaying and privacy preservation via edge-crowdsourcing in social Internet of Things," *Generation Computer Systems*, 2017.

[5] S. Vishal et al., "Computational offloading for efficient trust management in pervasive online social networks using osmotic computing," *IEEE Access*, vol. 5, pp. 5084–5103, 2017.

[6] S. Vishal, Y. Ilsun, and K. Ravinder, "Isma: Intelligent sensing model for anomalies detection in cross platform osns with a case study on iot," *IEEE Access*, vol. 5, pp. 3284–3301, 2017.

[7] V. Wyk and Hartman, "Automatic network topology detection and fraud detection," U.S. Patent No. 9,924,242. 20 Mar. 2018.

[8] A. Favila and P. Shivam, "Systems and methods for online fraud detection," U.S. Patent Application No. 15/236,077, 2018.

[9] C. Phua, V. Lee, K. Smith, and R. Gayler, "A Comprehensive Survey of Data Mining-based Fraud Detection Research," 2010.

[10] C.-C. Chiu and C.-Y. Tsai, "A web services-based collaborative scheme for credit card fraud detection," in *Proceedings of the e-Technology, e-Commerce and e-Service, 2004. EEE'04. 2004 IEEE International Conference on*, IEEE, 2004.

[11] K. K. Sherly and R. Nedunchezhian, "Boat adaptive credit card fraud detection system," in *Proceedings of the Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*, pp. 1–7, 2010.

[12] K. RamaKalyani and D. UmaDevi, "Fraud detection of credit card payment system by genetic algorithm," *International Journal of Scientific & Engineering Research*, vol. 3, no. 7, 2012.

[13] A. Kundu, S. Panigrahi, S. Sural, and A. K. Majumdar, "Blast-ssaha hybridization for credit card fraud detection," *IEEE Transactions on Dependable and secure Computing*, vol. 6, no. 4, pp. 309–315, 2009.

[14] A. Srivastava, A. Kundu, S. Sural, and A. Majumdar, "Credit card fraud detection using hidden markov model," *IEEE Transactions on dependable and secure computing*, vol. 5, no. 1, pp. 37–48, 2008.

[15] Nune, G. Kumar, and P. Vasanth Sena, "Novel Artificial Neural Networks and Logistic Approach for Detecting Credit Card Deceit," *International Journal of Computer Science and Network Security (IJCSNS)*, 2013.

[16] E. Aleskerov, B. Freisleben, and R. Bharat, "Cardwatch: A neural network based database mining system for credit card fraud detection," in *Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr)*, 1997.

[17] S. Maes et al., "Credit card fraud detection using Bayesian and neural networks," in *Proceedings of the 1st International Naiso Congress on Neuro Fuzzy Technologies*, 2002.

[18] V. R. Ganji and S. N. P. Mannem, "Credit card fraud detection using anti-k nearest neighbor algorithm," *International Journal on Computer Science and Engineering*, vol. 4, no. 6, 2012.

[19] Y. G. Sahin and E. Duman, "Detecting credit card fraud by decision trees and support vector machines," *Proceedings of the International MultiConference of Engineers and Computer Scientists 2011*, 2011.

[20] M. Zareapoor, K. R. Seeja, and M. Afshar Alam, "Analysis on credit card fraud detection techniques: based on certain design criteria," *International Journal of Computer Applications*, vol. 52, no. 3, 2012.

[21] V. V. Vlasselaer et al., "APATE: A novel approach for automated credit card transaction fraud detection using network-based extensions," *Decision Support Systems*, vol. 75, 2015.

[22] A. Shen, R. Tong, and Y. Deng, "Application of classification models on credit card fraud detection," in *Proceeding of the 2007 International Conference on Service Systems and Service Management*, IEEE, 2007.

[23] "2007 International Conference on Service Systems and Service Management," pp. 1–4, 2007.

[24] Y. Zhang, F. You, and H. Liu, "Behavior-based credit card fraud detecting model," in *Proceedings of the 2009 Fifth International Joint Conference on INC, IMS and IDC*, pp. 855–858, 2009.

[25] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud: A comparative study," *Decision Support Systems*, vol. 50, no. 3, pp. 602–613, 2011.

[26] F. N. Ogwueleka, "Data mining application in credit card fraud detection system," *Journal of Engineering Science and Technology*, vol. 6, no. 3, pp. 311–322, 2011.

[27] Z. Massimiliano et al., "Credit card fraud detection through parenclitic network analysis," *Complexity*, 2017.

[28] M. Carminati, L. Valentini, and S. Zanero, "A Supervised Auto-Tuning Approach for a Banking Fraud Detection System," in *Proceeding of the International Conference on Cyber Security Cryptography and Machine Learning*, Springer, Cham, Switzerland, 2017.

[29] P. Yamini, "Credit Card Fraud Detection using Deep Learning," *International Journal of Advanced Research in Computer Science*, 2017.

[30] F. Fadaei Noghani and M. Moattar, "Ensemble Classification and Extended Feature Selection for Credit Card Fraud Detection," *Journal of AI and Data Mining*, vol. 5, no. 2, pp. 235–243, 2017.

[31] G. Maryam and S. Mohammad Abadeh, "Fraud Detection of Credit Cards Using Neuro-fuzzy Approach Based on TLBO and PSO Algorithms," *Journal of Computer & Robotics*, vol. 10, no. 2, pp. 57–68, 2017.

[32] M. Sorkun Cihan and T. Toraman, "Fraud Detection on Financial Statements Using Data Mining Techniques," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 5, no. 3, pp. 132–134, 2017.

[33] M. Kamboj and G. Shankey, "Credit Card Fraud Detection and False Alarms Reduction using Support Vector Machines," *International Journal of Advance Research, Ideas and Innovations in Technology*, vol. 2, no. 4, 2016.

[34] F. Kang et al., "Credit Card Fraud Detection Using Convolutional Neural Networks," in *Proceeding of the International Conference on Neural Information Processing*, Springer International Publishing, 2016.

[35] M. Khodabakhshi and M. Fartash, "Fraud Detection in Banking Using Knn (K-Nearest Neighbor) Algorithm," *International Conference on Research in Science and Technology*, 2016.

[36] S. Kamaruddin and R. Vadlamani, "Credit Card Fraud Detection using Big Data Analytics: Use of PSOAANN based One-Class Classification," in *Proceeding of the International Conference on Informatics and Analytics*, ACM, 2016.

[37] S. Sharmila and S. Panigrahi, "Use of fuzzy clustering and support vector machine for detecting fraud in mobile telecommunication networks," *International Journal of Security and Networks*, vol. 11, no. 1-2, pp. 3–11, 2016.

[38] M. Carminati et al., "BankSealer: A decision support system for online banking fraud analysis and investigation," *Computers & Security*, vol. 53, pp. 175–186, 2015.

[39] M. Bansal and Suman, "Credit card fraud detection using self organised map," *International Journal of Information & Computation Technology*, vol. 4, no. 13, pp. 1343–1348, 2014.

[40] S.-J. Yen and Y.-S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5718–5727, 2009.

[41] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: a new over-sampling method in imbalanced data sets learning," *Advances in intelligent computing*, pp. 878–887, 2005.

[42] K. E. P. Baksai, *Feature Selection to Detect Patterns in Supervised and Semi Supervised Scenarios, Ph.D. thesis*, Pontificia Universidad Católica de Chile, 2010.

[43] L. Jundong et al., "Feature selection: A data perspective," *ACM Computing Surveys (CSUR)*, vol. 94, 2017.

[44] F. Bagherzadeh-Khiabani et al., "A tutorial on variable selection for clinical prediction models: feature selection methods in data mining could improve the results," *Journal of clinical epidemiology*, vol. 71, pp. 76–85, 2016.

[45] C. François, "Deep Learning with Python," 2017.

[46] D. M. W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 4672072, 15 pages
https://doi.org/10.1155/2018/4672072

Research Article

Analysis and Evaluation of SafeDroid v2.0, a Framework for Detecting Malicious Android Applications

Marios Argyriou,[1] **Nicola Dragoni**,[1,2] **and Angelo Spognardi**[3]

[1]*DTU Compute, Technical University of Denmark, Denmark*
[2]*Centre for Applied Autonomous Sensor Systems, Örebro University, Sweden*
[3]*Dipartimento Informatica, Sapienza Università di Roma, Italy*

Correspondence should be addressed to Angelo Spognardi; spognardi@di.uniroma1.it

Received 16 March 2018; Revised 2 July 2018; Accepted 12 August 2018; Published 5 September 2018

Academic Editor: Karl Andersson

Android smartphones have become a vital component of the daily routine of millions of people, running a plethora of applications available in the official and alternative marketplaces. Although there are many security mechanisms to scan and filter malicious applications, malware is still able to reach the devices of many end-users. In this paper, we introduce the SafeDroid v2.0 framework, that is a flexible, robust, and versatile open-source solution for statically analysing Android applications, based on machine learning techniques. The main goal of our work, besides the automated production of fully sufficient prediction and classification models in terms of maximum accuracy scores and minimum negative errors, is to offer an out-of-the-box framework that can be employed by the Android security researchers to efficiently experiment to find effective solutions: the SafeDroid v2.0 framework makes it possible to test many different combinations of machine learning classifiers, with a high degree of freedom and flexibility in the choice of features to consider, such as dataset balance and dataset selection. The framework also provides a server, for generating experiment reports, and an Android application, for the verification of the produced models in real-life scenarios. An extensive campaign of experiments is also presented to show how it is possible to efficiently find competitive solutions: the results of our experiments confirm that SafeDroid v2.0 can reach very good performances, even with highly unbalanced dataset inputs and always with a very limited overhead.

1. Introduction

The breakthrough of smartphones occurred some years ago thanks to the combination of telephony and Internet in a single portable machine, offered in many flavors, like Android, iOS, Windows Phone, and Blackberry. Nowadays, Android is by far the dominant platform, with an average worldwide market share of 85% [1]. It is acknowledged that the number of Android-based devices skyrocketed due to the open-source approach, as opposite to the closed approach followed by its competitors. That means high availability of applications outside the official marketplaces (Google Play) that Android users can freely download and install in their devices [2]. Beside being a great source of revenue for device manufacturers and app developers, availability of the apps also represent a gold opportunity for malicious developers that can hide and include harmful pieces of code (i.e., malware) in their apps [3], in order to evade the security checks of the official markets [4].

To face the challenge of Android malware detection, researchers have invested many efforts in designing systems and mechanisms to detect malicious activities in Android apps, like critical data leakages or unintentional hidden functions. The proposed solutions range within many different techniques, starting from signature-based detection, to artificial intelligence, leveraging static, dynamic, and hybrid analysis [5]. As further explored in the following sections, the most adopted approach among the detection proposals is the use of machine learning classifiers, while the most distinguishing element is the choice of the features to be considered for the classification that vary from app requested permissions and system commands to leery API calls. A

promising and also effective solution considers performing static analysis of the app binary code and relying on their different usage of API calls and packages to classify malicious and benign apps. With this direction in mind, we aim to advance an effective proposal for Android malware detection.

Contribution and Outline of the Paper. In this paper, we present SafeDroid v2.0 ([6] is a preliminary version of this paper), a complete framework to discriminate benign applications from malicious ones. A collection of scripts, a database, a web server, and an Android application form the SafeDroid v2.0 framework. These components pull together firmly in order to analyse, classify, and, eventually, decide whether an Android application is of benign or malicious nature.

The core of the framework is constituted by machine learning classifiers that are the base of real-time malware detection service for Android devices. Our solution introduces a complete, automated system for static analysis of Android applications, data extraction and storage, and comparison of different machine learning classifiers and algorithms, as well as visualisation of the produced results. The main target audiences of this framework are both the security research community and individual Android users. Our framework can be employed by Android malware analysts as an out-of-the-box solution to analyse and investigate the structural elements of malware. To foster further research activity, our framework supports many hyper-tuning parameters, which are evaluated throughout the detection process, that can be used to further explore the solution space and produce state-of-the-art and specialised models. We will show that the framework is fast, scalable, easy to use, and able to produce results of high accuracy, since it has been evaluated during an extensive campaign of experiments.

Our SafeDroid v2.0 framework is an elegant open-source solution that can also be used to provide a detailed overview and comparison of different classification algorithms and finally choose the best available model. The main novelty of SafeDroid v2.0 is the high level of automation of the testing procedure that includes application analysis, statistical analysis, feature vector extraction, service update, and feedback presentation to the user. This allows the security experts to explore many different choice of settings and options easily and extensively: the framework, in fact, supports user parameterization in the sense of allowing the user to choose exactly the conditions that the dataset has to fulfil. Eventually, the produced models can be tested in an integrated back-end server that comes along a light-weight Android application.

The rest of the paper is organised as follows. Section 2 presents related researches on the topic of Android malware detection. Section 3 illustrates the architecture of the proposed framework and a detailed overview of our methodology. We evaluate the performance of our proposal in Section 4 and we finally conclude the paper in Section 5.

2. Related Work

There are three main approaches that try to tackle the problem of malware detection. We classify the approaches based on the detection techniques they employ. We can have the static approach, which analyses the code of an application, the dynamic approach, which inspects the app behaviour at run time, and, finally, the hybrid approach, namely, a combination of the two previous methodologies. We classify and discuss the academic literature based on such three categories.

2.1. Detection Based on Static Analysis. Fuchs *et al.* [7] followed a modular analysis approach to discriminate malicious applications from benign ones. They created a tool called `ScAndroid` which allowed incremental checking of application as they are installed on the device. This tool solely extracts security specification from manifests that come along the applications and checks whether data flows are consistent with a set of predefined specifications. It uses static analysis to extract the appropriate information and makes automated security decisions based on the data flows such as deciding if it is safe to grant certain permissions to an application. Alternatively, it can pass security control to the user by providing relevant context like issued certificates and offline reviews.

Aafer *et al.* in [8] and Xiangyu *et al.* in [15] performed frequency analysis technique of the extracted API calls. In [8], the feature set was refined to use only API calls invoked by third-party applications. The APIs came mostly out of advertisement, web tracking, and web analysis packages. As the next step, they clustered the APIs by invocation methods (application specific, third-party package, or both). Their dataset consisted of 20,000 applications, 19.9% of which were malicious, and they were able to achieve 97.8% accuracy using machine learning algorithms. As a conclusion, they present statistics to argue in favour of the superiority of the API-based performance as compared to permission models. The same motif was engaged in [15] but produced opposite results. In this paper, the permission-based model managed to achieve 84.4% accuracy with the help of machine learning classifiers in contrast with the 70% that was achieved with the use of API-based model.

Ali-Gombe *et al.* in [9] followed the same mind-set. They concentrated efforts on op-code sequence analysis. After the Feature Extraction phase, the signature extraction produced a three-level cluster containing API calls made within the application, API call sequences from method signatures, and finally a collection of all method signatures for each class separately. F-Score, recall, and precision metrics were used to evaluate their model. Although they managed to get decent values (97.5%, 98%, and 97%, respectively) they argue that all their techniques could be circumvented with simple obfuscation techniques.

In another relevant publication of Bhattacharya *et al.* [10], the researchers focused on feature elimination techniques that would allow them to remove irrelevant features and thus limit the length of the feature vector set. The collection of the features is extracted from the Android Manifest file as they consider only the permissions. The feature elimination is carried out to define a subset of significant installation-time features to use in model construction. The main goal of this phase is to find a minimum set of features such that the

resulting probability distribution of data class is as close as possible to the original distribution created with all features. The limited size of the explored database (100 benign and 70 malicious) allowed the creation of 4 databases with different number of features, 83, 10, 5, and 2. K-fold validation (k=10) produced an accuracy score of 77%.

In [11], published in 2016, Yang *et al.* employed a new approach to analyse Android malware, namely, static analysis with intensive feature engineering. The authors consider features from different sources and different levels of the application. They create 5 feature sets. 3 of the feature sets come from the static analysis of the Dalvik Executable of the app, from the binary, the assembly, and the API levels of abstraction. The remaining 2 are acquired by analysing the Android Manifest file over the binary and information levels. Finally, the 5 feature sets are combined to form the final feature vector that is used for the training of the machine learning algorithm. This approach was evaluated over a dataset of 1,100 apps, out of which 50% was malicious. The research produced 98.1% detection accuracy and 8% of false positive rate, using the Random Forest classifier for a number of 50 trees.

In 2017, Fereidooni *et al.* proposed `ANASTASIA` [12], a system to detect malicious applications through statically analysing applications' behaviour. Although the techniques did not differ much from the previously discussed ones, they had the advantage of obtaining a big dataset of 11,187 malicious and 18,677 benign applications. For the model training procedure, they used 9 different classification algorithms, namely, XGboost, Adaboost, RandomForest, SVM, KNearestNeighbors, Logistic Regression, Naive Bayes, Decision Tree, and Deep Learning classifiers. The produced feature vector included 560 items and consisted of intents, permissions, system commands for root exploits, API calls, and malicious activities. They managed to perform fine tuning of the classification hyperparameters and thus achieve 97.3% accuracy on their training set.

In another work of 2017, Martín *et al.* in [13] developed a string-based malware detection mechanism employing supervised learning. Their study considered application binary information and permission-based features to estimate the nature of Android apps. The authors discuss the advantage of the string-based model over traditional signature-based ones, stating that it can deal with the different malware without the need to disassemble or run processes. To evaluate their solution, the authors compared the accuracy results with common antivirus engines. They employed the Bagging methodology to train the model and achieved a 97% discrimination accuracy, a result better than the 83% that the best antivirus engine had to offer.

The research of Milosevic *et al.* [14], published in 2017, utilised both classification and clustering techniques to discriminate between malware and benign apps in a tool called `Seraphimdroid`. The researchers formed a feature set that consisted of permissions and API calls. Android apps were first decompiled and then a text mining classification based on bag-of-words technique was used to train the model. In order to evaluate the performance of the Seraphimdroid tool, the researchers conducted experiments for permission-based and code-based clustering and classification. The results for the permission-based approach showed an F1 score of 87.1% and 64.6% correctly clustered instances while the results for the source-based approach produced maximum F1 accuracy of 95.8% and 82.3% correctly clustered instances.

In 2018, Wang *et al.* [16] extracted app characteristics found both in string values and structural features. Requested permission and API calls are acquired through the string values while function call nodes are inferred from the structural features. As a proof of point, experiments were performed on a data set of 2682 Android applications, out of which 1296 were malicious. The final results were obtained by weighting the respective detection results of the two types of features, achieving 97.9% accuracy in terms of F1 score.

2.2. Detection Based on Dynamic Analysis. Kuester *et al.* in [17] created a tool to monitor and collect relevant system events on the users' devices, namely, `MonitorMe`. The formality was to represent malware behaviour as actions while each set of actions represents a word. Collected words eventually will form a pseudo language. Every captured system call of each application is assigned a label, based on the component number and the intention of the action itself, and is sent back to the server responsible for determining the nature of the applications. Additionally, MonitorMe framework defines policies against privilege escalation attacks which aim on gaining elevated access to protected resources. The tool recognises malicious payload launches that occur after system boot. Among the drawbacks of this approach was that the device needs to be rooted throughout the procedure and the high percentage of false positives against true positive values, at 28% and 93.9%, respectively.

Burguera *et al.* in [18] try to challenge the obfuscation problem. They developed a crowdsourcing framework to collect samples of execution traces of applications, namely, `CrowdDroid`. Its main functionality lies on monitoring the underlying Linux kernel system calls and reporting the findings back to a centralised server. The primary goal is to identify different behaviours in applications that share identical names. The server then creates a dataset of behavioural data for each application. They cluster each dataset, using partitioning clustering algorithms, to differentiate between benign applications that demonstrate similar system call patterns to malicious ones.

Enck *et al.* in [19] demonstrated a tool named `TaintDroid` for real-time analysis by leveraging Android's virtualized execution environment. Their goal was to track the flow of private sensitive data through third-party applications, based on the assumption that downloaded third-party apps are unreliable. The tool monitors how these applications access and manipulate users' personal data. In order to do so, it observes which data hit the outgoing interface and their respective path by labelling them. The tool logs the data labels, the application responsible for transmission as well as the destination. The researchers observed 30 third-party applications found in Google Market and they discovered that the 65% of them transmitted sensitive data without user consent.

2.3. Detection Based on Hybrid Analysis. Sharid *et al.* in [20] focused on analysing application native code, monitoring, and detecting abnormalities during runtime. Native code refers to machine code that is executed directly by the main processor, opposed to bytecode, which is executed in the virtual environment of the Android JVM. Their static analysis focuses on defining certain patterns in the control flow. The patterns target ostensibly insignificant incidents that could pass otherwise unobserved. Their tool, `DroidNative`, is also able to evaluate bytecode by making use of the Android runtime. ART is the managed runtime that is used by applications and system services to compile bytecode and execute the Dalvik Executable format. In this case, ART is used to compile bytecode into native code for the purpose of analysis. Their experiment dataset consists of 5,490 applications, 1,758 of which are malwares. The two experiments produced contrasting results; the first one had 98.5% accuracy with a variation of 1% while the later ranged from 70% to 80%. The false positive rates showed high values for both, the lowest being around 20% and the highest up to 40%.

A case study by Poeplau *et al.* in [21] revealed that around 10% of the applications found in the official Android market are loading external code at runtime. This percentage rises further when one refers to the most popular applications. The primal intention of the research was to find commonalities among the dynamic loading of code by creating a tool to automatically detect loading attempts using static analysis techniques. The efforts were gathered around examining Java class loaders, package contexts, code disassembling, and ART. After finding that code-loading mechanisms are employed by the majority of benign applications, the authors suggested security policy reformation concerning dynamic code-loading allowance.

`DroidScreening` [22] framework scans applications at runtime to identify malicious patterns of execution. The solution followed the standard objectives of the antivirus industry, that are the malware sample screening and the identification of threat level. `DroidScreening` employed both static analysis to extract key identifiers of the targeted Android application to train the machine learning model and lazy associative classification algorithms to produce the classification model. The novelty of the solution lays on the trigger dynamic execution-based analysis environment that generates various system-wide events to trigger malicious activity. Their dataset numbered 1554 malware and 2446 benign applications. `DroidScreening` scored an F1 value of 91.1%. In addition to F1 score, the authors crafted a metric for the average misclassification cost, because they state that the cost of different misclassification errors should not have the same effect on the final accuracy result.

The idea behind the work of Bae *et al.* [23] was a detection system that monitors network usage, network connections, APIs, and permissions of an Android application. The first two were observed dynamically at run time while the latter were statically extracted by the application. The novelty behind this idea was to tackle the advent of new types of malware threats rather than focusing only on the already identified families of malware. The choice of building a hybrid solution is defined by observing that dynamic analysis adds significant amount of overhead to the device and static methods are easily evaded by obfuscation. The authors complimented the two methods to maximise the detection accuracy and minimise the introduction of high overhead functions. SVM classifier was employed to produce the machine learning model. The evaluation tests, conducted on a dataset of 413 benign and 398 malicious applications, produced 90.3% precision rate and a false positive rate of 13.9%. Moreover, the solution introduced a maximum overhead of 22.7% at the Android device.

2.4. Limitations of the Existing Approaches. The detection based on static analysis does not consume many resources and produces the most accurate results. From the negative side, there are techniques that can obfuscate the results, such as code encryption and dynamic load of code. Dynamic analysis techniques are more sophisticated but add a serious amount of computation overhead. They surpass static analysis in being resilient to evasion, code obfuscation, and polymorphism. They run as daemons processes, constantly monitoring the device for abnormal execution patterns or unprivileged information flow. Thus, they consume more resources, in terms of power and bandwidth, which leads to the degradation of the user experience. The observation of vague patterns produces high false positive rates, which makes dynamic analysis unfeasible in real-time scenarios. While it seems the most promising approach for future detection mechanisms, hybrid analysis is still in premature stages of development.

Finally, we can see how all the proposed solutions are not easily customisable, but are always provided as final solutions. Given the dynamic and rapidly changing field of malware creation, it is desirable to have an automated framework that allows to experiment and test with many different combinations of machine learning classifiers, with a high degree of freedom and flexibility in the choice of features to consider, dataset balance and selection. This can also give any attentive researcher the possibility of experimenting and validating unsolicited ideas and not common combinations of parameters.

3. SafeDroid v2.0 Framework Architecture

In this section, we present the architecture of our framework, namely, how the components are related as well as the logical steps of the execution. To enforce flexibility in the framework, each component is designed to run independently from the others. This means that each one has its own self-standing functionality and that each step can be iterated, split, altered in any component, and fully customised, according to the parameters defined by the user.

Figure 1 presents an overall overview of SafeDroid v2.0, with the main components and how they logically depend on each other. The framework can be summarised in one Android app, responsible to interact with the user, five main components, and a database. The first component that operates on the original dataset is the Feature Extraction

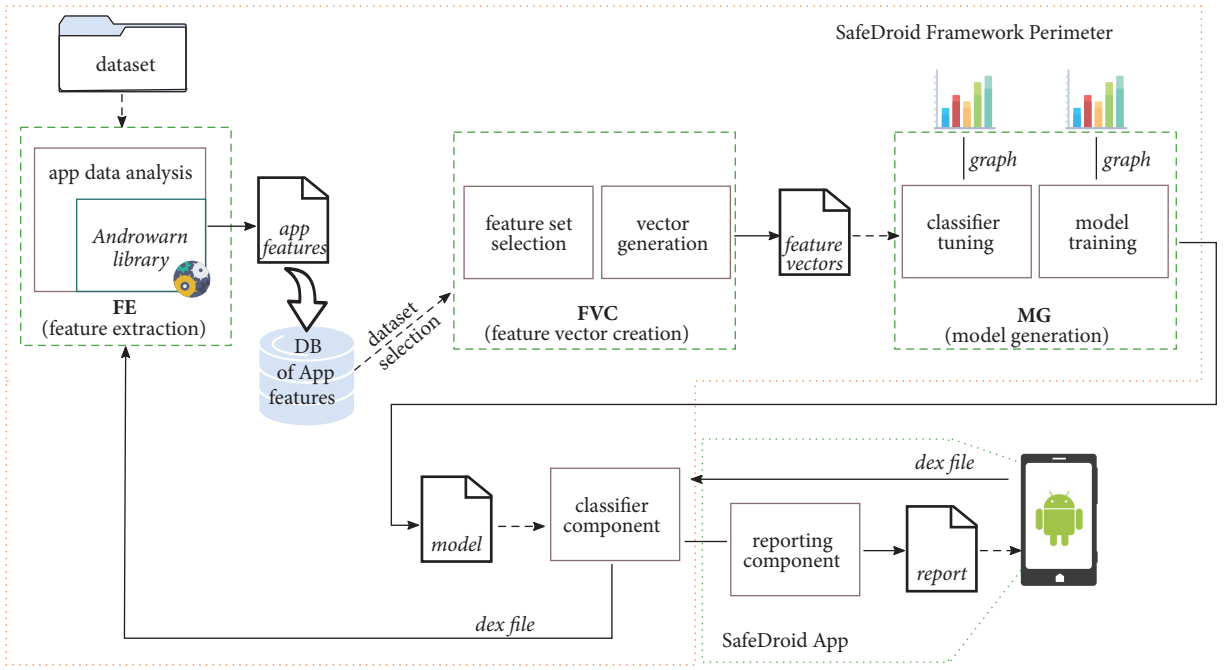


Figure 1: Overview of the SafeDroid v2.0 framework architecture, namely, the components and their interaction.

component that is responsible to reverse engineer the application binary code and to store the output in a database. The Feature Vector Creation component, which comes next, determines the most suitable features to use and builds the training and the testing sets for the Model Generation component. Several classifiers are generated and the most accurate one is selected as the model to be adopted in the SafeDroid v2.0 app. Finally, the model is exported to the Classification component that sends the final result of the analysis to the Reporting component that is the main interfaces with the Android users.

Before detailing the SafeDroid v2.0 framework, it is important to make an important premise about how the user privacy is taken into account when working with sensitive data as users' data and users' application preferences. SafeDroid v2.0 is a tool as many others proposed in literature: since the main focus of the security community is to protect against malware, the collection of user data and application list is an established practice in this research field. Thus, we can observe that we think that is important to protect the privacy of the users, but probably it is more important to protect her data against the malicious activity of malware (and then using all the possible resources for this aim).

As academic researchers, we provide as an additional guarantee of our *bona fide* several suggestions on how to address the privacy issue. First of all, we already released the full source code of our framework (https://github.com/Dubniak/SafeDroid-v2.0), together with the datasets and the parameters we used to perform the experiments presented in this paper: this means that it can be used directly by the Android user itself, without the need to give away its data to anyone and, still, check for the safety of its running or ready-to-install apps.

Secondly, we can easily imagine to include a transparent "anonymiser" element that addresses all those elements that could be used to mine the privacy of the user: the literature about this form of data-processing is huge and still growing, as an additional proof that the overall objective to obtain the security of the users is of outmost importance and can be pursued adopting the right tools that the state of the art offers.

In the following, we detail each single component and describe its main functionality together with the interaction among each component.

3.1. Feature Extraction. The Feature Extraction (FE) component is responsible for the analysis of the input apps. The main source of the apps for building the knowledge of the system is a dataset of labelled apps (malicious or benign). However, after the initialisation of the system, new apps are provided by the user for analysis and detection purposes, by means of the SafeDroid v2.0 app.

As a static-analysis-based framework, the role of the FE component is to extract all the useful data from the binary code of a given app such as identification information (name, alias, date, check-sums, etc.) and primitives that highlight the use of the app (API, permissions, third packages, etc.). The app analysis is performed by the FE component employing the functionality of `AndroWarn` (https://github.com/maaaaz/androwarn). AndroWarn is an open-source tool for reverse engineering of Android applications. The Feature Extraction component leverages an optimised revision of the library to perform a static analysis of the Dalvik bytecode of an application, which is represented as a Smali (intermediate file format between Dalvik Executable and Java source code). As detailed in Figure 2, the user has to provide as input a dataset of applications that will be

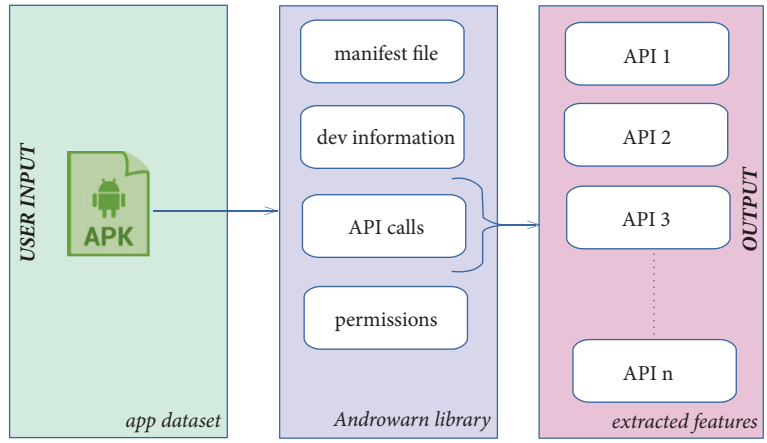


Figure 2: Overview of the Feature Extraction process.

processed by the FE component. Each app is analysed by the optimised AndroWarn library and all the APIs are extracted from the bytecode, to become the extracted features.

Since the app analysis is generally time and resource expensive (as detailed in Section 4), the result of the reverse engineering is permanently stored in another component of SafeDroid v2.0, the database, together with a label that designates the nature of an app. In this way the framework can realise a large dataset of applications that can be continually expanded, as new apps arrive at the FE component.

As in any other machine learning solution, the dataset constitutes the knowledge base for the whole framework, from the training to the testing phase up to the detection mechanism and all the other SafeDroid v2.0 components rely on it to perform their functions: for example, the Feature Vector Creation generates the input for the Model Generator (namely, feature vectors) evaluating the extracted features and their usage among the considered apps, while the Model Generator builds the classifiers leveraging the feature vectors. The distinction between malicious and benign applications managed by the FE component is critical for the correct outcome of the whole detection process.

3.2. Feature Vector Creation. The Feature Vector Creation component (Figure 1, shortened to FVC from now on) performs three main functions: it extracts the features from the dataset of apps (accessing to the DB), then it finds the most discriminatory features the detection mechanism will later use to identify the nature of any app, and, finally, it generates a feature vector for each app that will be the input for the next components. An overall picture of the involved steps is depicted in Figure 3. The FVC is tailored in such way to allow the user to choose the desired execution parameters, such as the size and the balance of the dataset: the high efficiency of the whole evaluation process allows to build several experiments that can be used to empirically evaluate and compare several possible solutions. Since the main target of SafeDroid v2.0 is the security research community, we give freedom to the user by providing a framework with many parameters that supports unconstrained execution modes. For example, one execution could be set to consider only a small part of the dataset, another one to consider only the dominant malwares, i.e., those apps that have the most distinguishable behaviour. As soon as the app dataset is selected and the parameters for the analysis are provided, the SafeDroid v2.0 framework is able to automate all the following steps, up to the generation of the model. However, this setup is, eventually, one of the most critical steps. Thus, users should acknowledge potential biases to the results that are induced by the choice of the execution parameters. However, as many other frameworks publicly released, our tool can only help the experts exploring several (possibly optimal) solutions, but we can clearly miss the *best* solution, as part of the human bias. In some sense, we could say that we can help removing the human bias, providing a framework that makes testing in an automated way several settings in parallel possible, but we can not eventually exclude a human bias or mistake in principle.

For this reason, the SafeDroid v2.0 allows the creation of multiple datasets in order to prepare the knowledge base for different experiments. Each dataset will be possibly composed of different features (namely, APIs), such that it will be able to generate different app classifiers. The whole procedure can be iteratively repeated when new applications are added to the whole dataset, to generate updated versions of the feature sets that, in turn, will generate new, updated versions of the app classifiers.

We will now detail all the steps performed by the FVC component in order to generate the feature vectors for the apps in the dataset. We start considering the dataset D the user provides for the Feature Selection phase.

Given a dataset of applications D, we can see it as the union of two distinct subsets, namely, $D = B \cup M$, where B and M are the subset of the *B*enign and the subset of the *M*alicious applications, respectively. The idea is to build a feature set leveraging the frequency of each API in every app in the dataset. The frequency is deducted by dividing the occurrences of each API over the total occurrences that the

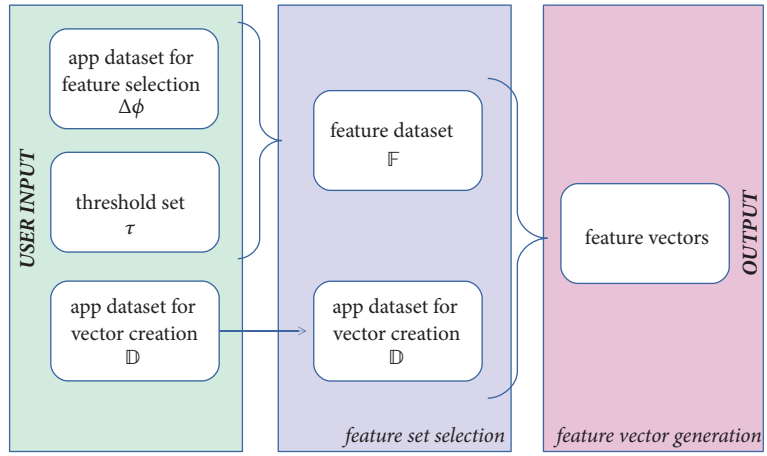


Figure 3: Overview of the Feature Vector Creation processing steps.

specific API has in the dataset, for both the malicious and the benign ones.

More formally, if we denote as App_X ⟵ API.a the fact that application App_X makes use of the API API.a, we can define a sort of indicator function (from Wikipedia: an *indicator function* is a function defined on a set X that indicates the membership of an element in a subset A of X, having the value 1 for all elements of A and the value 0 for all elements of X not in A) $\mathbf{1}_X(a)$ as

$$\mathbf{1}_X(a) = \begin{cases} 1 & \text{if App\_X} \longleftarrow \text{API.}a, \\ 0 & \text{else} \end{cases} \quad (1)$$

and, then, we can evaluate the frequency of API.a in the subset of the benign applications ϕ_a^B as

$$\phi^B(a) = \frac{1}{b} \sum_{X \in B} \mathbf{1}_X(a) \quad (2)$$

where b is the number of benign applications, namely, $b = |B|$. We can similarly define

$$\phi^M(a) = \frac{1}{m} \sum_{X \in M} \mathbf{1}_X(a) \quad (3)$$

as the frequency of API.a in the subset of the malicious applications, where m is the number of malicious applications, namely, $m = |M|$.

With the two values $\phi^B(a)$ and $\phi^M(a)$ we can, then, evaluate the difference

$$\Delta\phi(a) = \left|\phi^B(a) - \phi^M(a)\right| \quad (4)$$

that denotes if a given API.a is more frequent in one of the two subsets: informally, we can tell that the higher its $\Delta\phi$, the higher the power of an API to discriminate as malign or benign, an application that uses such API.

To perform the feature set selection, we set a threshold t for the value of $\Delta\phi$ and we consider only those API a such that $\Delta\phi(a) \geq t$. In this way, we are actually selecting a subset of the whole set of the APIs, with a discriminating power that depends on t. Such a subset of APIs can be used as a feature set to build a classifier for the dataset of our applications. Furthermore, if we select more than one threshold, we will have more feature sets. SafeDroid v2.0, indeed, given in input a set of k thresholds $\tau = \{t_1, \ldots, t_k\}$, is able to generate k different feature sets that can be used for the next steps of the classifier generation. In particular, given a set of thresholds $\tau = \{t_1, \ldots, t_k\}$ as input from the user, the framework generates a family $\mathbb{F}$ of feature sets as

$$\mathbb{F} = \bigcup_{t \in \tau} \{a \mid a \longleftarrow \text{App\_X} \in D \text{ and } \Delta\phi(a) >= t\}; \quad (5)$$

that is, for each threshold t in τ, it generates a new set with those APIs that belong to any of the apps in D, such that the $\Delta\phi$ of those API is greater than or equal to the threshold t. Each of those new sets is included in the family of feature sets $\mathbb{F}$. We can informally say that, for each value of a threshold, the framework resolves a feature set, namely, a set of APIs that tries to describe the behaviour of an application.

The last operation of the FVC is the generation of the relevant binary vector for a given app, namely, a representation of an app suitable to input the machine learning algorithms and generate a classifying model. The idea is to have a vector encoding the use of each API of the feature set for every app: each API is assigned a dimension of the vector, where there is a 1 if the corresponding API in the feature set is used by the app, and 0 otherwise. For this purpose, the FVC considers each app in the dataset, one by one, and checks if it uses any of the APIs of the selected feature set, again one by one.

For example, suppose to have an application App_X which uses the set {API.a, API.c, API.d, API.e, API.f} of APIs. Given the feature set {API.a, API.b, API.c, API.g}, we would obtain a binary feature vector

$$\text{FV(App\_X)} = 1\ 0\ 1\ 0 \quad (6)$$

In this example, the first dimension is 1, since it corresponds to API.a and this API is used by App_X, the second dimension is 0 because API.b is not used by App_X, and so

on. The number of the produced feature vectors depends on the number of apps in the dataset provided in input and the number of thresholds in τ, while their length will depend on the feature sets in $\mathbb{F}$.

Besides the option to receive a static dataset as input, SafeDroid v2.0 helps the setup of a batch of experiments since it is able to randomly generate a family of app datasets, with different combinations of parameters: namely, the user can specify any combination of dataset sizes (s) and balance (r) between malicious and benign applications, and the framework will generate datasets with the given characteristics. More formally, given a set ρ of pairs (s, r), where s is the expected size of the dataset and r is the expected ratio of the malicious applications in the dataset, the framework will produce a family $\mathbb{D}$ of app datasets as

$$\mathbb{D} = \bigcup_{(s,r)\in\rho} A_{s,r} \tag{7}$$

where, for each $A_{s,r} \in \mathbb{D}$, we have that

$$\left|A_{s,r}\right| = s, \tag{8}$$

$$\frac{\left|A_{s,r} \cap M\right|}{\left|A_{s,r}\right|} = r \tag{9}$$

Each dataset in $\mathbb{D}$ will be used as input for all the phases (Feature Selection, Feature Vector Creation, and the following steps described next) and will produce different results and classifiers. We just recall that the generation of $\mathbb{D}$ is very efficient because it is produced leveraging the application database, built in the previous Feature Extraction phase.

3.3. Model Generation. The goal of the Model Generation component (Figure 1, MG from now on) is to choose the best classification method for a given dataset of (D) received in input and to build the relative classifier (trained model) [24]. This is represented in Figure 1 by the *classifier tuning* and *model training* steps, since the MG evaluates several different machine learning classifiers and their most effective classification tuning parameters to maximise the accuracy metrics and minimise the mis-classifications. Again, we stress that the choice of the machine learning classifiers and the parameters are full responsibility of the user, since SafeDroid v2.0 is a framework to support the analysis of several possible settings of options and parameters.

The MG component supports three modes of execution, namely, `optimum feature vector`, `optimum dataset`, and `unattended`. In the optimum dataset mode, the dataset is split in subsets, as defined by the user. The framework then trains the prediction model based on the optimum slice of the dataset that is the one that provides the highest accuracy and lowest negative results. In the optimum feature vector mode, the framework creates multiple feature sets based on the user choice. Then, it adopts the one that optimally describes the nature of the apps. It is crucial to notice that the final accuracy results are obtained by testing on the whole dataset. In the unattended mode of execution, the framework does not apply any optimisation algorithm.

The methodology adopted for SafeDroid v2.0 is the standard machine learning trial-and-error approach [24] and the selection is done comparing the outcome of two procedures: a standard cross-validation experiment and a greedy look-up with a split of the input dataset in training/testing subsets. The parameters for both the procedures are set by the user (number of folds for the cross-validation and the training/testing set ratio).

In our experiments, the SafeDroid v2.0 framework employs the functionality of the following classification algorithms: K-Nearest Neighbor, Support Vector Machine, Random Forest, Decision Tree, Multilayer Perceptron, and Adaptive Boosting [25]. In particular, the framework uses the cross-validation technique to produce accuracy results of all classifiers and then compares the cross-validation scores of the different classifiers with the greedy look-up procedure. If the two values are different (e.g., the classifier algorithm), SafeDroid v2.0 reinitiates the procedure to produce one final classifier that will be used to produce the prediction model. At this stage, graphs that visualise both the learning and the training procedures are displayed to provide feedback to the user.

The findings of the previous phase (i.e., the classification algorithm along the tuning parameters) are loaded into the Model Generation component. During this last phase, the SafeDroid v2.0 inputs the tuned classifier to the constructor of the prediction classifier and creates the prediction model that will be employed in the classifier component.

3.4. Classification and Report Components. The Classification component (CC) works together with the Report component (RC). The CC is responsible for identifying and detecting malicious apps, while the RC task is to handle the interaction with the user. The CC receives the input data (namely, MD5,.dex) from the Android app and uses the precalculated feature set of APIs to create the binary feature vector, employing the functionality of the Feature Extraction component. The binary vector is finally tested on the prediction model and a new label is generated for the new app. This output is used to generate a report, with a feedback to be presented to the user, by means of the Android application. The report contains the features that were found to be considered malicious, so that the user can have an understandable feedback about the label given to the evaluated app. Additionally, the data extracted from the app are stored in the database component, in order to be used to update the prediction model.

3.5. SafeDroid v2.0 Android Application. The RC of the Android app retrieves the list of already installed applications on the device and sends the `dex` file of the objective application, along with the application name and the MD5 hash of the APK to the Classification and Report server. Finally, it presents the received report from the server to the user of the Android app.

Table 1: Composition of the datasets used in the experiments. The ratio column is the malicious app ratio in the considered dataset.

app dataset	size	ben. apps	mal. apps	ratio	APIs
Group 1	29344 kB	428	432	0.50	3491
Group 2	38306 kB	371	211	0.36	7007
Group 3	138613 kB	513	755	0.60	14327
Group 4	201897 kB	138	392	0.75	23229
Group 5	261358 kB	2843	755	0.20	37249
Group 6	313787 kB	1013	1611	0.60	26694
Group 7	342717 kB	1710	1611	0.50	32791
Group 8	451389 kB	3121	1611	0.33	50835
Group 9	972093 kB	3371	2481	0.40	94825

4. Evaluation

This section presents the results of our extensive experiments performed with our framework, that we recall is fully implemented and already released for further study to the research community (https://github.com/Dubniak/SafeDroid-v2.0.).

The purpose is to provide evidence about the SafeDroid v2.0 key characteristics, namely, feasibility of deployment, low computational cost, and detection accuracy. In particular, with the term feasibility we mean that the whole mechanism is actually feasible and easy to use. With low computational costs, we want to show that the whole detection process, together with the training and testing phase, is scalable and can be executed in real time. Finally, to show the detection accuracy, we want to report real use cases that confirm the capability of the SafeDroid v2.0 framework to distinguish between malicious and benign Android apps.

A total of 25,346 applications compose our *whole dataset*, out of which 20,208 (79.7%) are benign and 5,138 (20.3%) are of malicious nature. These applications were used as the knowledge base for our experiments. Part of our malware data has been kindly provided by the authors of a prior piece of research work [26]. The rest of the samples were collected from malware repositories such as `VirusShare`, `contagio Mobile`, `malware.lu` and applications dated between 2012 and 2016. The benign applications were downloaded from the official Google Play Store during the year 2013.

4.1. Feasibility of Deployment. The fully functional framework has been implemented and is available to be forked at the `github` hosting service. In particular, the version we used for the experiments in this paper has been developed with SDK v.24 on Android OS 7.0, using the `scikit-learn` and the AndroWarn library.

Given a fully functional prototype of the whole framework, we were able to carefully analyse and evaluate the single component that constitutes our implementation of SafeDroid v2.0, conducting a series of experiments. The experiments were conducted on 9 different input datasets, subsets of our initial whole dataset, over a series of 3 executions. The input datasets are detailed in Table 1 and were crafted to deliberately differ in both absolute number of apps, balance among malicious and benign apps, and number of employed APIs. Moreover, in order to test the feasibility of our framework, the settings of the execution were different in all sets of experiments. In this way we were able to acquire metrics to evaluate the computational costs of each architectural component and the efficiency of the produced prediction models on both balanced and unbalanced input datasets. We have to highlight that, since some of the experiments take a considerable amount of time, there are many external factors that influenced the framework, like the network connection to query the database, the race conditions among threads-pointers to the database, and so on. All the times we report in the following, where they are not differently stated, do not take into account such a fine grained detail, but simply report the single steps as a whole.

The experiments were conducted on a computing station with 24 CPUs (Intel® Xeon®) running at 2.76GHz with 64GB available memory.

We tested our framework on different number of processors to be able to estimate its overall performance in real-life scenarios and highlight the effect of parallelization. The base of our experiment employed 2 cores and doubled up, until to 16 cores. Table 2 presents the execution time in seconds to perform the whole Model Generation. It can be observed that the execution time decreases dramatically as the number of the employed cores increases. To grasp the overall improvement, we report the average execution time utilisation in Table 3. The results report the percentage of the enhancement of the execution time with respect to the previous setup (half the cores) and with respect to the original setup (2 cores). As the results show, as a harsh estimation, doubling the number of the cores results in decreasing the execution time by around 30%, while the overall reduction time reaches up to 65% for 16 cores.

4.2. Computational Costs. In this section we report our study to evaluate the computational costs of our SafeDroid v2.0, studying the time in seconds it takes to go through all the steps of the framework. In particular, we highlight the timing of the different phases as well as the improvement in speed we obtain using the database. To evaluate the framework in different settings, we designed a battery of tests. The experiments were conducted for 9 different input datasets (Table 1) over a series of 3 executions and they were conceived to highlight the costs of each phase of the framework and to provide evidences about the actual feasibility of the approach.

Table 2: Empirical time (in seconds) of the overall execution for each group.

app dataset	2 cores	4 cores	8 cores	16 cores
Group 1	210	112	67	38
Group 2	238	145	105	92
Group 3	1208	670	435	352
Group 4	1648	1322	1177	975
Group 5	2168	1790	1641	1314
Group 6	3396	2597	1910	1319
Group 7	4704	2577	1544	1108
Group 8	8610	4855	3293	841
Group 9	22498	15191	9820	7075

Table 3: Utilisation of execution time per core number.

number of cores	relative improvement	absolute improvement
2	-	-
4	34.7%	34.7%
8	28.5%	52.2%
16	30.4%	65.6%

To evaluate the costs of the Feature –API– Extraction phase (Section 3.1), which we recall to be based on the AndroWarn, we report in Table 4 the timing of the 3 executions for each group. The beneficial effects of the use of a database are all in the reduction of these timings. Moreover, to have an idea about how much the Feature Extraction phase impacts on the overall timing of the process, we draw in Figure 4 a plot for each of the 9 groups: they represent the timing of the single phases of the whole process, together with the overall timing, to have a grasp about how relevant such phase is in the whole process. The error bars in the Y-axis represent the variation between the slowest and the fastest execution of the several repeated experiments for each group. We can observe that, for all the groups, the Feature Extraction phase is always the predominant component for the overall execution time, being almost always close to 95% of the overall cost. Moreover, as the size of the apps dataset increases, the other phase that increases its time consumption is the Tuning one, where the framework tries different tuning parameters in order to maximise the results of the different machine learning algorithms. The Feature Vector Creation and the Training phase are always very limited in their time consumption.

We recall that the computational cost of the Feature Extraction phase is caused by the reverse engineering of the input apps: for example, in the case of an input of around 1GB (Group 9 of Table 1), it occupies most of the CPU, reaching at a peak of almost 21,000 seconds. We can also observe that the execution time of the Feature Extraction phase follows a linear increase towards the size of the input dataset.

The above results motivate our decision to include a database for permanently storing the results of the Feature Extraction for each app. In this way, the most costly operation is substituted by a simple database access, reducing by several

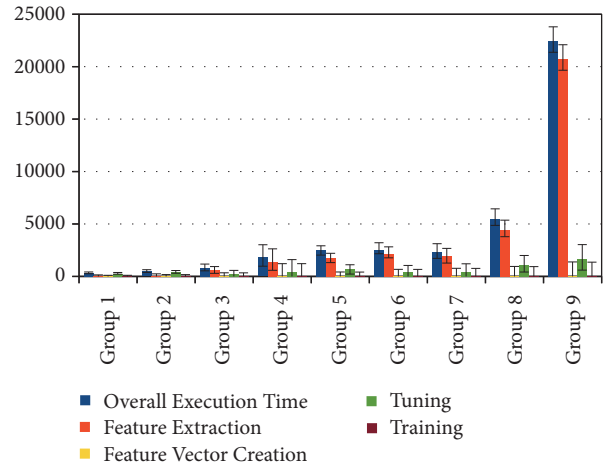


Figure 4: Comparison between execution timings, with the detailed time of each single phase. Times are in seconds.

order of magnitude the overall cost. This design choice follows the initial goal to create a framework that can be really used to assist security researchers and able to produce results quickly. We report in Table 5 the comparison between the overall time execution of the whole generation process, respectively, without and with the use of the database. In Figure 5, instead, the overall execution times are compared, again with and without the employment of the database, in order to have a visual representation of the improvement. The noticeable difference between the first—during which the reverse engineering of the input apps is taking place—and follow-up executions is visible for all sizes of the input. We can observe that as the size of the dataset to process increases, the advantage of the database increases even further: for example, for the smallest dataset (Group 1) it goes from around 5.1 minutes (316 sec.) without the database to 4.5 minutes (278 sec.) with the use of the database, with a saved time of around 11% of up to almost 92% for the largest dataset (Group 9), from 6.2 hours (22422 sec.) to 28 minutes (1734 sec.). Then, we can conclude that the adoption of the database makes extensive use of the framework—even with very large datasets—possible.

4.3. Detection Accuracy. Next, we explored the achievable detection rates of our SafeDroid v2.0 framework. The graph

Table 4: Feature Extraction timings for each group of apps. The results consider 3 runs for each group. Times are in seconds.

	Average	Min	Max	StDev
Group 1	41.59	32.85	58.14	14.34
Group 2	73.81	58.58	104.22	26.34
Group 3	597.58	301.07	922.16	311.50
Group 4	1414.33	645.07	2489.35	959.40
Group 5	1792.47	1451.24	1990.99	296.83
Group 6	2151.28	1801.20	2801.05	563.28
Group 7	1905.05	1216.81	2703.02	749.16
Group 8	4428.34	3466.56	5729.38	1168.94
Group 9	20734.27	19578.27	22275.95	1389.58

Table 5: Overall execution time comparison, without and with the use of the database. Times are in seconds.

	without DB		with DB		
	Average	StDev	Average	StDev	saved time
Group 1	316.05	91.08	278.24	76.76	11,96%
Group 2	467.76	148.72	396.30	122.41	15,28%
Group 3	837.53	324.16	245.52	12.66	70,69%
Group 4	1804.24	1075.82	391.67	120.77	78,29%
Group 5	2506.04	447.69	730.24	231.79	70,86%
Group 6	2541.09	579.78	401.92	25.46	84,18%
Group 7	2354.42	708.86	464.69	46.78	80,26%
Group 8	5507.69	827.99	1101.75	582.74	80,00%
Group 9	22442.46	1239.60	1734.36	797.11	92,27%

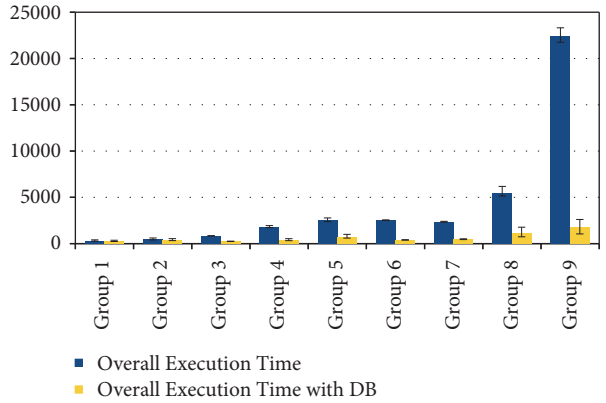


Figure 5: Comparison between the overall execution times, with and without the use of the feature database. Times are in seconds.

in Figure 6 shows the accuracy rates in terms of false positive rate, precision, specificity, and F1 values for the three different modes of execution (Section 3.3). The false positive rate is the ratio of false positive number over the total number of malicious apps in the datasets and it shows the percentage of falsely characterising benign apps as malicious ones. The specificity shows the proportion of benign malware that is correctly identified as benign and the precision refers to the framework ability of correctly identifying malware samples. The F1 score is used to measure the accuracy of the prediction model. The framework produced highly accurate results while it managed to confine the false positive rates. The results are in adequate level, acquiring values up to 99.5% for the positive indices. It is interesting to observe the framework behaviour when the input dataset is the group 4. As we recall from Table 1, group 4 refers to the highest unbalanced dataset, which consists of 75% malicious and 25% benign apps. In this case, the prediction model identified almost 93% of the cases while the false positive rate climbed up to almost 12.5%. Overall, the framework scored the highest accuracy for the optimum feature vector mode of execution.

The ability of the framework to choose the most accurate classification algorithm among the 6 available candidates can be observed in Figure 7. For the purpose of this experiment, we recorded the results for the 9 input datasets in terms of F1 score. The red colour denotes the framework prediction for the classification algorithm with the highest accuracy. The framework successfully chose the optimum classifier in all experiments, allowing the optimum alternative for the prediction model. As discussed in Section 3.3, the choice of the algorithm is taken by comparing the cross-validation scores that are produced after the exhaustive search of the tuning parameters for each candidate. The final scores, which are displayed in Figure 7, refer to the prediction models that are exported after the training and testing procedure (Figure 1). As so, the results provide a strong insight on the versatility of the fine tuning module; the vast majority of the prediction model scored values above 95%.

In order to observe the ability of the framework to produce prediction models on big input dataset, we conducted 3 experiments on the full population of the dataset that

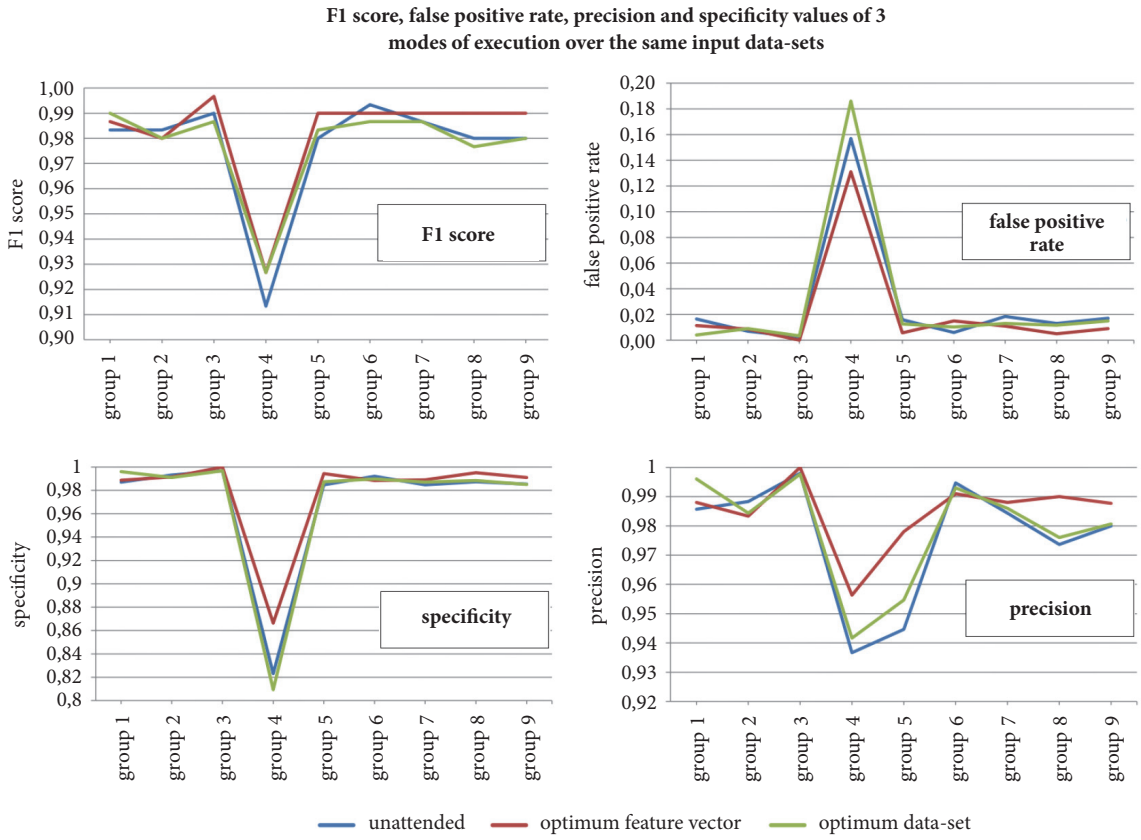


Figure 6: Accuracy scores for three different modes of execution: unattended, optimum feature vector, and optimum dataset.

we obtain (25346 apps, 20.3% malicious), for a series of 3 executions. The base hypothesis of our experiment is that the framework can produce acceptable prediction rates for both limited and extended feature sets. The choice of the feature is the key to characterise an app as malicious or benign. For the purpose of this experiment, the framework created 3 distinctive feature sets, subgroups of the initial set. The length of the feature sets along the prediction rates is presented in Table 6. The available feature space for the whole dataset is around 1.4 billion unique features. In the 3 consecutive cases, the framework relied on 157, 1,879, and 2,428 features, respectively, to construct the prediction models. The average F1 accuracy ranges from 96.9% for 157 features to 98.9% for 2428 features while the false positive rate is constrained in between 0.6% and 1.4%, respectively. Moreover, the punctual choice of the most discriminatory features is inferred from the results; the framework produced results of high accuracy for a feature set consisting of 157 characteristics of malicious apps. While the feature set expands, the framework gives higher accuracy values and lower false positives.

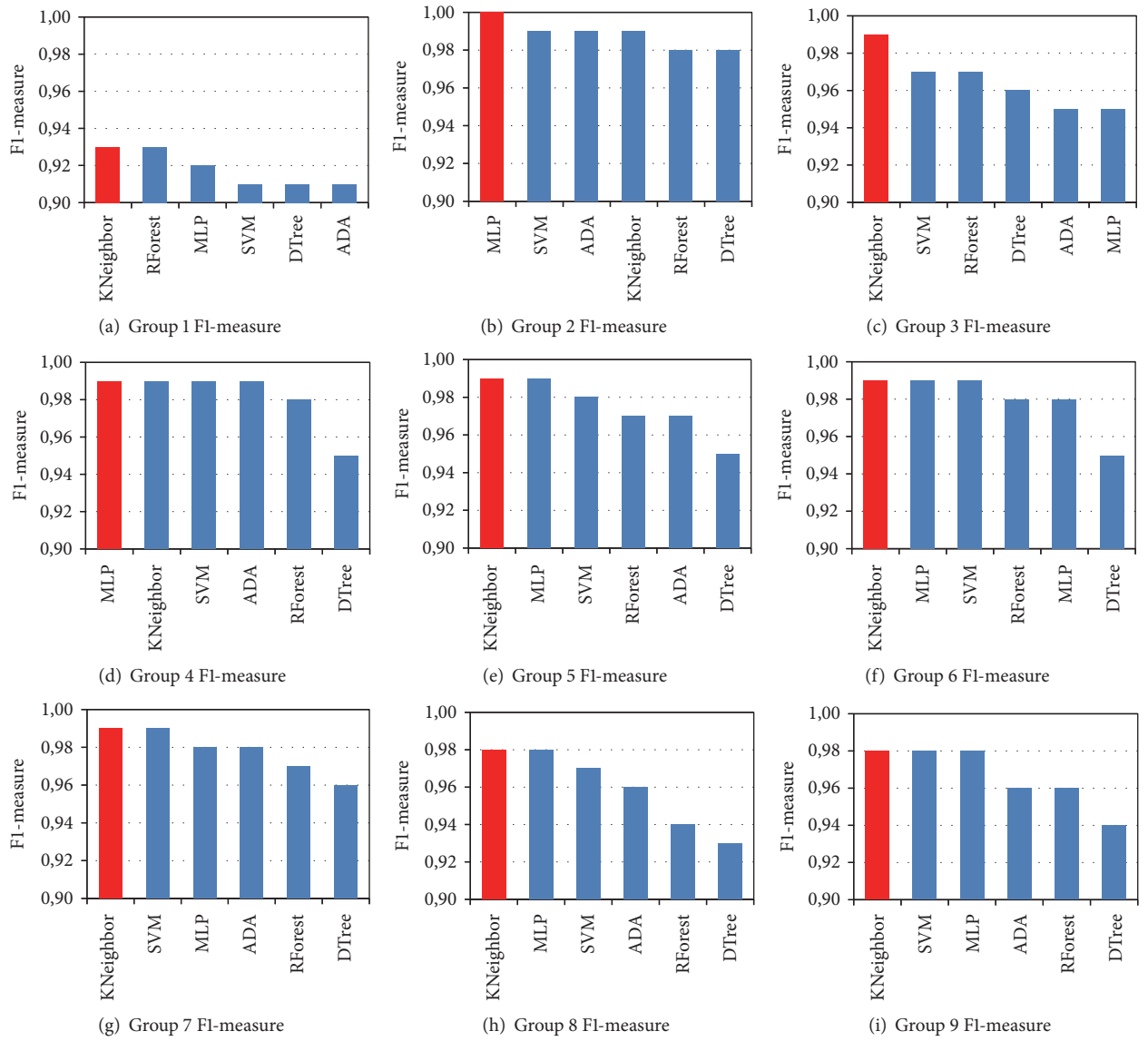
4.4. Performance Comparison. Table 7 compares our proposed solution with the most prominent alternatives as the state of the art of the literature. All compared frameworks follow the static analysis approach to produce their results. The table displays the F1 and False Positive Rate, which are the most common metrics to measure the performance of a framework, the number of the employed machine learning algorithms, and, finally, the ability of the framework to be employed for further security research. The dataset column refers to the population of the examined applications, while the malicious column refers to the proportion of malicious applications within the dataset. We use these two columns to point out the dataset imbalance in the compared methods. As we observed, a lower proportion of malware in the dataset results in a higher F1 accuracy and lower false positive rate. Feature column introduces the attributes that were used to examine the nature of the applications. All of the aforementioned metrics were employed by most of the related researchers and have been evaluated as the best candidates to measure the real performance of the solutions.

In order to compare the other solutions with our framework, we used the prediction scores that were produced by using the full dataset with the highest dimensionality of the feature space (Table 6). Although the number of malicious applications in our dataset is slightly lower than the average

Table 6: Summary for 25,346 apps and 1,4 billion features.

case	features	F1 score	FPR∗
1	157	0.969 ± 0.01	0.014 ± 0.002
2	1879	0.976 ± 0.009	0.012 ± 0.002
3	2428	0.989 ± 0.006	0.006 ± 0.001

∗FPR: False Positive Rate.



(a) Group 1 F1-measure (b) Group 2 F1-measure (c) Group 3 F1-measure

(d) Group 4 F1-measure (e) Group 5 F1-measure (f) Group 6 F1-measure

(g) Group 7 F1-measure (h) Group 8 F1-measure (i) Group 9 F1-measure

Figure 7: F1-measure obtained by the framework for the 9 input datasets. In red, the classification algorithm is automatically selected by the framework.

Table 7: Comparison with similar solutions. ML reports the number of tested machine learning algorithms; FSR means further security research feature. PRM refers to *permission based analysis*; INT refers to *application intent*; STR refers to *string-based analysis* (Section 2).

API based framework	additional features	ML	FSR	dataset size	mal. apps	F1	FPR
DroidApiMiner [8]	—	4	no	19978	19.9%	97.8%	2.2%
OpSec [9]	PRM	n/a	no	1758	79.5%	97.5%	2.2%
DMDAM [10]	PRM	15	no	170	41.2%	77.0%	n/a
IFE [11]	—	1	no	1100	50.0%	98.1%	8%
Anastasia [12]	PRM,INT	9	no	29864	62.5%	97.3%	5%
SBMD [13]	STR,PMR	6	no	10000	50.0%	97.0%	10%
SeraphimDroid [14]	PRM	5	no	400	50.0%	87.1%	n/a
SafeDroid v2.0	—	6	yes	25346	40.0%	98.9%	0.6%

number that the other solutions employed, the difference in the false positive rate is noticeable. Moreover, our method produced the highest accuracy in terms of F1 score, reaching up to 98.9%. The comparison table also shows a lack of out-of-the-box, customisable solutions in the literature: our framework is the only one available to be employed for further research activities.

5. Conclusion and Future Work

In this paper we presented SafeDroid v2.0, a complete framework for Android apps, to discriminate benign applications from malicious ones. Unlike prior research and other offered solution in this field, our work introduces an open-source, easy to deploy, research framework. Our primary intention was to offer mobile security researchers a technical solution able to conduct the analysis, the formation of the prediction models, and the classification of Android applications. The core of the framework is constituted by machine learning classifiers that are the base of real-time malware detection service for Android devices. Our solution introduces a complete system for static analysis of Android applications, data extraction and storage, comparison of different machine learning classifiers and algorithms, as well as visualisation of the produced results. We also presented a set of experiments showing that the framework is accurate, fast, scalable, and easy to use.

As future work, we aim at implementing a fully automated framework to make SafeDroid v2.0 self-updating, adding a new component that will continuously look up for new malware applications to be analysed and triggers the generation of a new and updated version of the classifier. Moreover, we are planning to perform additional experiments employing different families of malware and different versions of the same malware, in order to prove that SafeDroid v2.0 can also be employed to study the evolution of the different malwares over time. Finally, we are also considering adding another component to complement the static analysis with a dynamic analysis of the apps under analysis.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] IDC International Data Corporation, "Smartphone OS Market Share 2017-Q1," http://www.idc.com/promo/smartphone-market-share/os, 2017.

[2] B. Mike, Android's Piracy Problem Is Forcing Developers To Give Away Games: 'Alto's Adventure' Latest Freebie, 2012.

[3] Y. Zhou and X. Jiang, "Dissecting android malware: characterization and evolution," in *Proceedings of the 33rd IEEE Symposium on Security and Privacy*, pp. 95–109, San Francisco, Calif, USA, May 2012.

[4] J. Oberheide and C. Miller, "Dissecting the android bouncer," in *Proceedings of the SummerCon2012*, New York, NY, USA, 2012.

[5] A. Damodaran, F. D. Troia, C. A. Visaggio, T. H. Austin, and M. Stamp, "A comparison of static, dynamic, and hybrid analysis for malware detection," *Journal of Computer Virology and Hacking Techniques*, vol. 13, no. 1, pp. 1–12, 2017.

[6] R. Goyal, A. Spognardi, N. Dragoni, and M. Argyriou, "SafeDroid: A distributed malware detection service for android," in *Proceedings of the 9th IEEE International Conference on Service-Oriented Computing and Applications, SOCA'16*, pp. 59–66, Macau, China, November 2016.

[7] A. P. Fuchs, A. Chaudhuri, and J. S. Foster, "Scandroid: automated security certification of android applications," CS-TR-4991, UM Computer Science Department, 2009.

[8] Y. Aafer, W. Du, and H. Yin, "DroidAPIMiner: mining API-level features for robust malware detection in android," in *Security and Privacy in Communication Networks*, vol. 127 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pp. 86–103, Springer, 2013.

[9] A. Ali-Gombe, I. Ahmed, G. G. Richard, and V. Roussev, "OpSeq: android malware fingerprinting," in *Proceedings of the 5th Program Protection and Reverse Engineering Workshop*, ser.PPREW-5, pp. 7:1–7:12, ACM, New York, NY, USA, December 2015.

[10] A. Bhattacharya and R. T. Goswami, "DMDAM: data mining based detection of android malware," in *Proceedings of the First International Conference on Intelligent Computing and Communication*, J. K. Mandal, S. C. Satapathy, M. K. Sanyal, and V. Bhateja, Eds., pp. 187–194, Springer, Singapore, 2017.

[11] M. Yang and Q. Wen, "Detecting android malware with intensive feature engineering," in *Proceedings of the 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, pp. 157–161, Aug 2016.

[12] H. Fereidooni, M. Conti, D. Yao, and A. Sperduti, "ANASTASIA: android malware detection using static analysis of applications," in *Proceedings of the 8th IFIP International Conference on New Technologies, Mobility and Security, NTMS 2016*, pp. 1–5, Larnaca, Cyprus, November 2016.

[13] A. Martín, H. D. Menéndez, and D. Camacho, "String-based malware detection for android environments," in *Intelligent Distributed Computing X*, vol. 678 of *Studies in Computational Intelligence*, pp. 99–108, Springer International Publishing, Cham, Switzerland, 2017.

[14] N. Milosevic, A. Dehghantanha, and K.-K. R. Choo, "Machine learning aided Android malware classification," *Computers and Electrical Engineering*, vol. 61, pp. 266–274, 2017.

[15] Xiangyu-Ju, "Android malware detection through permission and package," in *Proceedings of the 2014 International Conference on Wavelet Analysis and Pattern Recognition, ICWAPR 2014*, pp. 61–65, Lanzhou, China, July 2014.

[16] W. Wang, Z. Gao, M. Zhao, Y. Li, J. Liu, and X. Zhang, "DroidEnsemble: Detecting Android Malicious Applications with Ensemble of String and Structural Static Features," *IEEE Access*, vol. 6, pp. 31798–31807, 2018.

[17] J.-C. Kuester and A. Bauer, "Monitoring real Android malware," in *Runtime Verification*, vol. 9333 of *Lecture Notes in Computer Science*, pp. 136–152, Springer International Publishing, Cham, Switzerland, 2015.

[18] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: behavior-based malware detection system for android," in *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM '11)*, pp. 15–26, ACM, New York, NY, USA, October 2011.

[19] W. Enck, P. Gilbert, S. Han et al., "TaintDroid: an information flow tracking system for real-time privacy monitoring on smartphones," *ACM Transactions on Computer Systems*, vol. 32, no. 2, pp. 5:1–5:29, 2014.

[20] S. Alam, Z. Qu, R. Riley, Y. Chen, and V. Rastogi, "DroidNative: Automating and optimizing detection of Android native code malware variants," *Computers & Security*, vol. 65, pp. 230–246, 2017.

[21] S. Poeplau, Y. Fratantonio, A. Bianchi, C. Kruegel, and G. Vigna, "Execute This! Analyzing Unsafe and Malicious Dynamic Code Loading in Android Applications," in *Proceedings of the Network and Distributed System Security Symposium*, pp. 23–26, San Diego, Calif, USA, 2014.

[22] J. Yu, Q. Huang, and C. Yian, "DroidScreening: a practical framework for real-world Android malware analysis," *Security and Communication Networks*, vol. 9, no. 11, pp. 1435–1449, 2016.

[23] C. Bae and S. Shin, "A collaborative approach on host and network level android malware detection," *Security and Communication Networks*, vol. 9, no. 18, pp. 5639–5650, 2016.

[24] T. M. Mitchell, *Machine Learning*, McGraw-Hill, Inc, New York, NY, USA, 1st edition, 1997.

[25] P. Srikanth, A. Singh, D. Kumar, A. Nagrare, and V. Angoth, "A comparison of machine learning classifiers," in *Advanced Materials and Information Technology Processing*, vol. 271-273, pp. 149–153, Trans Tech Publications, 2011.

[26] H. Kang, J.-W. Jang, A. Mohaisen, and H. K. Kim, "Detecting and classifying android malware using static analysis along with creator information," *International Journal of Distributed Sensor Networks*, vol. 11, no. 6, Article ID 479174, 2015.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 7243296, 8 pages
https://doi.org/10.1155/2018/7243296

Research Article

Detecting Potential Insider Threat: Analyzing Insiders' Sentiment Exposed in Social Media

Won Park, Youngin You, and Kyungho Lee

Institute of Cyber Security & Privacy, Korea University, Seoul 02841, Republic of Korea

Correspondence should be addressed to Kyungho Lee; kevinlee@korea.ac.kr

Received 9 March 2018; Revised 2 June 2018; Accepted 26 June 2018; Published 18 July 2018

Academic Editor: Ilsun You

In the era of Internet of Things (IoT), impact of social media is increasing gradually. With the huge progress in the IoT device, insider threat is becoming much more dangerous. Trying to find what kind of people are in high risk for the organization, about one million of tweets were analyzed by sentiment analysis methodology. Dataset made by the web service "Sentiment140" was used to find possible malicious insider. Based on the analysis of the sentiment level, users with negative sentiments were classified by the criteria and then selected as possible malicious insiders according to the threat level. Machine learning algorithms in the open-sourced machine learning software "Weka (Waikato Environment for Knowledge Analysis)" were used to find the possible malicious insider. Decision Tree had the highest accuracy among supervised learning algorithms and K-Means had the highest accuracy among unsupervised learning. In addition, we extract the frequently used words from the topic modeling technique and then verified the analysis results by matching them to the information security compliance elements. These findings can contribute to achieve higher detection accuracy by combining individual's characteristics to the previous studies such as analyzing system behavior.

1. Introduction

We are living in the world of IoT. Everything from a portable device to a home device is being connected to the Internet. A statistic research predicted that above 30 billion devices would be connected to the Internet in 2020 [1]. Furthermore, the number of social media users all over the world is predicted above three billion in 2021 [2]. Social networking is a type of communication that has continued since the creation of mankind, although the forms are different. Also, the current form of social media is in line with the development of Internet and IoT. As the IoT paradigm expands, interaction with social media becomes more active than in the mobile environment, which is expanding to the concept of SIoT (Social Internet of Things) [3]. Also, social media can be used as an interaction tool with the IoT devices [4]. In this way, social media research based on IoT is going on in a variety of ways. As the IoT device develops, the insider threat of the organization is getting bigger. IoT technology expands to become a part of people's lives and provides convenience to people, but efforts and costs for information security are increasing as the number of connection points with existing systems grows [5] because cyber threat such as Denial of Service (DOS) and information leakage can be caused easily by normal or malicious insiders' mobile devices. Also, many studies about analyzing system behavior to detect insider threats have been conducted, but relatively few researches about analyzing emotions that affect individual behavior have been conducted. In order to detect threats, social media analysis is needed, which is one of the media in which individual tendencies are well presented.

In this research, we aim to find the possible malicious insider to prevent the insider threat. A research indicates that many organizations are in face of insider threats. According to the 2018 Insider Threat Report by Cybersecurity Insiders and Crowd Research Partners, 90 percent of organizations felt they were vulnerable to insider threats, and 56 percent answered that regular employees among insiders were the biggest threat [6]. In this situation, many organizations focus on reinforcing internal system. They tend to think that internal threats, as well as external threats, can be somewhat defensible if only information security solutions are built

well. However, C. Colwill insists that information security should not lean in only the technological solution. He cited the survey and found that 94% of UK government agencies had data encryption for wireless networks, 97% used internal firewall protection, and 95% used email scans for protect from external threat. On the other hand, 70% of the frauds were caused by insiders, but 90% of the security controls and monitoring are concentrated on responding to external threats [7]. In this situation, N. Safa, R. Solms, and S. Furnell [8] have an idea that organizations ignoring individual security problem are possible to fail. Furthermore, they indicated that Information Security Organizational Policies and Procedures (ISOP) can improve employee's information security level. The basic theory of their research is Social Bound Theory (SBT) and Involvement Theory. SBT is the theory that the more individuals interact with others, the less likely they are to behave abnormally. In other words, it can be judged that a person who actively exchanges in an organization is unlikely to be a malicious insider and will not cause an insider threat. Involvement theory is that people who actively participate in certain activities are less likely to behave abnormally. There is some overlap with SBT, but it is assumed that the more active people are, the less likely they are to cause insider threats. In short, we are focusing on lowering the risk from insider threats especially by analyzing social media. Through this research, we aim to contribute to the detection of insider threats by combining the sentiment analysis of individuals with the concept of information security compliance. In addition, we expect to be able to detect malicious insiders with higher accuracy through combination with previous studies analyzing system behavior.

2. Related Works

There are many researches about the usage of social media to prevent threats in both real and cyber space. In terms of the insider threat, there are two kinds of research areas: technological and psychological. The first is researches through technological analysis. M. Salem, S. Hershkop, and S. Stolfo [9] presented a methodology for detecting malicious insiders through host and network-based user profiling. Host-based user profiling has provided a way to identify users in environments such as UNIX, Windows, and the web. Network-based user profiling provides a way to identify users by analyzing network traffic such as HTTP, SMB, SMTP, and FTP. Also, they set the criteria for detecting a "masquerader" and an "internal traitor" through two analysis methods. Masquerader is a type of malicious insider who steals and impersonates legal insider. Internal traitor is another type of malicious insider who has been accepted to access the system. A. Harilal et al. [10] simulated malicious behavior in an organization's system. They collected them from seven types of data source: mouse, key strokes, host activities, network traffic, and so on. Through several steps of periods, they collected benign/malicious users' behavior dataset. This dataset gives a chance to utilize a research in malicious insider's behavior in the organization's system. These analytical methods have the merit that they enable the formal analysis, but it has a disadvantage that it is difficult to reflect the tendency of the individual. Therefore, applying the methodology presented in this study makes it possible to detect malicious insiders more effectively. P. Legg et al. introduced a tool for detecting malicious insider called Corporate Insider Threat Detection (CITD) [11]. They conducted profiling of user and role based on system logs such as logins, portable storage usage, and email transmission. This research showed the possibility of adapting decision making process and decreasing the false positive rate when a system detects malicious behavior. Lastly, A. Tuor et al. analyzed CERT Insider Threat Dataset v6.2 of Carnegie Mellon University [12]. They found that the Neural Network model has higher performance than the Principal Component Analysis (PCA), Support Vector Machine (SVM), and Isolation Forest based on anomaly detection. They modeled system's normal behavior and set abnormal behavior as criteria of anomaly. M. Bishop et al. approached as the aspect of process [13]. They analyzed insider's behavior into the process model. In the process model, specific behaviors are broken into the subprocesses. Researches in this part contributed to establish a theory to detect malicious insiders based on the system behavior. However, it is important to understand the insiders' emotional state in order to analyze why they do malicious acts.

In this context, there are researches through psychological analysis. M. Kandias et al. [14] proved that psychological characteristics of negative attitude are closely related to insider attacker's malicious behavior. They compared three kinds of methodologies according to the learning accuracy: machine learning approach, dictionary-based approach, and flat data classification. They used data of well-known streaming service YouTube's comments subscribers, and video views, and so on. Among these methodologies, machine learning approach showed the highest accuracy because many words are learned by learning model. On the other hand, words had to be searched entire list to find what kind of words is the same as the list in the dictionary-based approach. Furthermore, flat data classification is harder to find the same words. V. Marivate and P. Moiloa [15] showed how to detect crime incidents by gathering and analyzing social media data. To prove this, they checked how much is corresponded with the real incident and the estimated event. Many studies have focused on the analysis of behavior of both system and user in an organization. L. L. Ko et al. approached to the insider threat in various areas such as cyber behaviors and communication behaviors [16]. This research has the meaning for analyzing insider threats in perspective of not only the cyber behavior and communication, but also the biometric and psychosocial behavior. Furthermore, they suggested future directions of the insider threat related to their perspectives. B. A. Alahmadi, P. A. Legg, and R. C. Nurse approached in perspective of OCEAN personality analysis [17]. They proved the correlation between Internet usage log and personal trait. Through this research, they were able to identify psychological characteristics by trends in the Internet usage logs and ultimately use them to detect potential malicious insiders. Together with system behavior analysis, researches about psychological aspects contribute to overcome the limitations of technological approaches by understanding the emotions of insiders.

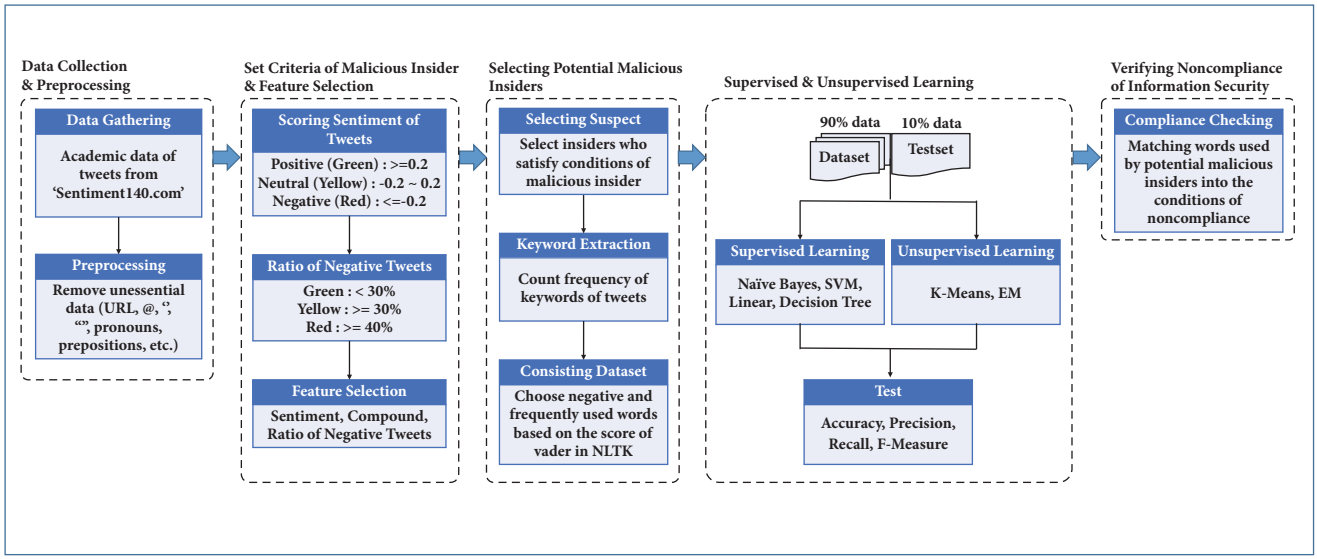


Figure 1: Overall process of the research.

3. Research Methods

In this section, we will introduce some methods which were adapted to analyze data. It is including description of the dataset, methodologies such as machine learning, sentiment analysis, topic modeling, and information security compliance. Overall process of this research is descripted in Figure 1.

3.1. Dataset. For this research's purpose to find a possibility of an insider threat, we needed to get data which are including many people's thinking. Training data from the web service "Sentiment140" is containing almost 1.6 million of tweets and the user who wrote tweets [18]. Among many social media platforms, Twitter is widely used because it can freely analyze using open API. Even though they already having service of sentiment analysis, we analyzed the sentiment of tweets again using open-sourced Python API called NLTK (Natural Language ToolKit). Originally, dataset of Sentiment140 has three levels of sentiment: negative (0), neutral (2), and positive (4). It is similar to the NLTK in terms of the kinds of sentiment. However, it is hard to understand the level of sentiment because it has only fixed sentiment score. To overcome the limitation, NLTK was used to get sentiment level more specifically (from -1 to 1).

3.2. Sentiment Analysis. NLTK is a well-known sentiment analysis tool in python language. It is a suit of program modules, data sets, and tutorials supporting research and teaching in computational linguistics and natural language processing [19]. Each word has an evaluated score used as criteria called "vader". NLTK calculate score to determine how positive/negative is the sentence's or document's sentiment level. We operated NLTK to overall tweets and got each word's sentiment level including overall, positive, negative, and neutral.

3.3. Topic Modeling. It is a statistical model used in natural language processing (NLP) to discover a subject of documents. Latent Dirichlet Allocation (LDA) is the basic topic model and is called a probabilistic model. It also has a generative process including hidden variables and defines a probability distribution [20]. It assumes that the words related to specific topic would appear more often than the other words. At first, all of sentences were broken into the words in corpus and all unnecessary words were deleted such as pronouns, prepositions, and articles.

3.4. Machine Learning. We should find the pattern of dangerous behavior of the insider threat. Both supervised and unsupervised learning algorithms were used because there are different advantages. In terms of the supervised learning, it has an advantage that we can give an answer of who has the possibility of doing dangerous behavior into the dataset. However, data labelling requires highly expensive, time-consuming works [21]. So, too much time could be consumed because almost 1 million of data were used. In the aspect of efficiency, unsupervised learning can be advantageous. Open-sourced machine learning software called "Weka" was used to compare the accuracy among several algorithms. The following are supervised/unsupervised learning algorithms.

3.4.1. Supervised Learning. Supervised Learning is useful when an exact answer is given. In this research, we labelled possible malicious insider. There are four supervised learning algorithms used in classification. Naïve Bayes is a generative model by using assumption to reduce the parameters to learn [22]. When there are n features, the number of parameters must be minimized since the example to be learned increases exponentially. Finally, it estimates the Naïve Bayes classifier which is most likely to be used as features. Support Vector

Machine (SVM) is a suitable algorithm to adapt text classification. In particular, it is being widely used to classify news stories and search result. T. Joachims [23] suggested the several advantages to apply the text classification and showed that SVM is the most accurate algorithm among other classification algorithms. Linear algorithm is normally used algorithm for a statistical case like predicting baby's body weight [22]. It has a loss function which calculates how accurately it predicts. So, Empirical Risk Minimization (ERM) is widely used to minimize the loss value. Lastly, in the Decision Tree algorithm, root makes branches from learned data and classifies each data through conditions of leaves. In terms of the NLP, this algorithm can be utilized to categorizing document, parts of speeches [24].

3.4.2. Unsupervised Learning. Unsupervised Learning is useful when an exact answer is not given. K-Means is a well-known algorithm so that it can adapt many unsupervised learning cases. Its basic principle is simple. First, select k centroids. Second, calculate cost value (distance) between each centroid and element. Lastly, adjust centroids' location and repeat these processes. Expectation-Maximization (EM) focuses on reducing log likelihood and its processes are executed repeatedly until mixture model is close to the current log likelihood [25]. There are two steps to deal with the data: E-step and M-step. E-step is a computing process to get expected value using currently predicted parameter and discovered data. M-step is a process of determining the value of the parameter using predicted data called "imputed" data. In this process, it assumes that the data of the expectation step is actually measured data [26]. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is another type of clustering algorithm. It has two kinds of parameters: Eps and MinPts. Eps means a distance between the core point and a point in a cluster. MinPts means the number of points. Cluster can be made when there are MinPts of points. It uses heuristic method to determine parameters and finds "thinnest" cluster that has the limit value of the maximum k-dist value [27].

3.5. Information Security Compliance. In the current IT environment, insider can cause massive harm and we cannot prevent or mitigate from the damage of malicious insiders. So, human factors should be considered to overcome the limit of technology-based information security compliance [7]. We reorganized research table of the information security compliance of [8] and matched each characteristic and used words. Through this process, we demonstrated that possible malicious insiders did not match the conditions of compliance of the information security.

4. Research Results

There are five steps to estimate possible malicious insider by analyzing collected tweets. After preprocessing, overall insiders' sentiment levels were scored and selected possible malicious insiders. Dataset was consisted based on the result of topic modeling. Machine learning algorithms have been used to classify and cluster possible malicious insiders. Finally, selected insiders who are in red group of possibility were adapted to the condition of noncompliance of information security and compared whether he/she corresponds to the condition of noncompliance. Similarly, W. Park, Y. You, and K. Lee [28] researched correlation between tweets and actual behavior in the real world. 4,000 Tweets of 2016 U.S. presidential election nominees (Donald Trump and Hillary Clinton) were crawled. After sentiment analysis process, daily events were correlated with the negative tweets based on daily average sentiment score. Through the machine learning process, this research proved that individual's social media reflects the real world's situation.

4.1. Preprocessing. The dataset of "Sentiment140" consisted of .csv file. It has user ID, date, tweets, and sentiment of tweet in the file. Some tweets have mathematical character "=". When this character is at the front of the sentence, excel program perceives it as a function. So, we had to remove "=" at the front of sentence. Also, we removed unessential words including the parts of sentence such as pronouns, prepositions, and articles. Through this process, we could get the groups of words and were erased by the topic modeling.

4.2. Analyzing Insiders' Overall Sentiment. At first, sentiment level was scored through the sentiment analysis of the whole data. After that, the average of the sentiment score and the ratio of negative tweets are calculated. Figures 2 and 3 show a part of analysis result. Figure 2 is about each user's sentiment score and Figure 3 is about the ratio of negative tweets. Each figure has a green line which means the criteria of the baseline of the negative sentiment. In this dataset, we classified 400 people depending on the criteria. Most insiders were in the green level, but 1 user (0.25%) was in the red level and 36 users (9%) were in the yellow level.

4.3. Selecting Possible Malicious Insider. In order to find a person who is possible to be a potential threat, we set criteria to classify people in three levels by sentiment score and the ratio of negative tweets. Table 1 shows how to classify people by the threat level. The degree of insider threat was divided into three stages: green, yellow, and red. Each level is assigned if both the sentiment score and the ratio of negative tweets are satisfied. The baseline for judging negative sentiment was set to -0.2 based on [28] and it was converted from [29]. Also, the ratio of negative tweets was set 40%. According to M. Losada and E. Heaphy [30], they separated three types of teams by efficiency: high, medium, and low-performance. In particular, in case of low-performance teams, ratio of positivity and negativity (P/N) is 0.363. Additionally, E. Ferrara and Z. Yang suggested ratio of the neutral messages in the social media [31]. They measured about 45% of messages had neutral emotion. Table 1 shows the criteria which we made to detect possible malicious insider through these researches. These criteria applied to classify the degree of insider threat, and the system administrator can flexibly respond when it comes to the system management by further

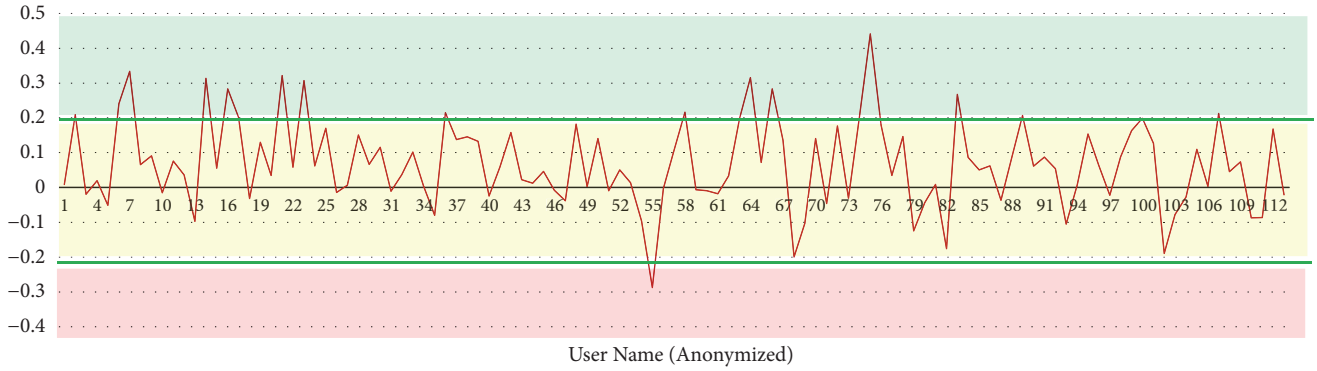


Figure 2: Example of the ratio of the negative tweets.

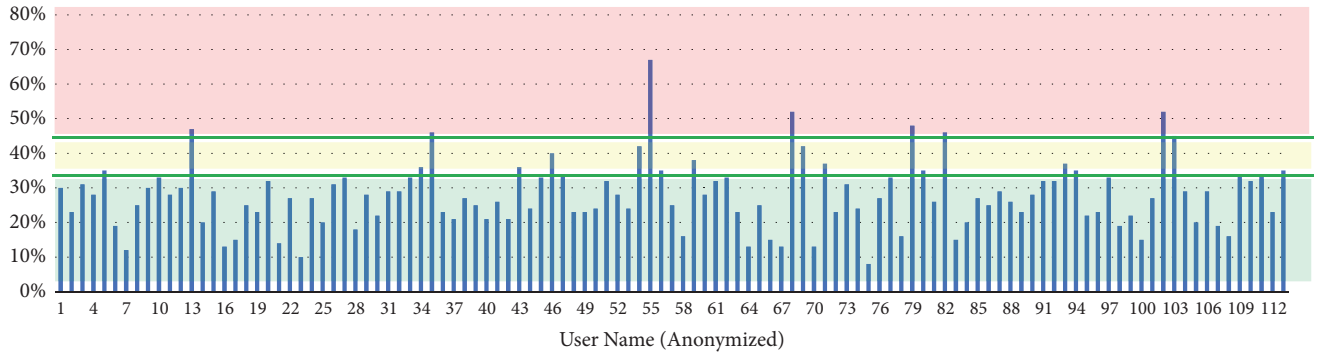


Figure 3: Example of the result of insiders' sentiment analysis.

Table 1: Three levels in classifying potential threats.

Level (Threat Possibility)	Sentiment Score (s)	Ratio of Negative Tweets (r)
Green (Low)	s >= 0.2	r < 35%
Yellow (Medium)	-0.2 < s < 0.2	r >= 35%
Red (High)	s <= -0.2	r >= 45%

Table 2: Comparing possible malicious suspects.

User (Anonymized)	Sentiment Score	Sentiment Level	Ratio of Negative Tweets
A	-0.2873	Red	67%
B	-0.1994	Yellow	53%
C	-0.1892	Yellow	52%
D	0.2409	Green	19%
E	0.2161	Green	16%

subdividing the criteria from the previous research. From these criteria, Table 2 shows five people of each level.

4.4. Machine Learning. As a way to detect malicious insider, Machine Learning methodology was used. Both supervised and unsupervised learning algorithms were adapted in this research and the result of learning is descripted in Sections 4.4.2 and 4.4.3. This process has a meaning as a way to identify possible malicious insiders and improve the accuracy. Topic Modeling was used to rank according to how often the words in tweets were used. And then, we listed 100 of negative words. Table 3 shows the example of dataset which was adapted to the research. We focused negative words because we assumed that people's negative thinking related to the threat would be presented by the words. To make a dataset, negative words were selected as features. Each of the features (word) and the number of how many times they were used were included in the dataset.

4.4.1. Feature Selection. There are so many words which consist all of tweets. Among them, we chose 100 of negative words by frequency. After that, we composed a dataset and checked if the word is in the sentence.

4.4.2. Supervised Learning. Several kinds of supervised learning algorithms were adjusted to analyze the accuracy of learning. Table 4 shows the result of supervised learning. Decision Tree had the highest accuracy of learning and is followed by SVM, linear, and Naïve Bayes. A kind of probabilistic model Naïve Bayes assumes that the population follows Gaussian or polynomial distributions, but the data in the research is not exactly of a distribution type, so it can have relatively low accuracy. The SVM can produce high accuracy when it classifies two categories by creating a nonprobabilistic

Table 3: Example of the dataset.

User	Sentiment Score	Ratio of Negative Tweets	Label	word1 (sad)	word2 (miss)	word3 (bad)	...	word100 (blocked)
A	-0.2873	0.67	Yes (1)	1	0	1	...	1
B	-0.1994	0.52	Yes (1)	0	0	1	...	0
C	-0.1892	0.52	Yes (1)	0	0	0	...	1
D	0.2409	0.19	No (0)	1	0	0	...	0
E	0.2161	0.16	No (0)	0	1	0	...	0

Table 4: Result of Supervised Learning.

Algorithm	Accuracy	Precision	Recall	F-Measure
Naïve Bayes	96.2%	1.000	0.962	0.980
SVM	99.5%	0.991	0.996	0.993
Linear	99.5%	0.994	0.995	0.994
Decision Tree	99.7%	0.998	0.997	0.996

Table 5: Result of Unsupervised Learning.

Algorithm	Accuracy	Incorrectness
K-Means	95.6%	4.4%
EM	83%	17%
DBSCAN	90.7%	9.3%

Table 6: Frequently used words.

User (Anonymized)	Sentiment	Words
A	Positive	can, packing, good, working, ...
	Negative	miss, bad, hate, wrong, hell, ...
B	Positive	like, want, awesome, hope, fine, ...
	Negative	damn, bad, sad, hell, dying, poor, ...

binary linear classification model, and the linear algorithm has similar characteristics. The Decision Tree algorithm has the property of iterating until a new prediction is no longer added because the process is iteratively recursive and can have a relatively high accuracy.

4.4.3. Unsupervised Learning. Several kinds of unsupervised learning algorithms were adjusted to analyze the accuracy of learning. Table 5 shows the result of unsupervised learning. K-Means had higher accuracy than EM and DBSCAN. Unsupervised learning result was relatively inaccurate because learning process occurs without information about who the malicious insider was. The K-Means algorithm can improve the accuracy of detection by minimizing the dispersion of the cluster and distance difference while forming a certain cluster. The EM algorithm can be used to obtain the maximum likelihood when the statistical model cannot be solved correctly, but it may not converge to the maximum possible degree. DBSCAN basically forms clusters based on eps and MinPts, but the dataset has some parts that do not fit well into the conditions.

4.5. Get Frequently Used Words. Topic modeling was used and ranked into the frequency of how often the words were used. We focused on negative words because we assumed that people's negative thinking related to the threat would be presented in the words. All the words in Table 6 were extracted by topic modeling from each user's tweets. Basically, words were classified to positive and negative based on NLTK's sentiment score database (vader_lexicon.txt). There are a lot of words and each score was evaluated by several people.

4.6. Possible Malicious Insider Selection. All of tweets are analyzed and classified within the criteria to detect the highly possible malicious insiders. Also, we only selected people who wrote at least 45 tweets. The words they used were matched to condition of the compliance. Finally, the most possible malicious insiders were selected. Table 7 shows the example of verifying insider threat based on the concept of information security compliance.

5. Conclusions

Social media can help people around the world communicate freely. Its role is also diversifying as the Internet environment has been expanding to the mobile and IoT-based. At the same time, we can use the social media as useful analysis media to detect the potential insider threat. In this paper, it has been shown that it is efficient to analyze individual tendencies to detect possible malicious insider. To do this, sentiment analysis was conducted on the collected tweets, and we classified the users by the level of threat according to criteria. And then, the classified possible malicious insiders were verified by performing information security compliance matching process. Machine learning algorithms were applied to detect possible malicious insiders. Decision Tree algorithm could detect possible malicious insiders with the highest accuracy of 99.7%. In addition, user A was expected not to meet information security compliance as using the words such as miss, hate, and wrong. In this way, a methodology has been proposed to prevent damage to the organization's information systems. Meanwhile, to prevent conflicts with privacy issues, it is necessary to sufficiently inform and consent internally that this analysis is merely a means for detecting insider threats and not for privacy violations. This

Table 7: Verifying malicious insider through the concept of information security compliance.

Construct	Description	Yes / No	Used Words
Spreading Knowledge of Information Security	He / She doesn't want to give knowledge of information security to the other people.	✔	hell, bad
	Others don't help to improve his / her knowledge of information security.		
	He / She doesn't like to join a seminar related to information security.		
Cooperating with Others	Cooperation is unhelpful to decrease the risk of our organization.	✔	wrong, badly
	He / She won't contribute to improve our organization through cooperation.		
Learning Information Security	He / She doesn't want to learn new things.	✔	old, hate
	Accepting new things are not helpful to him / her.		
Practicing Information Security	His / Her experience can't affect to organization positively.		
	Experiences don't motivate him / her.		
Attachment	He / She doesn't like to communicate others.	✔	miss, hate,
	He / She doesn't tend to obey the rules.	✔	damn, wrong
Commitment	He / She is careless about keeping valuable things safely.	✔	forgot
	He / She doesn't try to follow up the newest policies.	✔	weird

paper contributes to the analysis of data on social media to show that the criteria for detecting insider threats are based on the sentiment level and the ratio of negative emotions, and it can be verified based on the concept of information security compliance. Recent insider threats are not just the actions of the system suspension or information leakage, but it is very important to prevent the attack, because of the organization's reputation, customer compensation, and so on. Above all, to improve the level of information protection of the organization, it is necessary that not only the information protection person but also the management's active interest and efforts are put together. In the future, we will conduct research on methodology that can combine analysis of system behavior and individual's sentiment analysis to detect insider threat.

Data Availability

The .csv formatted data used to support the findings of this study may be released upon application to the "Sentiment140.com", who can be contacted at http://help.sentiment140.com/for-students/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2018-2015-0-00403) supervised by the IITP (Institute for Information and communications Technology Promotion).

References

[1] Statista, Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions), 2016.

[2] Statista, Number of social media users worldwide from 2010 to 2021 (in billions), 2017.

[3] A. M. Ortiz, D. Hussein, S. Park, S. N. Han, and N. Crespi, "The cluster between internet of things and social networks: review and research challenges," *IEEE Internet of Things Journal*, vol. 1, no. 3, pp. 206–215, 2014.

[4] M. Kranz, L. Roalter, and F. Michahelles, "Things that twitter: social networks and the internet of things, What can Internet Things do Citiz. Work," in *Proceedings of the 8th International Conference on Pervasive Computing (Pervasive '10)*, pp. 1–10, 2010.

[5] X. Liu, M. Zhao, S. Li, F. Zhang, and W. Trappe, "A security framework for the internet of things in the future internet architecture," *Future Internet*, vol. 9, no. 3, 2017.

[6] Cybersecurity Insiders and Crowd Research Partners, Insider threat 2018, 2017.

[7] C. Colwill, "Human factors in information security: the insider threat—who can you trust these days?" *Information Security Technical Report*, vol. 14, no. 4, pp. 186–196, 2009.

[8] N. S. Safa, R. V. Solms, and S. Furnell, "Information security policy compliance model in organizations," *Computers and Security*, vol. 56, pp. 1–13, 2016.

[9] M. B. Salem, S. Hershkop, and S. J. Stolfo, "A survey of insider attack detection research," *Advances in Information Security*, vol. 39, pp. 69-70, 2008.

[10] A. Harilal, F. Toffalini, J. Castellanos, J. Guarnizo, I. Homoliak, and M. Ochoa, "TWOS: a dataset of malicious insider threat behavior based on a gamified competition," in *Proceedings of the 2017 International Workshop on Managing Insider Security Threats*, pp. 45–56, ACM, 2017.

[11] P. A. Legg, O. Buckley, M. Goldsmith, and S. Creese, "Caught in the act of an insider attack: detection and assessment of insider

threat," in *2015 IEEE International Symposium on. IEEE*, pp. 1–6, 2015.

[12] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, "Deep learning for unsupervised insider threat detection in structured cybersecurity data streams," https://arxiv.org/abs/1710.00811.

[13] M. Bishop, H. M. Conboy, . Huong Phan et al., "Insider Threat Identification by Process Analysis," in *Proceedings of the 2014 IEEE Security and Privacy Workshops (SPW)*, pp. 251–264, San Jose, Calif, USA, May 2014.

[14] M. Kandias, V. Stavrou, N. Bozovic, and D. Gritzalis, "Proactive insider threat detection through social media: The YouTube case," in *Proceedings of the 1st ACM Workshop on Language Support for Privacy-Enhancing Technologies, PETShop 2013 - Co-located with the 20th ACM Conference on Computer and Communications Security, CCS 2013*, pp. 261–266, Germany, November 2013.

[15] V. Marivate and P. Moiloa, "Catching crime: Detection of public safety incidents using social media," in *Proceedings of the 2016 Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference, PRASA-RobMech 2016*, South Africa, December 2016.

[16] L. L. Ko, D. M. Divakaran, Y. S. Liau, and V. L. L. Thing, "Insider threat detection and its future directions," *International Journal of Security and Networks*, vol. 12, no. 3, pp. 168–187, 2017.

[17] B. A. Alahmadi, P. A. Legg, and J. R. Nurse, "Using Internet Activity Profiling for Insider-threat Detection," in *Proceedings of the 12th Special Session on Security in Information Systems*, pp. 709–720, Barcelona, Spain, April 2015.

[18] "Dataset of Sentiment140," http://help.sentiment140.com/for-students/.

[19] E. Loper and S. Bird, "NLTK: The Natural Language Toolkit," in *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pp. 1–4, 2004.

[20] D. M. Blei, "Probabilistic topic models," *Communications of the ACM*, vol. 55, no. 4, pp. 77–84, 2012.

[21] A. Dundar, J. Jin, and E. Culurciello, "Convolutional Clustering for Unsupervised Learning," pp. 1-11, 2015.

[22] S. Ben-David and S. Shalev-Shwartz, *Understanding Machine Learning: From Theory to Algorithms*, 2014.

[23] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 1398, pp. 137–142, 1998.

[24] S. Schrauwen, *Machine Learning Approaches to Sentiment Analysis Using the Dutch Netlog Corpus*, 2010.

[25] P. Bradley, U. Fayyad, and C. Reina, "Scaling EM (Expectation-Maximization) Clustering to Large Databases," *Microsoft Research*, pp. 1–25.

[26] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 47–60, 1996.

[27] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD '96)*, vol. 96, pp. 226–231, 1996.

[28] W. Park, Y. You, and K. Lee, "Twitter sentiment analysis using machine learning," *Research Briefs on Information & Communication Technology Evolution*, http://rbisyou.wixsite.com/rebicte/volume-3-2017, 2017.

[29] S. Feng, J. S. Kang, P. Kuznetsova, and Y. Choi, "Connotation lexicon: a dash of sentiment beneath the surface meaning," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 1774–1784, 2005.

[30] M. Losada and E. Heaphy, "The Role of Positivity and Connectivity in the Performance of Business Teams: A Nonlinear Dynamics Model," *American Behavioral Scientist*, vol. 47, no. 6, pp. 740–765, 2004.

[31] E. Ferrara and Z. Yang, "Measuring emotional contagion in social media," *PLoS ONE*, vol. 10, no. 11, Article ID e0142390, 2015.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 8176984, 13 pages
https://doi.org/10.1155/2018/8176984

Research Article

A Cascaded Algorithm for Image Quality Assessment and Image Denoising Based on CNN for Image Security and Authorization

Jianjun Li,[1] Jie Yu,[1] Lanlan Xu,[1] Xinying Xue,[1] Chin-Chen Chang,[2] Xiaoyang Mao,[3] and Junfeng Hu[4]

[1] *School of Computer Science and Engineering, Hangzhou Dianzi University, China*
[2] *Department of Information Engineering and Computer Science, Feng Chia University, Taiwan*
[3] *Department of Computer Science and Media Engineering, University of Yamanashi, Kofu, Japan*
[4] *Key Lab of Data Link, China Electronics Technology Group Corporation, Xi'an, China*

Correspondence should be addressed to Chin-Chen Chang; alan3c@gmail.com

Received 15 March 2018; Revised 26 June 2018; Accepted 8 July 2018; Published 17 July 2018

Academic Editor: Ilsun You

With the rapid development of Internet technology, images on the Internet are used in various aspects of people's lives. The security and authorization of images are strongly dependent on image quality. Some potential problems have also emerged, among which the quality assessment and denoising of images are particularly evident. This paper proposes a novel NR-IQA method based on the dual convolutional neural network structure, which combines saliency detection with the human visual system (HSV), used as a weighting function to reflect the important distortion caused by the local area. The model is trained using gray and color features in the HSV space. It is applied to the parameter selection of an image denoising algorithm. The experiment proves that our proposed method can accurately evaluate image quality in the process of denoising. It provides great help in parameter optimization iteration and improves the performance of the algorithm. Through experiments, we obtain both improved image quality and a reasonable result of subject assessment when the cascaded algorithm is applied in image security and authorization.

1. Introduction

With the continuous increase in the size of video surveillance networks, current large-scale video surveillance systems contain tens of thousands of cameras and millions of millions of images or pictures. However, they may be interfered and attacked [1, 2] during authorization, encryption [3], acquisition, coding, and transmission. As a result, the quality of video has deteriorated, which has affected the development of related work. In terms of artificial intelligence, for example, image content analysis, text detection, and recognition of traffic signs [4] require high image quality to ensure the accuracy and reliability of detection and recognition.

Today's systems of security and authorization are very large and there are millions of images and videos produced by the Internet every day. Under such circumstances, it is unrealistic to employ a large number of people to subjectively evaluate the quality of each monitored image and video. Therefore, how to objectively evaluate such quality to meet the needs of monitoring and authorization has become a new direction of research in the field of security and authorization.

Image quality assessment is divided into subjective and objective evaluation methods. The subjective evaluation method [5–7] mainly evaluates image quality through human viewing and is also suitable for the evaluation of video image quality. However, this method is influenced by both subjective human factors and objective factors arising from the observation environment. Therefore, the stability of the evaluation results is poor. An objective evaluation method is based on a mathematical model, which has the advantages of simple operation, fast calculation speed, and the ability to be embedded in the system. Objective evaluation methods of image quality are divided into three types: no parameter, semiparameter, and parameter. This article is based on the actual situation of the application, using the parameter-free algorithm for image distortion.

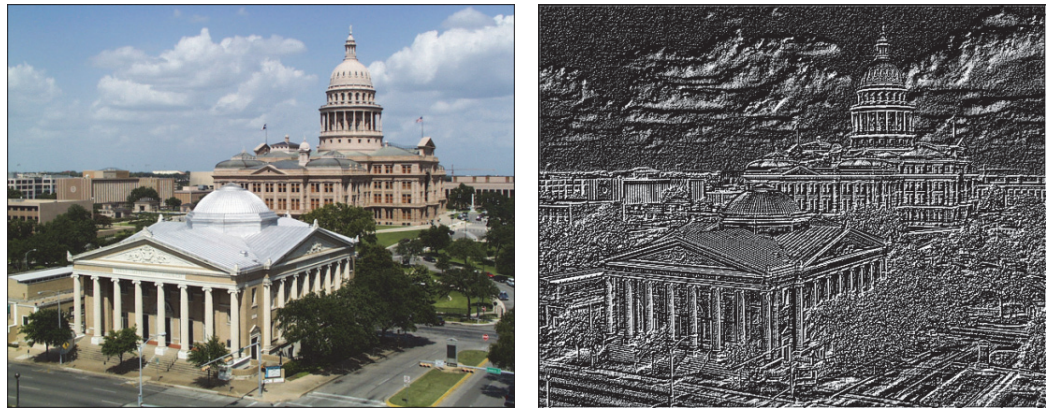


Figure 1: Example of image local normalization.

A no-reference image quality assessment (NR-IQA) method can not only evaluate image processing algorithms but also optimize the image system [8, 9]. At present, remarkable results have been achieved. Therefore, this paper proposes a parameter selection for a denoising algorithm based on the evaluation of no-reference image quality [10] and applies the no-reference image quality evaluation method to the image denoising algorithm [11–13], making it possible to adaptively select the optimal parameters. We embed it in the ROF model to experiment [14] in the search for the optimal parameters, while reducing the number of iterations of the algorithm. This achieved the best denoising effect.

2. Related Work

2.1. Image Preprocessing. Image preprocessing is a very important part of the image system. It can provide reliable data for subsequent image processing, thereby improving detection and recognition accuracy. There are many methods, such as image denoising and image enhancement, which can eliminate interference information and purposefully enhance useful information respectively. However, there are very few methods of image preprocessing in the field of image quality evaluation, because most preprocessing operations will enhance or reduce the distortion information in the image. The existing no-reference image quality evaluation methods, such as those discussed in the literature [15, 16], use local contrast normalization as a preprocessing operation. As Ruderma described in [17], image local normalization has the effect of image decorrelation. Given an image, the grayscale image is obtained and each pixel in the image is subtracted from its local mean and divided by its local variance, as shown in Figure 1. Assuming that $I(i, j)$ is the pixel value at position (i, j) in the image, the local comparison operation is normalized:

$$\hat{\mathrm{I}}(i,\mathrm{j}) = \frac{I(\mathrm{i},\mathrm{j}) - \mu(i,\mathrm{j})}{\sigma(\mathrm{i},\mathrm{j}) + \mathrm{C}} \tag{1}$$

where $i \in 1, 2, \ldots, M$, $j \in 1, 2, \ldots, N$, M and N are the height and width of the image, and C is a positive constant to avoid having the denominator be zero.

$$\mu(\mathrm{i},\mathrm{j}) = \sum_{k=-K}^{K}\sum_{l=-L}^{L} \omega_{k,l} I_{k,l}(i,j) \tag{2}$$

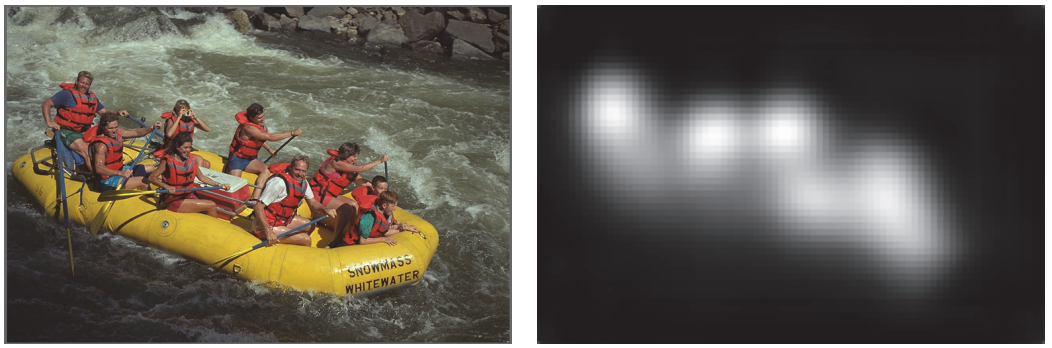
$$\sigma(\mathrm{i},\mathrm{j}) = \sqrt{\sum_{k=-K}^{K}\sum_{l=-L}^{L} \omega_{k,l}\left(I_{k,l}(i,j) - \mu(i,j)\right)^2} \tag{3}$$

where $w = \{w_{k,l} \mid k = -K, \ldots, K,\ l = -L, \ldots, L\}$ is a 2-dimensional symmetric Gaussian weighting function and K and L are the normalized window sizes. The literature [15] shows that when calculating the local mean and variance, the performance of the algorithm is relatively stable with the change of window scale. However, when the window scale becomes very large, the performance of the algorithm will begin to decrease because the calculated mean and variance are no longer local information.

2.2. Visual Saliency. In the method proposed by Le Kang, the average value of all local quality scores for the test image is taken as the quality value for the entire image. However, such methods do not take into account human visual perception of the image, because the content of the local image block and its position in the entire image will affect the viewer's visual perception. For example, people are less sensitive to distortion in flat areas of an image (such as blue sky) than in complex textured areas (such as edges). Each block in the image has a different effect on human-perceived image quality. We therefore use the saliency of image blocks in the following study as a weight to represent the degree of human visual perception. Figure 2 shows how to predict a saliency map.

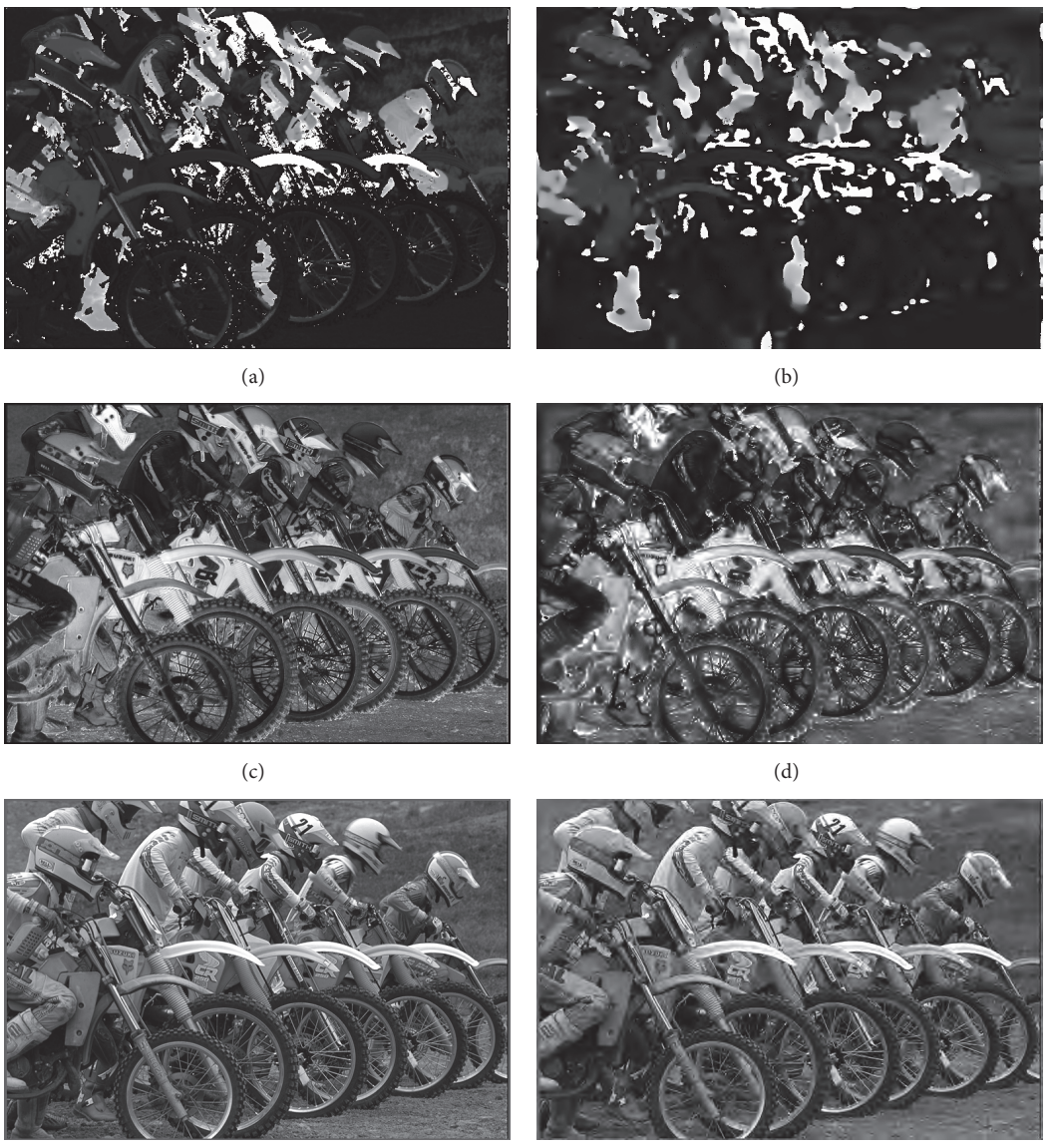
3. Method

3.1. Input. Previous NR-IQA methods based on deep learning, such as [16, 18], consider only the information of grayscale images and ignore the distortion information contained in the color components of the image. Yet, as can be seen in Figure 3, distortion has the most significant effect on the hue component. Therefore, we use HSV as the color space and use the image hue component and grayscale information



(a) (b)

Figure 2: Schematic detection results: (a) original image and (b) saliency map.



(a) (b)

(c) (d)

(e) (f)

Figure 3: The representation of an image in the HSV color space. The first column contains the three components of the reference image in the HSV color space. The second column is the three components of the JPEG2000 distorted image in the HSV color space. (a) and (b) represent a hue component; (c) and (d) represent a saturation component; (e) and (f) represent a value component.

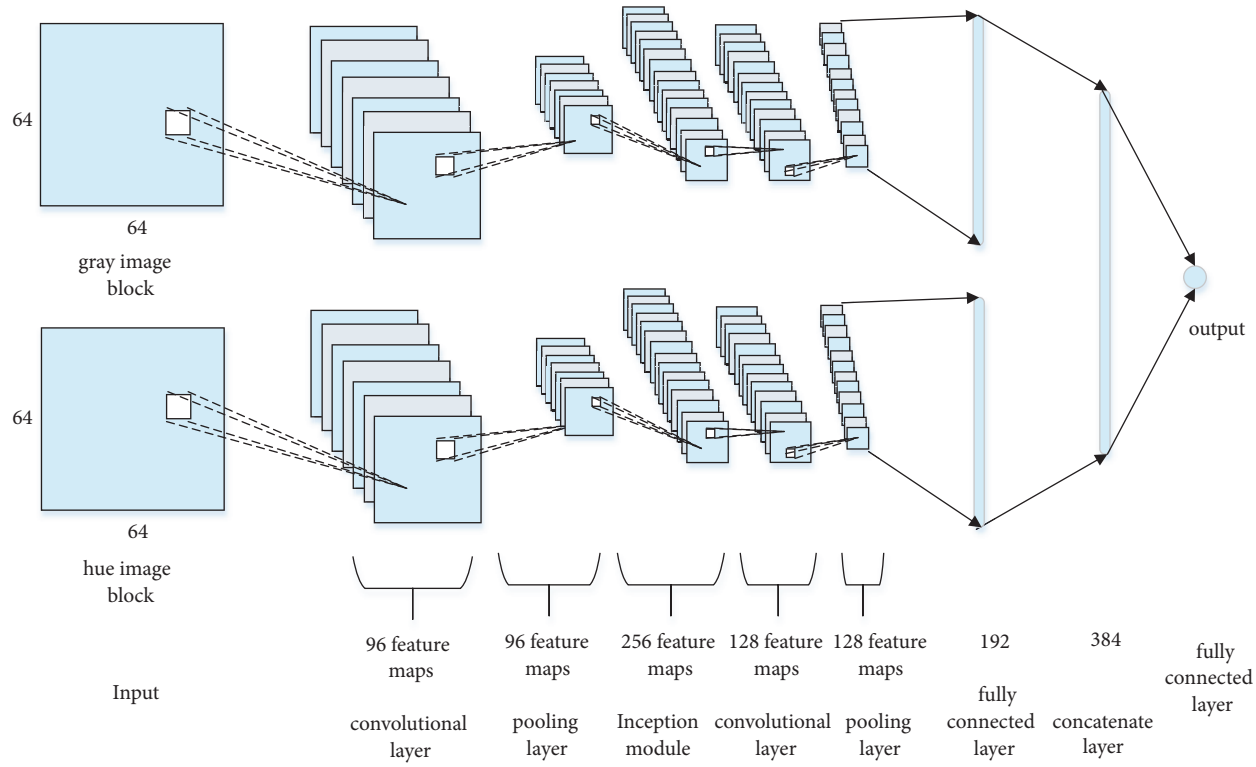


Figure 4: CNN model architecture.

as the input for the CNN model to extract features related to image distortion.

3.2. CNN. In this section, we will introduce the proposed CNN model in detail. Figure 4 shows the network structure of the model. It consists of two parts. Apart from the different inputs, the other structures, including the number and size of the convolution kernels, are the same. One of the inputs is the gray scale image block and the other input is the hue image block. The size of the block is 64×64. For the network, the first layer is a convolutional layer with 96 convolutional kernels with size of 5×5 and stride of 2 pixels for each. It produces 96 feature maps with size of 30×30 for each. A max pooling layer follows the convolutional layer to reduce each feature map to size of 15×15. After the pooling layer, there is an inception module that produces 256 feature maps with size of 15×15 for each. The next layer is a convolutional layer with 128 convolutional kernels with size of 3×3 and a padding of 1 pixels. It produces 128 feature maps with size of 15×15 for each. Following the convolutional layer is a max pooling layer that samples feature maps to a size of 7×7. The sixth layer is a fully connected layer of 192 nodes. The next layer is a concatenate layer that connects two feature vectors with dimension of 192 for each layer. It has two components. The last layer is also a fully connected layer that gives the quality score of the image block. Except for the last fully connected layer, each layer is followed by a ReLUs (Rectified Linear Units) activation layer.

In addition, to prevent overfitting when training the CNN model, we apply dropout regularization with a ratio of 0.5 before the last fully connected layer. Because image quality assessment is a quantitative problem, the CNN needs continuous variable prediction. Thus, the Euclidean distance loss function is chosen as the learning loss function:

$$\mathrm{E} = \frac{1}{2N} \sum_{n=1}^{N} \left\| \widehat{y}_n - y_n \right\|_2^2 \tag{4}$$

where $\widehat{y}_n$ is the image block quality score predicted by CNN, y_n is the image block quality score, and N is the number of blocks.

3.3. Image Quality Assessment. Through the proposed CNN model, the local image quality evaluation can be obtained. An RGB image is given, converted into a grayscale image and a hue image in the HSV color space, and then the grayscale and hue images are sampled separately. The sampled image block is then subjected to a preprocessing operation, i.e., the local contrast normalization described in (3). Figure 5 is a preprocessed gray image. Similarly, the hue image is subjected to the same sampling operation and preprocessing to obtain a corresponding hue image block. Finally, the prepared gray image block and the hue image block are inputted in pairs into the CNN model, and a series of convolution and pooling processes are performed to output the image block quality value, i.e., the local image quality evaluation.

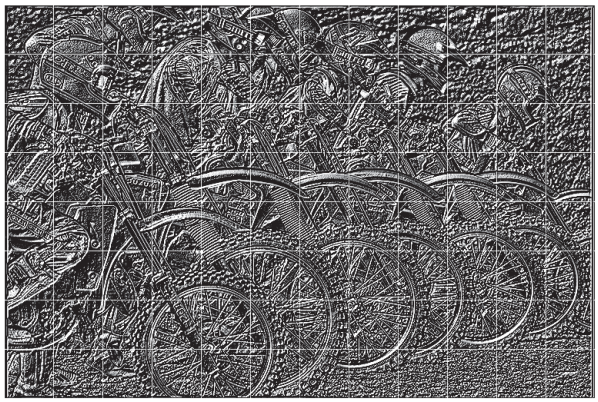


Figure 5: Preprocessed grayscale image blocks.

For a test image G, we obtain the corresponding saliency map S through the saliency detection model. In order to obtain the image block weight corresponding to the CNN model, we adopt a nonoverlapping sampling strategy for the saliency map S. The weight of the image block is expressed as

$$c_k = \sum_{m=-H}^{H} \sum_{n=-H}^{H} p_{s,k}(i,j) \tag{5}$$

$$w_k = \frac{c_k}{\sum_{l=1}^{l} c_k} \tag{6}$$

For test image $G = [P_{g,1}, P_{g,2}, \ldots, P_{g,l}]$ and saliency map $S = [P_{s,1}, P_{s,2}, \ldots, P_{s,l}]$, l is the number of blocks. There is a one-to-one correspondence between P_g and P_s. $p_{s,k}(i,j)$ is the pixel value of the position (i,j) in the significant image block $P_{s,k}$, and c_k is the sum of the coefficients of the image block $P_{s,k}$. Finally, we calculate the global quality score of test image G:

$$Q = \sum_{k=1}^{l} w_k q_k \tag{7}$$

The algorithm structure is shown in Figure 6.

3.4. Image Denoising. Nowadays, many algorithms [19–21] have achieved very good denoising effects. For example, Rudin et al. [21] proposed a classic total variation (TV) denoising algorithm in 1992, namely, the ROF model. However, an iterative denoising algorithm is always computationally intensive since it requires multiple rounds of image quality assessment to find the best parameters. To solve this problem, we propose a parameter selection framework which reduces the number of iterations of the algorithm while finding the optimal parameters, in order to save execution time.

3.4.1. No-Reference Image Quality Assessment. The assessment method for image quality will be applied to an image processing algorithm to achieve optimal parameter selection. Based on the success of the CNN model for image quality assessment in Section 3.3, we propose a simple CNN model for evaluating the quality of denoised images, as shown in Figure 7. First, we perform a local normalization for a gray image and then sample nonoverlapping image blocks where the size is 64×64 pixels from normalized image. Our network consists of ten layers: 64×64-$32 \times 30 \times 30$-$32 \times 15 \times 15$-$96 \times 15 \times 15$-$96 \times 7 \times 7$-$128 \times 7 \times 7$-$128 \times 3 \times 3$-800-800-1. We apply a dropout regularization with a ratio of 0.5 after the second fully connected layers. Finally, we obtain average scores of all of the image blocks to represent the entire image quality score.

3.4.2. Image Denoising Algorithm. The ROF model is one of the most effective methods for image denoising. It aims to model the problem of image denoising as a minimization of the energy function to make the image smooth while the edge can be maintained well. Generally, the total variation value of the noisy image is larger than the clear image. The effect of noise removal can be achieved by solving the minimization function of the total variance. Therefore, the ROF model for image denoising is as follows:

$$\operatorname{argmin}\left\{\frac{\mu}{2}\|u-f\|_2^2 + \|u\|_{TV}\right\} \tag{8}$$

where f is the noise image and u is the denoising image. The first term of the above formula is the fidelity term, so that the denoised image u keeps as much of the information in the noise image f as possible. The second item is the TV regular term, which allows the model to effectively preserve edges while denoising. $\mu > 0$ is a regularization parameter. The larger μ is, the smoother the denoising image will be, resulting in blurred image. The smaller μ is, the worse the denoising effect is. Therefore, choosing a suitable μ has a great effect on the results of the denoising. There are two cases of total variance. In this section, we only consider the isotropic ROF denoising problem:

$$\min\left\{\left\|\sqrt{(\nabla_x u)^2 + (\nabla_y u)^2}\right\|_1 + \frac{\mu}{2}\|u-f\|_2^2\right\} \tag{9}$$

Split Bregman method [22] is used to solve formula (9); assuming $d_x \approx \nabla_x u$ and $d_y \approx \nabla_y u$, the image denoising problem is as follows:

$$\begin{aligned}\min &\left\|(d_x, d_y)\right\|_2 + \frac{\mu}{2}\|u-f\|_2^2 + \frac{\lambda}{2}\|d_x - \nabla_x u - b_x\|_2^2 \\ &+ \frac{\lambda}{2}\|d_y - \nabla_y u - b_y\|_2^2\end{aligned} \tag{10}$$

Using Bregman iterative algorithm to solve (10), the isotropic ROF algorithm is as shown in Algorithm 1.

G denotes solving the linear equations with Gauss-Seidel iteration. We focus on the influence of parameter selection on the performance of the total variation denoising algorithm.

3.4.3. The Framework of Parameter Selection. In order to study the effect of parameter selection based on image quality on denoising results, we denoise the Park image [23] with 25 different values of parameter μ. The values are uniformly

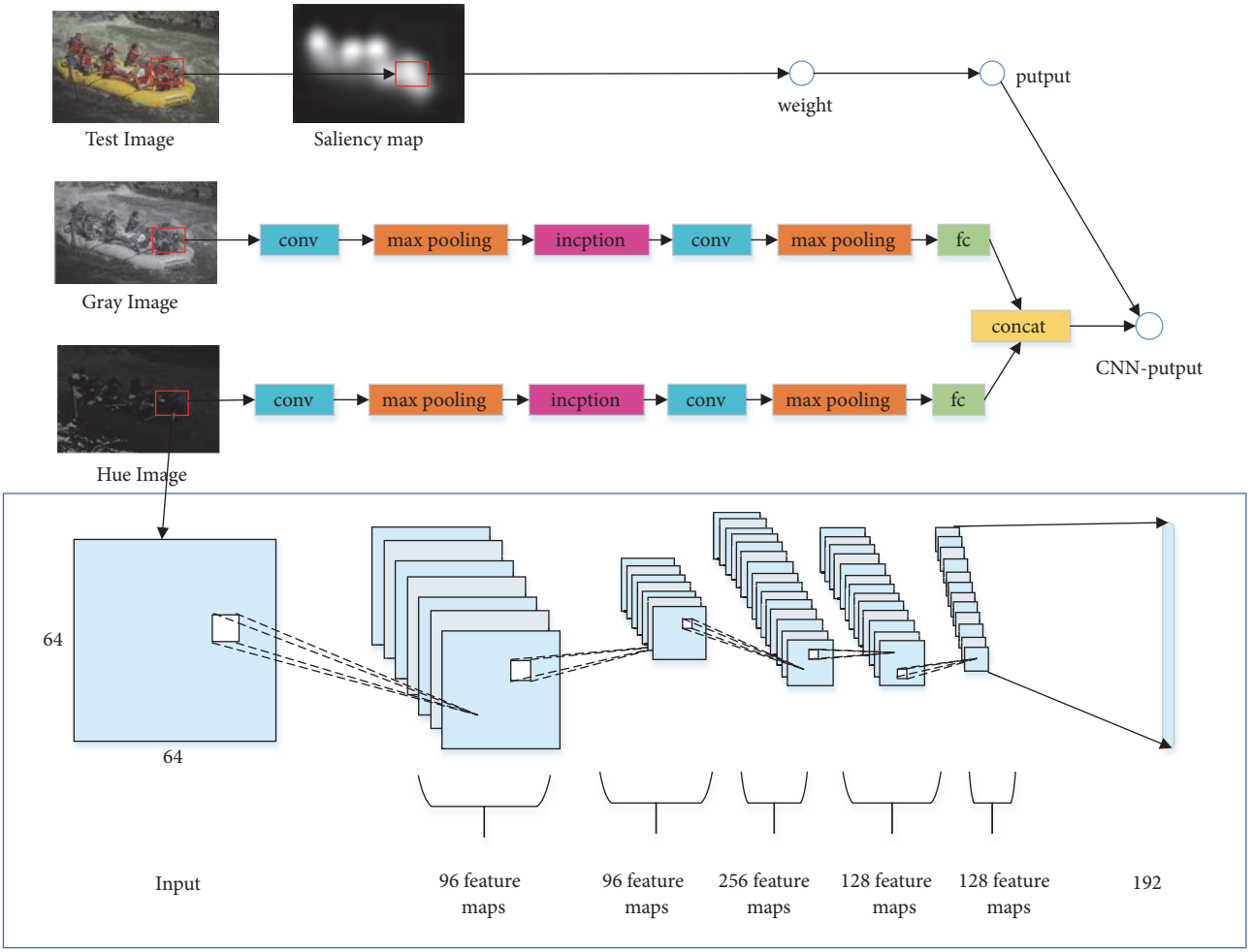


Figure 6: Algorithm architecture.

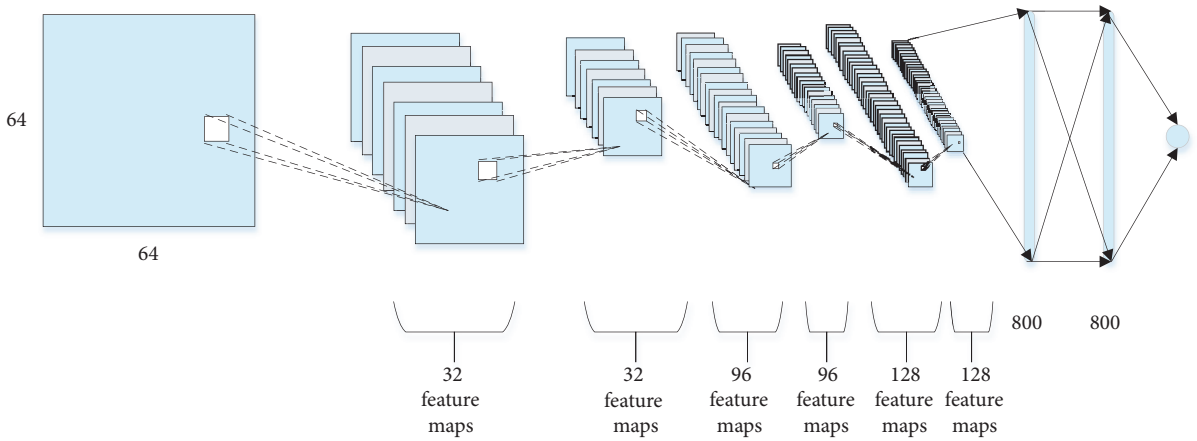


Figure 7: CNN model architecture.

sampled from 1 to 49. Figure 8(a) shows the variation of the denoised image quality in the iterative process for different parameter values. Figure 8(b) is the result of the denoised image obtained using the SSIM and CNN Q methods to evaluate different parameters. Their predicted image quality trends are the same. It is obvious that the best denoising effect is obtained when parameter μ is 13.

However, image quality assessment is performed after the completion of the whole processing of iterative convergence when the range of parameter is large. Therefore, choosing an

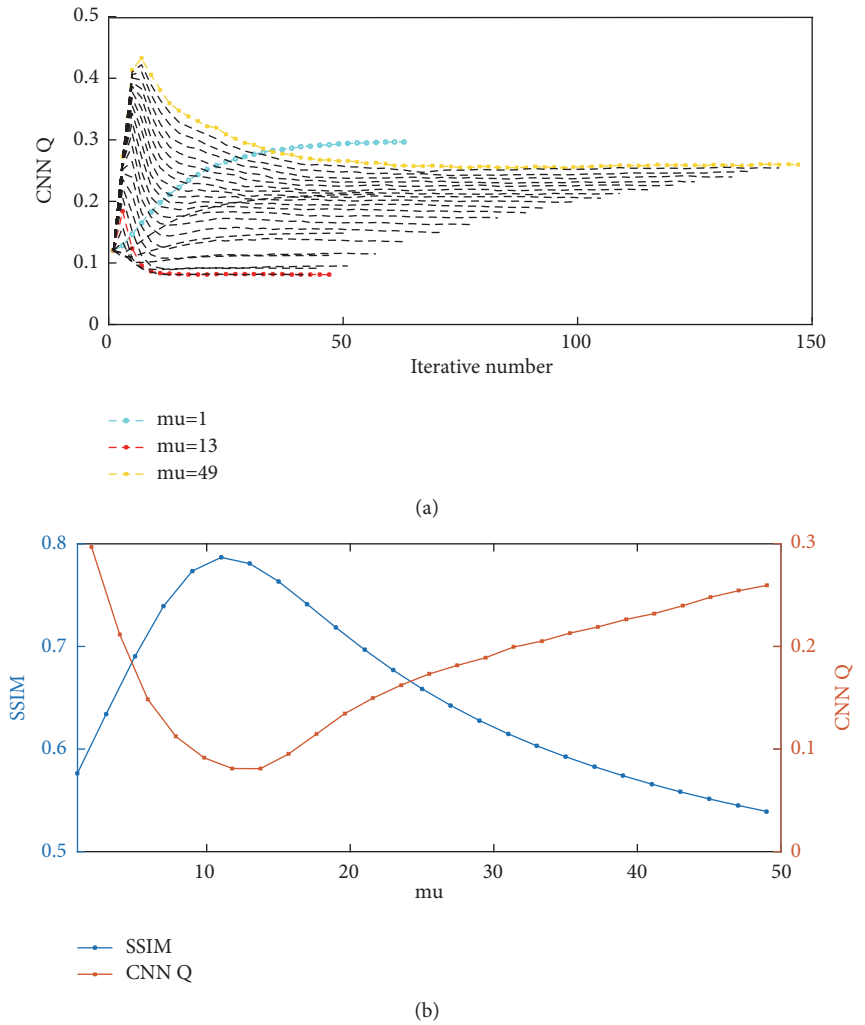


Figure 8: (a) Iterative process of different parameter values; (b) using SSIM and CNN Q to evaluate the denoising image quality of different parameter values.

Initialize: $u^0 = f, d_x^0 = d_y^0 = b_x^0 = b_y^0 = 0$
1: **while** $\|u^k - u^{k-1}\|_2 > tol$ **do**
2: $u^{k+1} = G(u^k)$
3: $d_x^{k=1} = \dfrac{\max(s^k - 1/\lambda, 0)(\nabla_x u^k + b_x^k)}{s^k}$
4: $d_y^{k=1} = \dfrac{\max(s^k - 1/\lambda, 0)(\nabla_y u^k + b_y^k)}{s^k}$
5: $b_x^{k+1} = d_x^k + (\nabla_x u^{k+1} - d_x^{k+1})$
6: $b_y^{k+1} = d_y^k + (\nabla_y u^{k+1} - d_y^{k+1})$
7: **end while**

Algorithm 1

optimum parameter in a certain range requires a very large number of iterations.

Therefore, we use the changes in iteratively generated denoising image quality as a basis for determining if

if $\max(D_m, D_{m-1}, \ldots, D_{m-l}) < T_q$ **then**
Converge
else
Continue iterating
end if

Algorithm 2

the iterative algorithm converges. Assuming that Dm = Q(m+1)−Q(m), Q(m) represents the quality score of the denoised image produced by the m-th iteration, and the algorithm convergence is presented in Algorithm 2.

"l" denotes the iterative length of the change of image quality that needs to be taken into account to ensure that the algorithm converges and produces a permissible change in image quality. The larger the "l" value is, the smaller the "Tq" value becomes, indicating that the convergence condition of the algorithm is stricter.

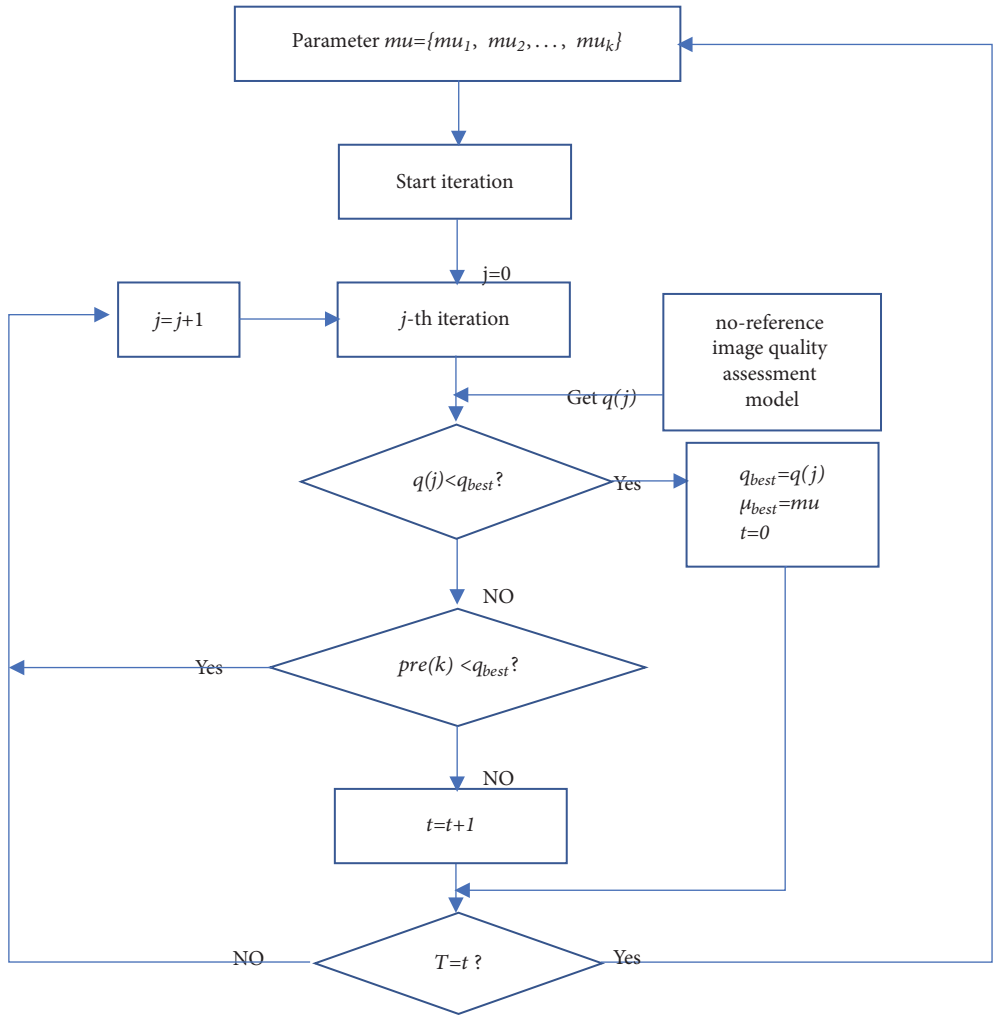


Figure 9: Framework of parameter selection.

Our improvement to the ROF algorithm is as follows. The algorithm can determine whether the parameter value is the optimal parameter we need before convergence, as shown in Figure 9.

Suppose $pre(k) = q(j) + dev(j) * l_{pre}$; $pre(k)$ is the predicted quality value of the parameter k. $q(j)$ represents the quality of the denoised image from the jth iteration. $dev(j)$ is the numerical derivative of the jth iteration. l_{pre} is the predicted iteration length.

Suppose q_{best} is the quality score of denoised image. And t is the count and T is the threshold.

At the end of each iteration of the ROF algorithm, we use the above-mentioned evaluation algorithm to evaluate the quality of the denoising image $q(j)$. In order to obtain an optimal-quality image, $q(j)$ is compared with the quality of the best image q_{best} obtained previously. The iterative process terminates if the quality of the denoised image resulting from T consecutive iterations is worse than q_{best}. At this time we believe that the current parameter has achieved the best denoising effect. More iterations of the current parameter are unnecessary. However, there may be an abrupt change in the quality of the image produced by subsequent iterations, so the judgment condition $pre(k) < q_{best}$ is increased to prevent misjudgment. Next, we need to iteratively evaluate other parameters in the same way.

If we want to further improve the performance of the parameter selection framework, we can simplify the complexity of the image evaluation network model, as described in Section 3.4.1, to speed up the evaluation time at the expense of minor loss of precision. Threshold T relates to the stringency of the convergence conditions. Finding a reasonable *T-value* for the speed and accuracy is helpful for the algorithm. In addition, we can also reduce the number of candidate parameters by reducing the fineness of the range division of the parameter μ within an acceptable range.

4. Results and Analysis

4.1. Image Quality Assessment Experiment. We obtain the quality evaluation of the image by weighting the quality score of the sampled image block. Therefore, the sampling strategy can affect the overall algorithm performance. In this section,

Table 1: LCC and SROCC with different patch size.

Size	80	72	64	56	48
LCC	0.967	0.966	0.964	0.958	0.953
SROCC	0.975	0.964	0.962	0.957	0.951

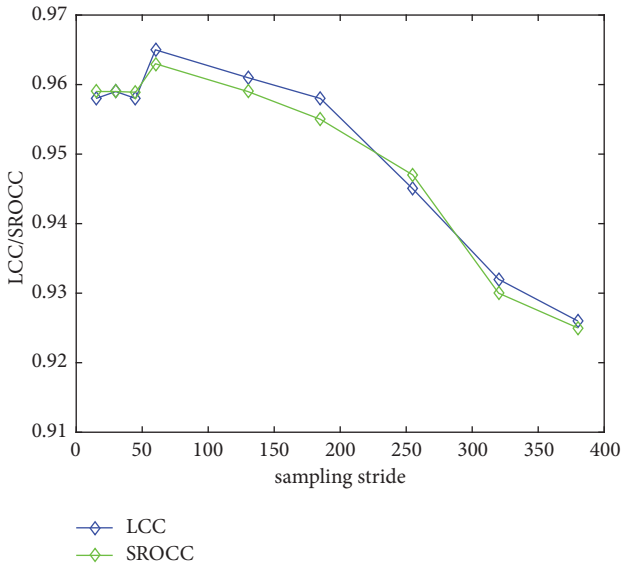


Figure 10: LCC and SROCC with different sampling stride.

we study the effect of two parameters on the performance of the algorithm: the block size of the sampled image and the number of samples per image.

(1) Block size: we assume the sampled image is nonoverlapped. In order to study the effect of image block size on the performance of our algorithm, we use fixed sampling strides and different image block sizes. The results are shown in Table 1. We find that the performance increases slightly with the increasement of patch size. Moreover, the performance of the algorithm will begin to decrease if the size is very large because the local information of the image has been lost. Here, we choose the size of 64x64.

(2) Sampling stride: we study the effect of the number of image blocks sampled on each image. We use a fixed block size of 64x64 and various strides. Figure 10 shows that different sampling strides lead to changes in the performance of the algorithm. We have found that the number of image blocks seriously affects the performance of the algorithm. In general, a larger stride leads to lower performance because less image information is used to evaluate image quality.

In order to test the performance of our proposed algorithm, we conducted experiments on the LIVE database [24]. In order to ensure the robustness of our proposed method, we performed 100 training and testing experiments, independently of the image content and the specific training test set. For each experiment, we randomly selected distorted images corresponding to 60% of the reference images in the LIVE database as the training set, 20% distorted image as the verification set, and the remaining images as the test set. Therefore, all the results shown in this chapter are averages of the results of these 100 experiments.

Tables 2 and 3 are the results of comparing our algorithm to the existing representative image quality assessment methods, including no-reference image quality evaluation methods and reference image quality evaluation methods. No-reference methods include DIIVINE [24], BLIINDS-II [25], BRISQUE, CORNIA [26], and CNN.

It can be seen that the method we propose has a high degree of consistency with human subjective evaluation. Compared to other no-reference image quality assessment methods, our method has the best performance on the three distortion types, JP2K, WN, and FF.

In order to test the generalization ability of our proposed algorithm, we conducted an experiment on the TID2008 database. We train our model on the entire LIVE database and then test model on the TID2008 database. In Table 4, the LCC indicator shows that our algorithm's performance is slightly better than those of other algorithms, similarly to the CNN method in [16]. The SROCC value of our algorithm is very high, which is obviously better than the performance of the other algorithms. In general, the independence of our proposed algorithm is strong.

4.2. Algorithm Evaluation Experiment. First, we evaluate the performance of our proposed evaluation algorithm. We train our CNN model on the LIVE database and test the performance of model on the TID2008 database. Our training data is WN and blur types of distortion images from the LIVE database. By default, all results reported are averaged from 100 train-test iterations. In each iteration, we randomly select 80% distorted images as the training set and the remaining 20% as the validation set. The result is shown as Table 5.

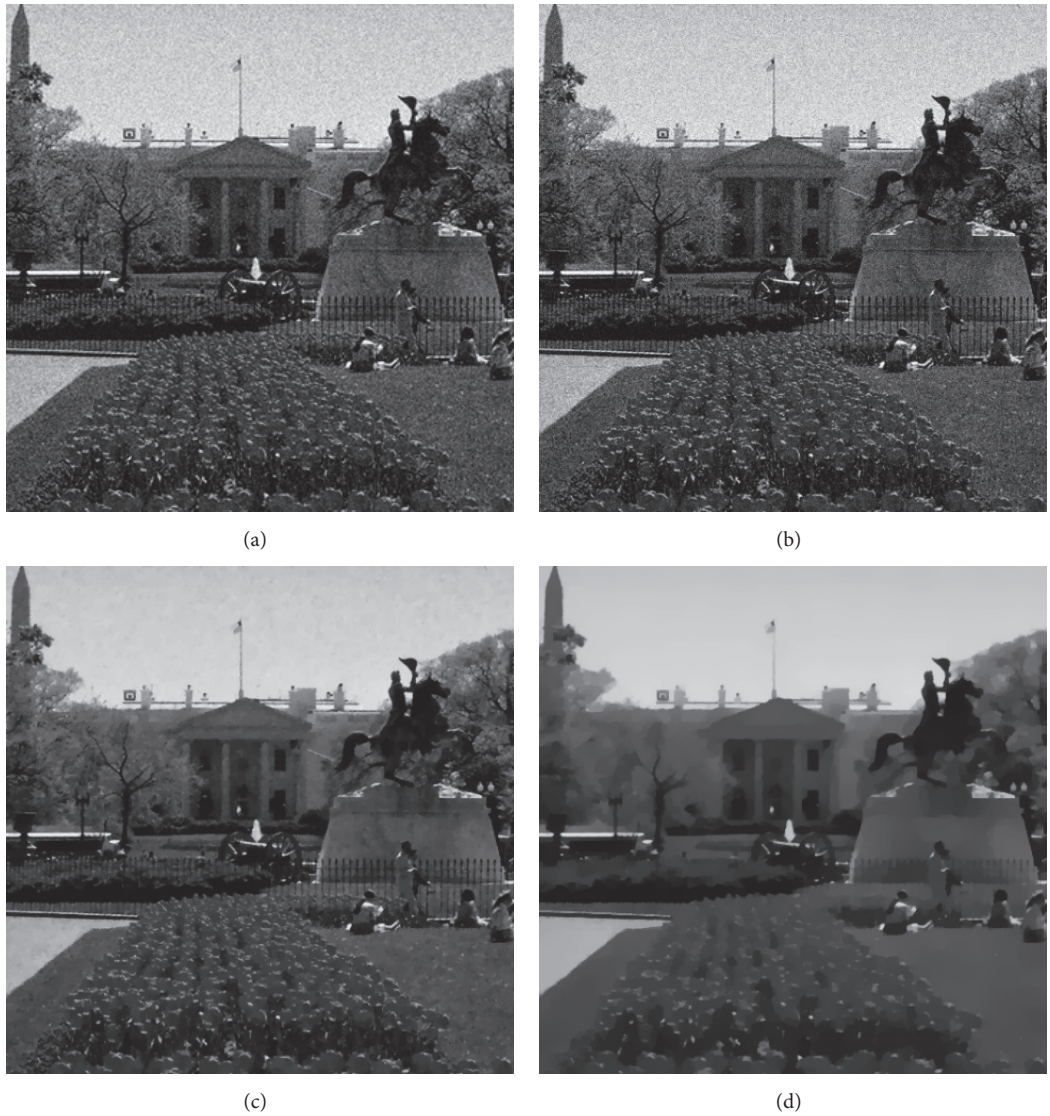
We apply the parameter selection framework to the ROF denoising model. First, we use different parameters to denoise the two images and observe the denoising results. Figure 11 presents the denoising results of the Park image at three different parameter values. For Park images, the image denoising effect is better than that of other values when the μ parameter value is 13.

Next, we study the influence of the parameter selection framework on the iterative denoising algorithm. In our experiment, we set $T_q = 0.001$, $l = 5$, $l_{pre} = 5$, and $T = 10$. We evaluate the quality of the denoised image produced by each iteration to predict the quality of the denoised image during subsequent iterations. Figure 11 is the result of comparative experiments with a parameter selection framework and with no parameter selection framework for denoising a Park graph. As shown in Figure 12, with a denoising algorithm that uses a parameter selection framework, the number of iterations per parameter is significantly reduced. Moreover, the optimal parameter values selected by the two experiments are the same, which indicates that our proposed parameter selection framework not only greatly reduces the number of iterations of the algorithm, but also accurately selects the optimal parameter value.

In order to further prove this conclusion, we calculated the data generated by the denoising experiments on multiple

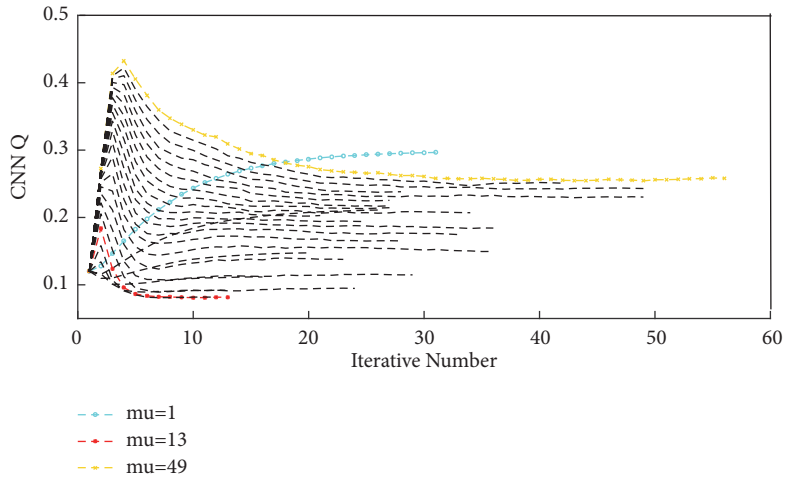
Table 2: Median SROCC across 100 train-test iterations on the LIVE database.

SROCC	JP2K	JPEG	WN	BLUR	FF	ALL
PSNR	0.870	0.885	0.942	0.763	0.874	0.866
SSIM	0.939	0.946	0.964	0.907	0.941	0.913
FSIM	0.970	0.981	0.967	0.972	0.949	0.964
DIIVINE	0.913	0.910	0.984	0.921	0.863	0.916
BLIINDS-II	0.929	0.942	0.969	0.923	0.889	0.931
BRISQUE	0.914	0.965	0.979	0.951	0.877	0.940
CORAIN	0.943	0.955	0.976	**0.969**	0.906	0.942
CNN	0.952	**0.977**	0.978	0.962	0.908	0.956
Proposed	**0.967**	0.968	**0.986**	0.962	**0.931**	**0.962**

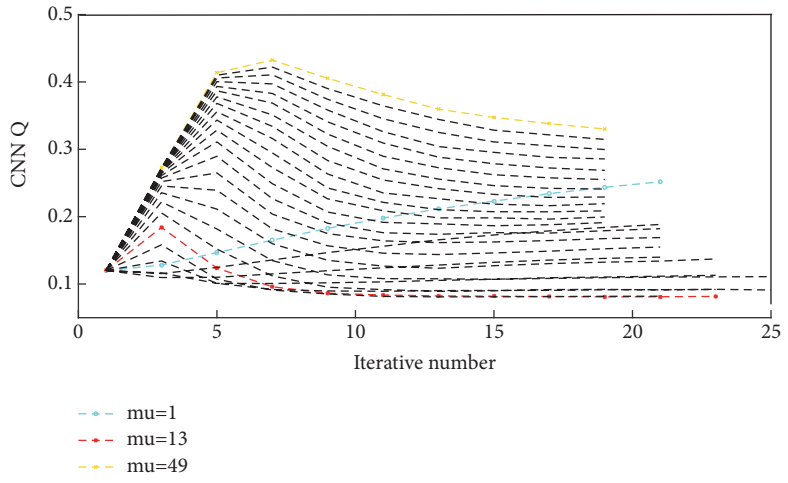


(a) (b) (c) (d)

Figure 11: (a) Park image of noise image; (b) denoising result with μ parameter value equal to 1; denoising result with μ parameter equal to 13; (d) denoising result with μ parameter equal to 49.



(a)



(b)

Figure 12: Park diagram, (a) ROF denoising process without parameter selection framework; (b) ROF denoising process with parameter selection framework.

Table 3: Median LCC across 100 train-test iterations on the LIVE database.

LCC	JP2K	JPEG	WN	BLUR	FF	ALL
PSNR	0.873	0.876	0.926	0.779	0.870	0.856
SSIM	0.921	0.955	0.982	0.893	0.939	0.906
FSIM	0.910	0.985	0.976	0.978	0.912	0.960
DIIVINE	0.922	0.921	0.988	0.923	0.888	0.917
BLIINDS-II	0.935	0.968	0.980	0.938	0.896	0.930
BRISQUE	0.922	0.973	0.985	0.951	0.903	0.942
CORAIN	0.951	0.965	0.987	**0.968**	0.917	0.935
CNN	0.953	**0.981**	0.984	0.953	0.933	0.953
Proposed	**0.974**	0.975	**0.991**	0.963	**0.953**	**0.964**

Table 4: Experimental LCC and SROCC indicators on the TID2008 database.

	BRISQUE	CORNIA	CNN	Proposed
LCC	0.882	0.892	0.920	**0.922**
SROCC	0.892	0.880	0.903	**0.921**

Table 5: Results on LIVE validation sets and TID2008 test sets.

	LCC	SROCC
LIVE validation set	0.966	0.957
TID2008 testing set	0.9161	0.9013

Table 6: Performance of our parameter selection framework.

	Parameter selection framework			without parameter selection framework			Reduced iteration
	μ best of A	Iterations of A	CPU time of A(s)	μ best of B	Iterations of B	CPU time of B(s)	
Park	13	259	40.125	13	658	79.528	60.7%
Fisher	11	238	37.524	11	932	122.45	74.5%
Pepper	21	245	37.283	21	709	88.194	65.4%
Man	11	217	30.478	11	707	86.526	69.3%
Cameraman	13	274	45.589	13	1063	132.36	74.2%
Barbara	13	235	34.235	13	817	106.75	71.2%

images. The experimental results are shown in Table 6. Obviously, they are consistent with the above findings. According to experiments, the running time of the improved ROF algorithm is much lower than that of the original algorithm. In other words, the computational cost of the algorithm mainly comes from the iteration; i.e., the computational cost is proportional to the number of iterations. We improve the algorithm performance by reducing the number of iterations while ensuring the optimal parameters are obtained

5. Conclusions

We propose a cascaded algorithm for no-reference image quality assessment and image denoising based on CNN and visual saliency, to be applied in image security and authorization. This algorithm uses a strategy of significantly weighting image blocks to obtain a quality assessment more in line with the human visual system. This method has achieved very good performance on the LIVE database. The evaluation method is embedded into an image denoising algorithm, such as the total variation denoising algorithm. We propose a parameter selection framework that can judge the advantages and disadvantages of the parameters according to the quality change of the denoising image generated during the iterative process. And it is verified that the iteration number of denoising process is significantly reduced by parameter selection. Meanwhile, we can accurately find the best parameters from a bunch of parameter candidates.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Z. Huang, S. Liu, X. Mao, K. Chen, and J. Li, "Insight of the protection for data security under selective opening attacks," *Information Sciences*, vol. 412-413, pp. 223–241, 2017.

[2] L. Sun, Z. Li, Q. Yan, W. Srisa-an, and Y. Pan, "SigPID: significant permission identification for android malware detection," in *Proceedings of the 2016 11th International Conference on Malicious and Unwanted Software (MALWARE)*, pp. 1–8, Fajardo, PR, USA, October 2016.

[3] Y. Zhang, X. Chen, J. Li, D. S. Wong, H. Li, and I. You, "Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing," *Information Sciences*, vol. 379, pp. 42–61, 2017.

[4] C. Yan, H. Xie, S. Liu, J. Yin, Y. Zhang, and Q. Dai, "Effective Uyghur Language Text Detection in Complex Background Images for Traffic Prompt Identification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 220–229, 2018.

[5] Q. Xu, T. Jiang, Y. Yao, Q. Huang, B. Yan, and W. Lin, "Random partial paired comparison for subjective video quality assessment via hodgerank," in *Proceedings of the 19th ACM International Conference on Multimedia ACM Multimedia 2011, MM'11*, pp. 393–402, USA, December 2011.

[6] L. L. Thurstone, "A law of comparative judgment," *Psychological Review*, vol. 34, no. 4, pp. 273–286, 1927.

[7] K. Chen, C. Wu, Y. Chang, and C. Lei, "A crowdsourceable QoE evaluation framework for multimedia content," in *Proceedings of the the seventeen ACM international conference*, p. 491, Beijing, China, October 2009.

[8] W.-S. Lai, J.-B. Huang, Z. Hu, N. Ahuja, and M.-H. Yang, "Comparative study for single image blind deblurring," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 1701–1709, USA, July 2016.

[9] H. Liang and D. S. Weller, "Comparison-based image quality assessment for selecting image restoration parameters," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5118–5130, 2016.

[10] X. Chen, S. Chen, and Y. Wu, "Coverless information hiding method based on the Chinese character encoding," *Journal of Internet Technology*, vol. 18, no. 2, pp. 91–98, 2017.

[11] J. L. Herraiz, S. Gabarda, and G. Cristobal, "Automatic parameter selection in PET image reconstruction based on no-reference image quality assessment," in *Proceedings of the 2012 IEEE Nuclear Science Symposium and Medical Imaging Conference Record, NSS/MIC 2012*, pp. 3371–3374, USA, November 2012.

[12] X. Zhu and P. Milanfar, "Automatic parameter selection for denoising algorithms using a no-reference measure of image content," *IEEE Transactions on Image Processing*, vol. 19, no. 12, pp. 3116–3132, 2010.

[13] H. Liang and D. S. Weller, "Regularization parameter trimming for iterative image reconstruction," in *Proceedings of the 49th Asilomar Conference on Signals, Systems and Computers, ACSSC 2015*, pp. 755–759, USA, November 2015.

[14] J. Li, L. Xu, H. Li, C. Chang, and F. Sun, "Parameter Selection for Denoising Algorithms Using NR-IQA with CNN," in *MultiMedia Modeling*, vol. 10704 of *Lecture Notes in Computer Science*, pp. 381–392, Springer International Publishing, Cham, 2018.

[15] A. Mittal, A. K. Moorthy, and A. . Bovik, "No-reference image quality assessment in the spatial domain," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4695–4708, 2012.

[16] L. Kang, P. Ye, Y. Li, and D. Doermann, "Convolutional neural networks for no-reference image quality assessment," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, pp. 1733–1740, USA, June 2014.

[17] D. L. Ruderman, "The statistics of natural images," *Networks*, vol. 5, pp. 517–548, 1994.

[18] S. Bosse, D. Maniry, T. Wiegand, and W. Samek, "A deep neural network for image quality assessment," in *Proceedings of the 23rd IEEE International Conference on Image Processing, ICIP 2016*, pp. 3773–3777, USA, September 2016.

[19] A. Buades, B. Coll, and J. M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, vol. 4, no. 2, pp. 490–530, 2005.

[20] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[21] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1–4, pp. 259–268, 1992.

[22] T. Goldstein and S. Osher, "The Split Bregman Method for L1-Regularized Problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 323–343, 2009.

[23] E. C. Larson and D. M. Chandler, "Most apparent distortion: full-reference image quality assessment and the role of strategy," *Journal of Electronic Imaging*, vol. 19, no. 1, Article ID 011006, 2010.

[24] H. Sheikh, A. Bovik, and L. Cormack, "No-reference quality assessment using natural scene statistics: JPEG2000," *IEEE Transactions on Image Processing*, vol. 14, no. 11, pp. 1918–1927, 2005.

[25] M. A. Saad, A. C. Bovik, and C. Charrier, "Blind image quality assessment: a natural scene statistics approach in the DCT domain," *IEEE Transactions on Image Processing*, vol. 21, no. 8, pp. 3339–3352, 2012.

[26] P. Ye, J. Kumar, L. Kang, and D. Doermann, "Unsupervised feature learning framework for no-reference image quality assessment," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2012*, pp. 1098–1105, USA, June 2012.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 8510256, 12 pages
https://doi.org/10.1155/2018/8510256

Research Article

Design and Analysis of Push Notification-Based Malware on Android

Sangwon Hyun,[1] **Junsung Cho**,[2] **Geumhwan Cho**,[2] **and Hyoungshick Kim**[2]

[1]*Department of Computer Engineering, Chosun University, Gwangju, Republic of Korea*
[2]*Department of Software, Sungkyunkwan University, Suwon, Republic of Korea*

Correspondence should be addressed to Hyoungshick Kim; hyoung@skku.edu

Received 14 March 2018; Accepted 6 June 2018; Published 9 July 2018

Academic Editor: Francesco Palmieri

Establishing secret command and control (C&C) channels from attackers is important in malware design. This paper presents design and analysis of malware architecture exploiting push notification services as C&C channels. The key feature of the push notification-based malware design is *remote triggering*, which allows attackers to trigger and execute their malware by push notifications. The use of push notification services as covert channels makes it difficult to distinguish this type of malware from other normal applications also using the same services. We implemented a backdoor prototype on Android devices as a proof-of-concept of the push notification-based malware and evaluated its stealthiness and feasibility. Our malware implementation effectively evaded the existing malware analysis tools such as 55 antimalware scanners from VirusTotal and SandDroid. In addition, our backdoor implementation successfully cracked about 98% of all the tested unlock secrets (either PINs or unlock patterns) in 5 seconds with only a fraction (less than 0.01%) of the total power consumption of the device. Finally, we proposed several defense strategies to mitigate push notification-based malware by carefully analyzing its attack process. Our defense strategies include filtering subscription requests for push notifications from suspicious applications, providing centralized management and access control of registration tokens of applications, detecting malicious push messages by analyzing message contents and characteristic patterns demonstrated by malicious push messages, and detecting malware by analyzing the behaviors of applications after receiving push messages.

1. Introduction

Stealthiness is a key requirement of malware for the persistence of attack. *Remote triggering* can enhance the stealthiness of malware by allowing attackers to periodically trigger and execute their malware only whenever they want, while normally hiding the presence of malware from victims. In addition, it is also desirable for attackers to send attack commands to their malware for the flexible control of the malware's behaviors. Some examples of attack commands include the following: "perform a DDoS attack against foo.com at a certain transmission rate during a certain period of time"; "collect the screen lock password of the device"; and "gather pictures taken or SMS messages received on a certain date." Such features as remote triggering and delivering attack commands essentially require communication channels between attackers and their malware on victims' devices, and these communication channels are commonly referred to as *command and control (C&C) channels*.

Remote C&C channels of malware should meet the following requirement for their stealthiness. Many malware scanner tools can try to intercept the network traffic of the tested application and analyze the captured traffic for detecting suspicious behavior of malware. Thus, a secret C&C channel of malware should be established to avoid detection by scanners using the network traffic analysis.

Several groups of researchers presented mobile botnets exploiting push notification services to establish their C&C channels [1–3] and evaluated the feasibility of push notification services as C&C channels in terms of stealthiness, scalability, and efficiency. However, those previous studies mainly focused on the design of botnet using push notification services. In addition, those studies offered only a brief discussion on how to mitigate push notification-based mobile botnets.

This paper is based on our preliminary work [4]. In this paper, we extend the backdoor implementation presented in the preliminary work [4] to develop a more *generic* and *flexible* push-based backdoor design that can be applied to any type of malware (e.g., botnet, backdoor, and other malware types) for the purpose of setting up a C&C channel from the attacker via push notifications. In addition, we comprehensively analyze the attack procedure of push notification-based malware and propose a set of defense mechanisms to mitigate push-based malware in each step of the attack procedure.

Push notification services can be used to secretly hide the communication channel between malware and remote network parties (e.g., C&C server). This is because push notification services are also popularly used for benign applications, and thus this makes detection difficult. Moreover, push notification-based malware requires permissions that are only needed to use push notification services; thus our malware is resilient against malware detection techniques using permissions, which are also very commonly used in antimalware scanners. Because the behaviors of malware can easily be switched on and off by push messages from the attacker, the chance of detection can be reduced.

Sending a push message to malware requires the attacker's prior knowledge of the registration token of the installed malware. Therefore, a covert way of delivering the registration token of the malware to the attacker is needed for push notification-based malware. In this paper, we suggest using a public cloud-based database service (e.g., Firebase Realtime Database) as a medium of registration token delivery. Incorporating a public intermediate service makes it hard to distinguish the malicious activity of delivering the registration token from other benign activities using the same service.

To show the feasibility of our malware design, we implemented backdoor (as a representative type of push notification-based malware) on Android 5.1 and evaluated its stealthiness and feasibility. The evaluation results showed that existing malware detection tools based on static and/or dynamic analysis were ineffective to detect our backdoor implementation. Specifically, all the tested antimalware scanners (VirusTotal [5] and SandDroid [6]) failed to identify our backdoor as malware. In addition, we observed that our backdoor successfully cracked about 98% of all the tested unlock secrets (either 4-digit PINs or patterns) in 5 seconds with only a fraction (less than 0.01%) of the total power consumption of the device.

To reduce the risk of such malware, we carefully analyzed the push notification-based malware architecture and the attack procedure over it and consequently suggest several possible defense strategies to mitigate push notification-based malware. Our defense strategies include filtering subscription requests for push notifications from suspicious applications, providing centralized management and access control of registration tokens of applications, detecting malicious push messages by analyzing message contents, and detecting malware by analyzing the behaviors of applications after receiving push messages.

The remainder of this paper is organized as follows. The next section overviews the related work, and Section 3 gives a brief overview of Firebase. Section 4 defines the threat model considered in this paper. Section 5 presents the design of push notification-based malware. Section 6 describes the evaluation of our proof-of-concept malware implementation. Section 7 discusses possible defense strategies to mitigate push notification-based malware. Section 8 concludes this paper.

2. Related Work

There exist a number of malware examples that exploit various communication methods available on mobile devices. Several malware misused Short Message Service (SMS) for their malicious activities. Zeng et al. [7] presented a mobile botnet that uses SMS as a medium to deliver bot commands and studied a P2P network architecture suitable for the botnet in terms of the number of message transmissions and delay. Hua et al. [8] analyzed the impact of network topology on dissemination cost of bot commands over SMS-based mobile botnets. Mulliner et al. [9] and Anagnostopoulos et al. [10] proposed mobile botnets that use a hybrid of SMS, HTTP, and/or Bluetooth as a C&C medium. There also exists an SMS-based Trojan [11] that causes financial loss to users by sending messages to premium rate numbers without the users' recognition.

There is another group of malware examples which is based on other communication methods such as Bluetooth and social networking services. Android.Obad.OS [12] is a worm that propagates copies of itself via Bluetooth. Singh et al. [13] studied the feasibility of Bluetooth as C&C channels of mobile botnets. DR-SNBot [14] and Andbot [15] are mobile botnets that use social networking services as C&C channels. In these botnets, the botmaster posts bot commands disguised as image files on blog sites; then bots download the image files and extract the bot commands from the image file.

Most relevant to our research, several groups of researchers presented mobile botnets exploiting push notification services as their C&C channels [1–3] and evaluated the feasibility of push notification services as C&C channels in terms of stealthiness, scalability, and efficiency. In particular, Zhao et al. [1] proposed an approach that exploits upstream push messages for each bot to secretly deliver its registration token to the botmaster. Chen et al. [3] suggested the use of multiple push servers, not relying on a single server, to provide improved scalability and fault tolerance of mobile botnets. In our preliminary work [4], we presented a design of Android backdoor exploiting push notification services. Based on those previous studies, we designed a generic malware architecture exploiting push notification services to support remote triggering and flexible control of malware and implemented a backdoor prototype on Android devices to show the feasibility of that malware architecture. We also suggested possible defense strategies to mitigate push notification-based malware.

3. Overview of Firebase

Firebase (https://firebase.google.com) is a mobile application development platform to facilitate the development of applications using cloud-based services such as Firebase

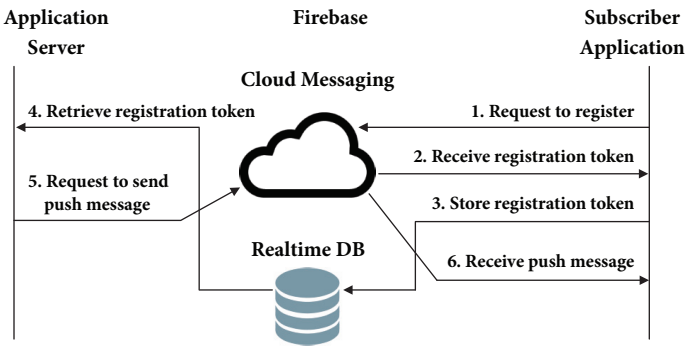


Figure 1: Overview of Firebase.

Cloud Messaging (FCM, https://firebase.google.com/products/cloud-messaging/) and Firebase Realtime Database (FRD, https://firebase.google.com/products/realtime-database/).

FCM is a cloud-based messaging service that allows third-party developers to send push messages to their applications on user devices. Figure 1 shows the infrastructure and the overall procedure of a push messaging service based on FCM. In this infrastructure, the application server managed by a third-party developer sends messages to the subscriber application on a user device via the FCM server.

To develop an application that can receive push messages via FCM, the developer first needs to sign up for a developer account of his (or her) application in Firebase with his (or her) Google ID and password and the package name of the application. As a result, the developer is given an application ID (a unique identifier of the developer's application), a sender ID (a unique identifier assigned to the developer as a push message sender), and an authorization key (a credential required to request the FCM server to send push notifications) from the FCM server. The developer then preconfigures his (or her) application package with the application ID and the sender ID given.

After the application package has been installed on a user device, the application first registers itself with the FCM server (Step 1 in Figure 1). This registration request contains the device ID (a unique identifier of the user device hosting the application) as well as the application ID and the sender ID that have been preconfigured with the application package by the developer. Once receiving this registration request, the FCM server generates and delivers to the application a *registration token* that uniquely identifies the application on this particular user device (Step 2).

Sending a push message to a subscriber application requires prior knowledge of the registration token of the recipient application. To achieve this, we use FRD, which allows sharing information between a subscriber application and an application server. After receiving the registration token from the FCM server, the subscriber application stores it in the FRD (Step 3), and then the application server downloads the registration token from the FRD (Step 4).

The application server is now ready to send a push message to the subscriber application. To send a push message, the application server sends the message content, the registration token of the target application, and the authorization key of the developer to the FCM server (Step 5). If the authorization key and the registration token are valid, the FCM server finally sends the message to the subscriber application (Step 6).

4. Threat Model

This section describes the threat model considered in our malware design.

To effectively distribute push notification-based malware, an attacker embeds his (or her) malicious codes into the codes of a normal application to disguise his (or her) malware as a legitimate application and distributes this repackaged application to victims. This method is referred to as *repackaging* and is known as a popular method to distribute mobile malware [16]. In practice, this repackaging method increases the chance that a victim installs the malware without being aware of the risk because the malware is embedded secretly into a legitimate application. Furthermore, the repackaged malware can be delivered to a victim via social engineering methods [17] (e.g., sending a gift app), and this further increases the chance that the malware is installed on the victim's device. If an insider attacker (e.g., friends, colleagues, or family members) often gains physical access to a victim's device, it is also possible that the attacker installs the malware secretly on the victim's device. Thus we assume that the malware is installed secretly on the smartphone of a victim.

For application repackaging, the attacker first selects a popular application as a host application for embedding the malicious codes and downloads the Android application package (APK) file of the selected application. The attacker then reconstructs the original codes of the host application in human readable format (e.g., smali codes) by unarchiving and decompiling the APK file of the host application. The next step is to add some malicious codes to the original application codes by either modifying the existing code file or adding a new separate code file and also modifying `AndroidManifest.xml` for additional configurations required for the malicious activities. The attacker then generates a new APK file by rebuilding the modified

application codes (with the malicious codes), signs the new APK file using his (or her) private key, and attaches the self-signed certificate of his (or her) public key. It is worth noting that the use of the self-signed certificate allows the attacker to claim whatever identity he (or she) wants in the certificate. The attacker distributes the new APK file attached with the certificate to mobile application stores. This repackaging method increases the chance that a victim installs the malware without being aware of the risk because the malware is embedded secretly into a legitimate application. We assume that the repackaged malware is installed secretly on the smartphone of a victim.

In this paper, we consider two types of malware, *backdoor* and *DDoS (Distributed Denial-of-Service) bot*, as examples of malware. The following describes our assumptions on each type of malware.

4.1. Backdoor. In this attack scenario, we assume an *insider* attacker (e.g., friends, colleagues, or family members) who often gains physical access to a victim's smartphone. A previous study [18] demonstrated that smartphone users are mostly concerned about preventing unauthorized access to the private information on their devices from such insiders. We assume that the victim's smartphone is protected with a screen lock mechanism that can provide a proper level of security (e.g., 4-digit PIN or screen lock pattern). We also assume that the victim regularly changes the unlock secret of the smartphone. Under these conditions, the objective of the *backdoor* installed on the victim's smartphone is to find the unlock secret of the device and delivers the secret to the attacker without being detected by the victim. That is, the attacker who wants to secretly look into the private information of the victim obtains the unlock secret of the victim's smartphone by triggering the backdoor secretly, unlocks the victim's smartphone, and investigates the private information inside the smartphone. In this way, the attacker can persistently obtain the private information on the victim's smartphone even when the unlock secret is often changed.

4.2. DDoS Bot. In this attack scenario, the attacker refers to the botmaster who has the control of bot applications installed on victims' devices. We assume that the attacker already knows the registration tokens of the bot applications so that the attacker can send push messages to the bot applications. FCM supports the maximum payload size of 4 KB (https://firebase.google.com/docs/cloud-messaging/concept-options), and we assume that this payload size is sufficient to contain a triggering signal and some basic information (e.g., the network identifiers of attack target hosts) that is necessary for attacks. The victims' smartphones should have capabilities to perform DDoS attacks. In fact, recent smartphones provide not only powerful computing capabilities but also a persistent Internet connection via cellular networks or Wi-Fi. Also, they support a high data transmission rate via, for instance, LTE (https://www.tomsguide.com/us/best-mobile-network,review-2942.html). Thus smartphones are powerful enough to perform DDoS attacks.

5. Design and Implementation of Push Notification-Based Malware

This section describes the design and implementation of push notification-based malware on Android devices. Our implementation is composed of three components: *triggering application*, *Firebase*, and *malware*.

Remote triggering of malware is important for hiding the malware from antimalware tools. We use FCM (Firebase Cloud Messaging) to achieve this by allowing an attacker to trigger malware via push notifications. Because FCM is also used by many normal applications demanding push notification services, it might be difficult for existing antimalware tools to distinguish the traffic of malware from the traffic of benign applications via FCM.

The triggering application is executed on the attacker's device, while malware is executed on the victim's device. In our implementation, Nexus 5X on Android 6.0 and Nexus 5 on Android 5.1 were, respectively, used for the attacker's and the victim's devices. Our malware implementation can be adapted to other Android versions as well. We confirmed that our malware also performs well on lower Android versions than Android 5.1 by slightly modifying the code. Consequently, our malware can currently be applied to 42.3% of all Android smartphones according to the latest statistics on Android version distribution (https://developer.android.com/about/dashboards).

5.1. Attack Procedure. This section describes the overall attack procedure in our malware design based on push notification services. As discussed in Section 4, we assume that malware is already installed on a victim's smartphone. The triggering application and malware in our implementation correspond to the application server and the subscriber application in Figure 1, respectively.

After being installed on the victim's smartphone, the malware sends the FCM server a request to register itself as a push message recipient (Step 1 in Figure 1), and the FCM server issues a unique registration token to the malware in response (Step 2). The malware then uploads the received registration token to the FRD (Firebase Realtime Database) (Step 3) so that the triggering application (on the attacker's device) can retrieve the registration token from the FRD (Step 4). After the triggering application obtains the registration token of the malware, the attacker is now ready to send a push message to the malware. Whenever the attacker wants to trigger the malware on the victim's smartphone, the attacker uses his (or her) triggering application to send the FCM server a push notification request that contains the registration token of the malware and a push message to deliver to the malware (Step 5); the push message in the request may contain a signaling message to trigger the malware and an attack command to deliver. Once receiving the push notification request with the registration token, the FCM server checks which application on which device has been assigned this token, which is the malware in this case, and then forwards the push message to the malware (Step 6). The push message that arrives at the victim's device is first delivered to the client-side agent of push notification service, which is running on the victim's device;

```
Authorization : key =" AIzaSyBCkAgv4−iucpSqdpd6g . . . "
{
        message : {
                token : " bk3RNwTe3H0 : CI2k_Hl3pja99tls . . . " ,
                data : {
                        body : <command to trigger and
                                control malware >,
                        . . .
                }
        }
}
```

Figure 2: JSON-formatted push message from triggering application.

then the agent process hands the push message over to the malware. Then the malware is triggered automatically, checks the attack command contained in the received message, and launches the malicious activities specified in the attack command such as cracking the victim's unlock password, accessing some private information of the victim, and transmitting attack packets to the target host of a DDoS attack.

5.2. Triggering Application. The triggering application is installed and executed on the attacker's device and requests the FCM server to send push messages to trigger and/or command the malware through HTTP. As already mentioned, the triggering application corresponds to the application server in Figure 1. In fact, any programming language and platform can be utilized to implement and run the triggering application. Figure 2 shows an example push message sent by the triggering application in JSON format. As mentioned in Section 3, `Authorization key` is required to request the FCM server to send push messages. The `token` field encloses the registration token that identifies the target malware to which the message should be eventually delivered.

5.3. Malware. This section describes the design and implementation of the malware, which corresponds to the subscriber application in Figure 1. *Stealthiness* is a key property required by the malware. To this end, we carefully designed the malware to provide the following properties. First, by adopting the remote triggering feature, our malware does not have to be always run on a victim's device. Instead, our malware is triggered and executed only when the attacker wants, and this can significantly reduce the chance of being detected by the victim. Second, all communications between the remote triggering application and malware can always be done through a push messaging service that is also popularly used by other benign applications. Hence, it is hard for malware detection tools based on network traffic monitoring to detect our malware. Third, our malware implementation requires the permissions (`WAKE_LOCK`, `ACCESS_NETWORK_STATE`, `INTERNET`, `<package-name>.permission.C2D_MESSAGE`, `com.google.android.c2dm.permission.RECEIVE`) that are only needed to use FCM, and these permissions are also required by other benign applications using FCM. Thus it is also difficult even for security-conscious users to notice the existence of our malware on their smartphones. For the same reason, malware detection techniques using permissions (e.g., [19]) are also ineffective for detecting our malware.

The following subsections describe two types of malware: *backdoor* and *DDoS bot*. In particular, we implemented the backdoor application to validate the feasibility of push notification-based malware.

5.3.1. Backdoor. The goal of backdoor application installed on a victim's device is to identify the unlock password of the victim and secretly deliver it to the attacker. Figure 3 shows the overall attack procedure of push notification-based backdoor. The attacker who wants to access the victim's device requests an FCM server to send a push message to the backdoor application that has been installed on the victim's device by using the triggering application (Step 1). The FCM server then sends the requested push message to the backdoor application (Step 2). Once the push message arrives at the victim's device (Step 3), the backdoor application is triggered automatically (Step 4) and identifies the lock mechanism (e.g., PIN or screen lock pattern) being used by the victim (Step 5). The backdoor application is able to know the lock mechanism in use by accessing `locksettings.db` file stored in `/data/system/`. Specifically, `lockscreen.password_type` field in the file indicates which lock mechanism is currently in use with an integer value; the value for PIN is `131072` or `196608`, and the value for screen lock pattern is `65536`.

After identifying the lock mechanism in use, the backdoor application tries to find out the password by executing the cracking module corresponding to the identified lock mechanism (Step 6). When storing a user input password in a file, Android stores the hash value of the password rather than the password itself. Thus cracking the password requires accessing the file where the hash value of the password is stored, and accessing this special file requires a root privilege. For this reason, it is assumed that the victim's device is rooted before the backdoor application is installed. While this assumption may seem rather strong, it is one that appears to be commonly made in practice. There exist several survey results showing that a large number of Android devices were rooted by device owners for the purpose of getting rid of unnecessary built-in apps or updating to the latest version of Android. According to the official report from Google in 2016 [20], about 5.6% of all Android users were using rooted devices, either intentionally or due to security bugs. Furthermore, the survey results in [21] showed that about 7% of all respondents were using rooted Android devices. Tencent also conducted a survey of Android users in China during 2014 [22] and found that about 80% of the survey participants were using rooted Android devices. To make matters worse, security bugs can make Android devices rooted forcibly [23].

We implemented the password cracking algorithm shown in Algorithm 1 to efficiently guess a victim's unlock password from the hashed password stored in the victim's device. In this algorithm, first of all, the backdoor application reads the hashed password (in line (1)) from either `gesture.key` (for screen lock pattern) or `password.key` (for PIN). The backdoor application also retrieves the corresponding salt (in line (7)) from `locksettings.db`, especially for PIN.

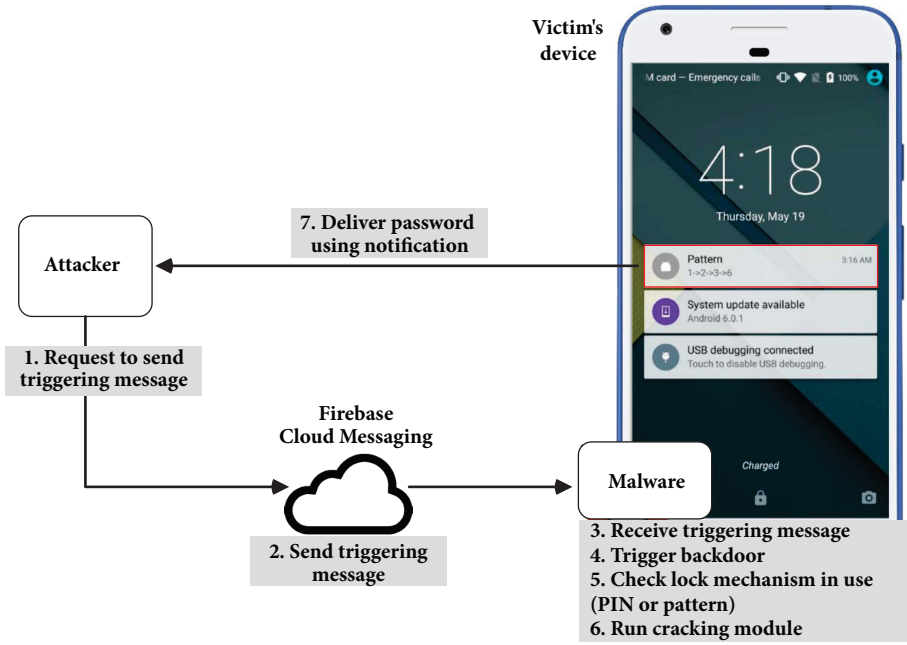


Figure 3: Attack procedure of push notification-based backdoor.

```
(1)  passwdHash ⟵ readPasswdHash()
(2)  oldPasswdHash ⟵ readOldPasswdHash()
(3)  if passwdHash = oldPasswdHash then
(4)      oldPasswd ⟵ readOldPasswd()
(5)      showNotification(oldPasswd)
(6)  else
(7)      randomSalt ⟵ readSalt()
(8)      while passwdToTest ⟵ readDictionary() do
(9)          saltedPasswd ⟵ passwdToTest ‖ randomSalt
(10)         if computeHash(saltedPasswd) = passwdHash then
(11)             showNotification(passwdToTest)
(12)             break
```

Algorithm 1: Password guessing algorithm.

To improve the efficiency of the password cracking procedure, the attacker can build a dictionary from real datasets of PINs (or patterns) and preconfigure the backdoor application with the dictionary. Algorithm 1 briefly summarizes our dictionary attack procedure. To avoid unnecessary computations, first of all, the password cracker checks if the victim's password has been updated by comparing the hash value of the old password found previously (`oldPasswdHash` in line (2)) with the current hash value (`passwdHash` in line (1)) read from `password.key` or `gesture.key` file. Only when `passwdHash` is different from `oldPasswdHash`, which means that the victim's password has been updated, does the password cracker proceed to the dictionary attack procedure (in lines (7)–(12)).

In the dictionary attack procedure, the password cracker first reads the salt from `locksettings.db` file only if the PIN is used by the victim (in line (7)). On the other hand, no salt is used for the pattern lock mechanism; thus the password cracker skips line (7) for patterns. The password cracker then reads the candidate passwords to test one by one from the PIN or pattern dictionary (in line (8)). If the candidate password is a PIN, then it is concatenated with the salt (in line (9)). For each candidate password, the password cracker computes the hash value of the candidate by calling *computeHash()* (in line (10)) and then compares the computed hash value with the hash value read from `password.key` or `gesture.key` file (in line (10)). If both hash values are the same, this means that the candidate password that has been tested is the victim's password. The internal process of *computeHash(input)* differs depending on whether the given input is a PIN or a pattern. If the input is a PIN, *computeHash(input)* returns *SHA-1(input)‖ MD5(input)*. Otherwise, if the input is a pattern, *computeHash(input)* returns *SHA-1(input)*.

Unlike 4-digit PINs, the Android pattern lock mechanism does not require a random salt, and this allows attackers to precompute and use the dictionary of all possible patterns to

shorten the time to guess a victim's pattern. However, for all 389,112 possible patterns, the dictionary file size is approximately 5.2 Mbytes, and embedding such a large dictionary file in the APK file of a legitimate application causes a significant increase in the size of the APK file. This significant change on the APK file could be a clue for detecting the backdoor. For such reasons, it is not recommended to use the dictionary of screen lock patterns.

After finding out the victim's password, the backdoor application finally informs the attacker of the password by showing up a notification popup (Step 7). A video demo of our push notification-based backdoor is available at (https://youtu.be/iyWhsYyPFnc).

5.3.2. DDoS Bot. DDoS bot application is designed to perform DDoS attacks against target servers and connected with the triggering application (controlled by the botmaster) through push notification services. Whenever the botmaster wants to launch a DDoS attack on a target server, the botmaster sends push messages to his (or her) bot applications via FCM servers. These push messages may contain some information necessary for DDoS attacks such as the network identifier of the target server and the transmission rate of attack traffic as well as a command to trigger the bot applications. Once receiving this push message from the triggering application, each DDoS bot application is automatically triggered and extracts the information of attack from the received push message. Based on the information retrieved from the push message, the bot application performs DDoS attacks against the target server.

6. Evaluation

This section shows the feasibility of push notification-based malware through our backdoor implementation. Specifically, we analyzed the stealthiness of the backdoor using commercial antimalware tools and also measured the execution time and power consumption of the backdoor.

6.1. Stealthiness of Push Notification-Based Malware. When a casual user encounters the installed backdoor on his (or her) Android device, it is difficult for him (or her) to notice that the backdoor is being used, since our backdoor never changes his (or her) unlock password. In addition, our backdoor is *temporarily* executed only when an attacker sends a specific push message to the backdoor.

To validate the resilience of our backdoor against malware detection, it is important to see whether our backdoor implementation could be detected by existing antimalware tools. For this validation, we analyzed the APK file of our backdoor application with several popular antimalware tools (e.g., VirusTotal [5] and SandDroid [6]). For a comparison purpose, we also analyzed the APK files of other normal applications under the same conditions.

We first analyzed the APK file of our backdoor application using 55 different types of antimalware scanners provided by VirusTotal [5]. However, all the scanners failed to classify our backdoor application as malware. This web page (https://goo.gl/5vOeuG) shows the detailed results of our analysis using VirusTotal.

SandDroid [6] provides both static and dynamic analyses of given Android APK files and returns the estimated risk score of the APK files in the range of 0 to 100; 0 and 100, respectively, represent the lowest and the highest level of risk. In this second experiment, we analyzed the APK file of our backdoor application using SandDroid, and the resulting risk score was 24. To understand the risk level of this score (24), we randomly selected 30 Google applications available on the Google Play Store and analyzed the APK files of all those applications in the same manner. The average risk score of all those applications was 26.47, which is very close to that of our backdoor application, and the standard deviation was 17.87. We also performed the same analysis for 30 malware samples. Unlike our backdoor application, their risk scores were all 100. From these results, we concluded that the existing antimalware scanners were ineffective to detect our backdoor application.

Repackaging malware in a legitimate application is a popular method to distribute Android malware [16]. If repackaging our backdoor causes significant changes on the original APK file, such changes could be a clue for detecting the backdoor. To see the impact of such repackaging on an APK file, we repackaged our backdoor in the APK file of a legitimate application that originally uses Firebase and investigated changes in the APK file in several aspects: required permissions, file size, total number of contained files, services, activities, intent filters, and so forth. In summary, the dictionary files and codes of the backdoor caused an increase in the APK file size by about 1.5 Mbytes and an increase in the number of contained files by 8. But, except these changes, no other changes were observed in the APK file.

6.2. Execution Time. The execution time of our backdoor is important because the longer the execution time is, the more likely it is to be detected. Thus we measured the execution time of the backdoor application by conducting both brute-force and dictionary attacks for PIN and screen lock pattern. We used existing real-world datasets of 4-digit PINs [24] and patterns [25]. The PIN dataset consists of 204,508 PINs collected from that number of iPhone users in 2011, and the pattern dataset consists of 389,112 patterns generated from the patterns collected from 312 Android users in 2015 using the 3-gram Markov model [25]. We used each of the entire datasets as a PIN and pattern dictionary for our dictionary attack. For measuring the execution time of our backdoor application, we randomly selected 5,000 PIN (or pattern) samples from the entire dataset of PINs (or patterns) and measured the average execution time (i.e., the average time taken to identify the lock mechanism and crack the tested PIN (or pattern)) of the backdoor application after it has been triggered by a push message.

Table 1 summarizes the experiment results. For 4-digit PINs, the dictionary attacks cracked 52.68% of all the tested PINs within 1 second, whereas the brute-force attacks cracked only 10.60%. The dictionary attacks show on average 1.5 times faster execution time than the brute-force attacks. For screen lock patterns, the dictionary attacks cracked 85.34% of all the

Table 1: Average password cracking time (B: brute-force attack, D: dictionary attack).

Time (sec)	4-digit PIN		Lock pattern	
	B	D	B	D
<1	10.60%	52.68%	44.10%	85.34%
<2	57.88%	74.88%	62.36%	93.76%
<3	70.98%	85.76%	68.90%	96.22%
<4	83.72%	93.34%	71.00%	97.54%
<5	94.28%	98.72%	76.60%	98.28%
Avg.	2.3 s	1.5 s	5.0 s	1.0 s

Table 2: Power consumption (J) of backdoor application and ratio (%) of how much of the total power usage is consumed by backdoor application (STD: standard deviation).

	Changed		Not changed	
	Pattern	**PIN**	**Pattern**	**PIN**
Avg.	74.597 J	10.393 J	0.043 J	0.050 J
STD	4.493 J	1.944 J	0.056 J	0.050 J
Ratio	10.862%	1.669%	0.007%	0.008%

tested unlock patterns within 1 second, whereas the brute-force attacks cracked only 44.10%. The brute-force attacks show on average 4.93 times slower execution time than the dictionary attacks.

6.3. Power Consumption. There exist several attempts to detect mobile malware by monitoring the power consumption patterns of applications [26–28]. For example, if our backdoor application consumes an excessive amount of battery power deviated from typical power usage, this would make our backdoor detected easily by antimalware systems based on power consumption monitoring. Thus, we performed experiments to investigate the power usage of our backdoor application. In these experiments, PowerTutor [29] was used to measure the power consumption of the backdoor. To see the impact of the password cracking process only, we compared the power consumption of the backdoor in two different cases; `Changed` and `Not changed`. In `Changed` case, the password is changed for every execution, and thus the password cracking process in the backdoor application is always executed. On the other hand, in `Not changed` case, the password is never changed, and so the only thing the backdoor application does is to show up a notification of the previously cracked password without performing the password cracking process.

Table 2 shows the average power consumption of the backdoor application and the ratio of how much of the total power usage is consumed by the backdoor. For 4-digit PINs, the backdoor application consumed only a small fraction of the total power usage in both cases of `Changed` and `Not changed`, whereas, for screen lock patterns, the backdoor application had a considerable impact on power consumption, especially in `Changed` case. In general, however, it is uncommon for users to frequently change the unlock passwords of their devices.

7. Push Notification-Based Malware Prevention

This section discusses possible defense strategies to mitigate push notification-based malware.

7.1. Filtering Registration Requests. The first line of defense against push notification-based malware is to prevent malicious applications from being registered into FCM servers. In order to achieve this, when receiving a registration request from a subscriber application, a FCM server should be able to determine whether the requesting application is suspicious or not and then stop the registration of the subscriber application if it is regarded as suspicious. This defense might be effective because push notification-based malware whose registration request has been denied by a FCM server is unable to receive any message or command from the attacker.

To detect suspicious applications, a FCM server can take advantage of a reputation system of mobile applications [30]. As an example, the reputation system counts how often a given application is observed in a large group of FCM users, and uncommon applications may be tagged as suspicious. If a given application, requesting a subscription for push notifications, is considered as suspicious according to the reputation system, then the FCM server denies the registration request. In this way, we can effectively block suspicious applications from receiving messages through push notification services.

In Figure 4, `Registration Manager` in the FCM server inspects registration requests incoming from subscriber applications on user devices and issues registration tokens only to applications that pass the checks through the reputation system.

7.2. Filtering Token Requests. In order for attackers to send push messages to their malicious applications, the attackers

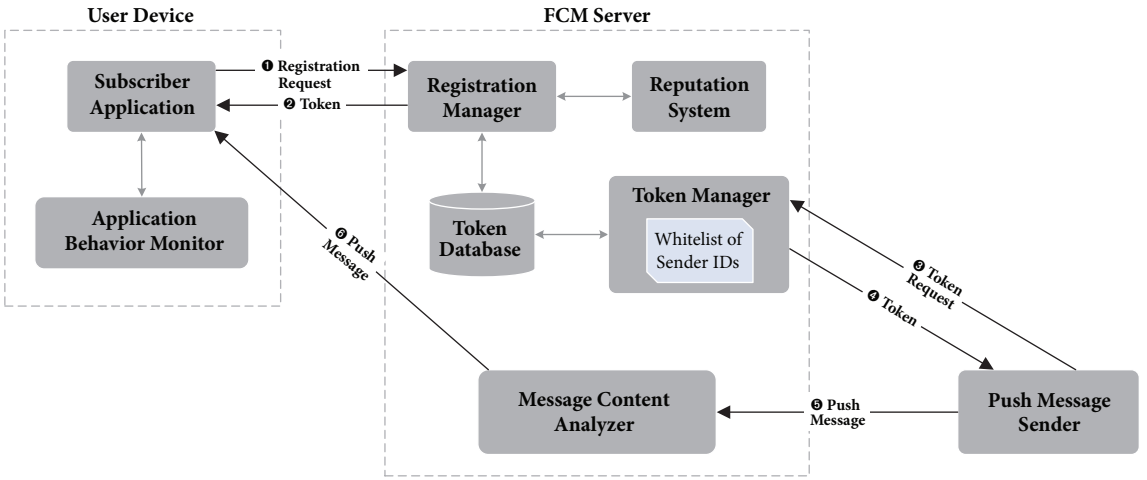


Figure 4: System architecture to mitigate push notification-based malware.

should first obtain the registration tokens of those applications. That is, if we prevent the attackers from obtaining the registration tokens of their malicious applications, we can disconnect the communication channels based on push notification between the attackers and their malicious applications.

In most push notification services including FCM, subscriber applications themselves forward their registration tokens to application servers after obtaining the tokens from the FCM server. That is, malware can easily forward the token to the attacker if the malware obtains the registration token.

A possible option to prevent this is not to allow subscriber applications to send out their tokens. Instead, the FCM server should manage the registration tokens of subscriber applications in a centralized manner and control the accesses of the tokens from application servers. In order to block the tokens stored in the user devices from being sent out, we need a way to detect and block attempts to send tokens outside, and techniques such as TaintDroid [31] can be utilized for this objective.

Senders who want to send push messages to their applications should first request the FCM server for the tokens of the applications, and the request contains the identifier of the sender. Once receiving the request for a token, the FCM server determines whether or not it can provide the token to the sender. For this filtering decision, the FCM server can maintain a whitelist of *trusted* sender IDs. That is, if a received token request is from a sender ID that is not included in the whitelist, the FCM server denies the request.

Although blacklisting is effective in preventing known malicious applications, it can be weak against preventing new ones. In practice, it is difficult to efficiently update the central database of malicious subscriber applications. Whitelisting, on the other hand, is effective against unknown malicious subscriber applications since only those considered 'trusted' are allowed to send push messages. Therefore, our recommendation would be to use whitelisting rather than blacklisting.

In Figure 4, `Token Manager` receives token requests from push message senders, searches the whitelist for the senders' IDs, and delivers the requested token only to the whitelisted senders.

7.3. Analyzing Push Message Contents. Another defense strategy is to detect malicious push messages by analyzing the contents of push messages. To avoid permission-based detection mechanisms, an attacker may repackage his or her malware in a legitimate application that originally uses a push notification service. Let app_x and app'_x denote the normal version and the malicious repackaged version of application x, respectively. From the attacker's perspective, it is important to minimize the difference between app_x and app'_x to avoid detection. Therefore, in many cases, the attacker is unlikely to modify the original behaviors of app_x while embedding the malicious logic into app_x. Then app'_x receives push messages from both the legitimate senders and the attacker. Under this condition, the malicious logic embedded in app'_x should be triggered only for push messages from the attacker while keeping silent for push messages from the legitimate senders.

Let us assume that the backdoor is repackaged from app_x. If the backdoor launches its password cracking activity even for push messages from the legitimate senders, the notification popup of the password shown by the backdoor could be found by the victim and this situation would increase the chance for the victim to detect the backdoor. To prevent this, the backdoor should be able to distinguish the attacker's push messages from other push messages from the legitimate senders so that the password cracking activity is launched only when it has been requested by the attacker. To achieve this, a push message from the attacker should include a specific pattern so that the malware can identify the attacker's command. In this case, it is possible to detect the attacker's messages by checking the existence of the specific patterns on received push messages.

In Figure 4, `Message Content Analyzer` in the FCM server inspects the contents of messages requested for push notifications by senders and filters out messages that exhibit malicious patterns.

We implemented a proof-of-concept of `Message Content Analyzer` which performs analysis of the contents of

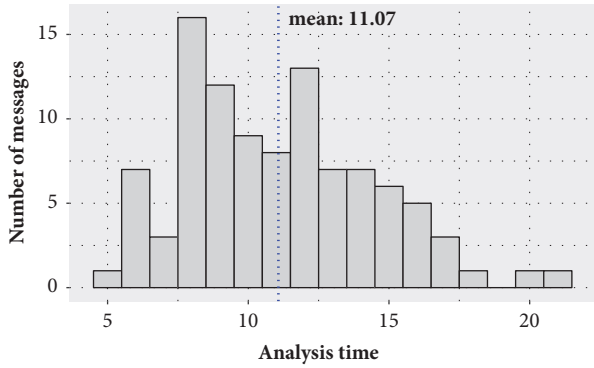


Figure 5: Distribution of the analysis times observed for 100 push messages (the blue dotted line shows the average analysis time).

push messages with a rule-based model. We assume that the analyzer has a database of malicious patterns. When the analyzer receives a push message, it verifies that each malicious pattern defined in the database is included in the received push message. If the received push message contains any of the malicious patterns defined in the database, the analyzer drops the message so that the message is not delivered to the user device.

To show the feasibility of `Message Content Analyzer`, we evaluated our proof-of-concept with 100 push messages by measuring the time taken to analyze those messages. For this evaluation, we used Google Pixel 2 with Qualcomm Snapdragon 835, 4 GB RAM, and Android 8.0 Oreo. On average, our analyzer took 11.07 milliseconds for analyzing each message (standard deviation was 3.37 milliseconds). Figure 5 shows the distribution of the analysis times observed for the 100 messages. The x-axis shows the observed analysis times ranging from 5 to 21 milliseconds, and the y-axis shows the number of messages showing each analysis time in that range. The most frequently observed analysis time was 8 milliseconds, and the analysis times of 72 out of 100 messages fell into the range of 8 to 14 milliseconds.

7.4. Machine Learning-Based Defense Strategy. To improve the effectiveness of suspicious push message detection, machine learning techniques can be utilized. To achieve this goal, we can build a classifier with a set of features that can be collected by the FCM server for sending a push message. Naturally, it is critical to choose proper features of push messages that can be used to distinguish the push messages for malware from push messages for normal applications.

A promising candidate feature is the number of recipient applications of a push message because the number of applications receiving the attack messages is typically much smaller than the number of applications receiving the normal messages. Also, push message size, interarrival time, and keywords used in push messages can be used as reasonable features for our purpose.

7.5. Monitoring Application Behaviors. Another defense strategy against push notification-based malware is to monitor and analyze an application's behaviors after the application receives push messages. If an application exhibits behaviors similar to those of known malware or if an application's behaviors deviate from normal behavior patterns, then we can consider the application as abnormal application and take appropriate counteractions.

Our backdoor prototype, for example, has the following behavior patterns: after being triggered by a push message received from the triggering application, the backdoor application first accesses the password hash file, performs computationally intensive tasks, and then pops up a notification. If some application exhibits such behaviors, then we can suspect the application as backdoor. As another example, we can consider a push message-based bot application [1, 2]. This bot application typically transmits a large amount of traffic to a specific host after being triggered by the received push message. In other words, it shows a behavior characteristic that transmits a very large number of messages after receiving a few push messages. Therefore, we can detect the push message-based bot application by monitoring the application's transmission patterns after receiving a push message.

7.6. Reviewing and Analyzing Security Sensitive Events. Another mitigation strategy is to keep the history of all security-sensitive events that have occurred on a user device for later investigation. For this, Google or a device manufacturer's server can collect security-sensitive events from user devices, and the device owners who want to know what happened on their devices can review the collected events by visiting the server website. Reviewing the history of suspicious events increases the chance of detecting the malware.

Another possible approach is to collect and analyze diverse types of events occurring on a user device using machine learning algorithms. Through this analysis, usual patterns of events observed on the user device can be identified. By monitoring the deviation from the usual patterns, it is possible to automatically detect suspicious events.

8. Conclusions

We presented a design of push notification-based malware which allows attackers to remotely trigger and control malware on victims' devices. We also implemented a backdoor prototype based on our malware design on Android devices and evaluated its stealthiness and feasibility. The evaluation results showed that our malware implementation could effectively evade existing malware detection tools. In addition, our backdoor successfully cracked about 98% of all the tested PINs and patterns in 5 seconds with only a fraction (less than 0.01%) of the total power consumption of the device. Finally, we proposed several defense strategies to mitigate push notification-based malware by carefully analyzing its attack process.

Data Availability

The source codes of our implementation and the data files of the evaluation results are available at a GitHub repository (https://github.com/rymuff/os-data-availability). The GitHub repository contains the URL links to the source codes

of the triggering application and the backdoor application. The repository also contains the URL links to the following data files of the evaluation results: the data files of the analysis results of the backdoor application using VirusTotal and SandDroid and the data files of the execution time and power consumption measurements of the backdoor application.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the MSIT (Ministry of Science and ICT), Republic of Korea, under the ITRC (Information Technology Research Center) support program (IITP-2018-2015-0-00403) supervised by the IITP (Institute for Information & communications Technology Promotion) and by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2016R1A6A3A11930593).

References

[1] S. Zhao, P. P. Lee, J. C. Lui, X. Guan, X. Ma, and J. Tao, "Cloud-based push-styled mobile botnets: a case study of exploiting the cloud to device messaging service," in *Proceedings of the the 28th Annual Computer Security Applications Conference*, pp. 119–128, Orlando, FL, USA, December 2012.

[2] H. Lee, T. Kang, S. Lee, J. Kim, and Y. Kim, "Punobot: mobile botnet using push notification service in android," in *Proceedings of the International Workshop on Information Security Applications*, pp. 124–137, Springer, 2013.

[3] W. Chen, X. Luo, C. Yin, B. Xiao, M. H. Au, and Y. Tang, "Muse: Towards robust and stealthy mobile botnets via multiple message push services," in *Australasian Conference on Information Security and Privacy*, pp. 20–39, Springer, 2016.

[4] J. Cho, G. Cho, S. Hyun, and H. Kim, "Open sesame! design and implementation of backdoor to secretly unlock android devices," *Journal of Internet Services and Information Security*, vol. 7, no. 4, pp. 35–44, 2017.

[5] V. Total, "VirusTotal-Free online virus, malware and URL scanner," https://www.virustotal.com.

[6] B. R. Team et al., "SandDroid: An APK Analysis Sandbox. Xian Jiaotong University," https://sanddroid.xjtu.edu.cn.

[7] Y. Zeng, K. G. Shin, and X. Hu, "Design of SMS commanded-and-controlled and P2P-structured mobile botnets," in *Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '12)*, pp. 137–148, April 2012.

[8] J. Hua and K. Sakurai, "A SMS-based mobile botnet using flooding algorithm," in *IFIP International Workshop on Information Security Theory and Practices*, pp. 264–279, Springer, 2011.

[9] C. Mulliner and J.-P. Seifert, "Rise of the iBots: owning a telco network," in *Proceedings of the 5th International Conference on Malicious and Unwanted Software (Malware '10)*, pp. 71–80, IEEE, October 2010.

[10] M. Anagnostopoulos, G. Kambourakis, and S. Gritzalis, "New facets of mobile botnet: architecture and evaluation," *International Journal of Information Security*, vol. 15, no. 5, pp. 455–473, 2015.

[11] G. Andre and P. Ramos, "Boxer sms trojan," Technical Report, ESET Latin American Lab, Bratislava, Slovakia, 2013.

[12] E. Messmer, "Backdoor.androidos.obad.a," *Network World*, Oct. 2013, (Accessed on Jan. 30, 2018), http://contagiominidump.blogspot.in/2013/06/backdoorandroidosobada.html=0pt.

[13] K. Singh, S. Sangal, N. Jain, P. Traynor, and W. Lee, "Evaluating Bluetooth as a medium for botnet command and control," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 61–80, Springer, 2010.

[14] T. Yin, Y. Zhang, and S. Li, "DR-SNBot: a social network-based botnet with strong destroy-resistance," in *Proceedings of the 9th IEEE International Conference on Networking, Architecture, and Storage (NAS '14)*, pp. 191–199, IEEE, August 2014.

[15] C. Xiang, F. Binxing, Y. Lihua, L. Xiaoyi, and Z. Tianning, "Andbot: towards advanced mobile botnets," in *Proceedings of the 4th USENIX Conference on Large-Scale Exploits and Emergent Threats*, p. 11, USENIX Association, 2011.

[16] W. Zhou, Y. Zhou, X. Jiang, and P. Ning, "Detecting repackaged smartphone applications in third-party android marketplaces," in *Proceedings of the 2nd ACM Conference on Data and Application Security and Privacy (CODASPY '12)*, pp. 317–326, ACM Press, San Antonio, TX, USA, Feburary 2012.

[17] S. Grzonkowski, A. Mosquera, L. Aouad, and D. Morss, "Smartphone security: an overview of emerging threats," *IEEE Consumer Electronics Magazine*, vol. 3, no. 4, pp. 40–44, 2014.

[18] I. Muslukhov, Y. Boshmaf, C. Kuo, J. Lester, and K. Beznosov, "Know your enemy: the risk of unauthorized access in smartphones by insiders," in *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '13)*, pp. 271–280, ACM Press, Munich, Germany, August 2013.

[19] Y. Zhou and X. Jiang, "Dissecting android malware: characterization and evolution," in *Proceedings of the 33rd IEEE Symposium on Security and Privacy (S&P '12)*, pp. 95–109, San Francisco, Calif, USA, May 2012.

[20] A. Ludwig and M. Mille, "Diverse protections for a diverse ecosystem: Android security 2016 year in review," https://goo.gl/6o4tBf, March 2017, https://source.android.com/security/reports/Google_Android_Security_2016_Report_Final.pdf=0pt.

[21] F. Howarth, "Is rooting your phone safe? the security risks of rooting devices," https://goo.gl/axbkX9, October 2015, https://insights.samsung.com/2015/10/12/is-rooting-your-phone-safe-the-security-risks-of-rooting-devices=0pt.

[22] A. Boxall, "80% of android phone owners in china have rooted their device," https://goo.gl/dH4zQZ, April 2015, http://www.businessofapps.com/80-android-phone-owners-china-rooted-device=0pt.

[23] H. Zhang, D. She, and Z. Qian, "Android root and its providers: a double-edged sword," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*, pp. 1093–1104, Denver, CO, USA, October 2015.

[24] H. Kim and J. H. Huh, "PIN selection policies: are they really effective?" *Computers & Security*, vol. 31, no. 4, pp. 484–496, June 2012.

[25] Y. Song, G. Cho, S. Oh, H. Kim, and J. H. Huh, "On the Effectiveness of Pattern Lock Strength Meters," in *Proceedings of the the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*, pp. 2343–2352, Seoul, Republic of Korea, April 2015.

[26] H. Kim, J. Smith, and K. G. Shin, "Detecting energy-greedy anomalies and mobile malware variants," in *Proceedings of the*

6th International Conference on Mobile Systems, Applications, and Services (MobiSys '08), pp. 239–252, Breckenridge, CO, USA, June 2008.

[27] T. K. Buennemeyer, T. M. Nelson, L. M. Clagett, J. P. Dunning, R. C. Marchany, and J. G. Tront, "Mobile device profiling and intrusion detection using smart batteries," in *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS '08)*, p. 296, Waikoloa, HI, USA, January 2008.

[28] L. Liu, G. Yan, X. Zhang, and S. Chen, "VirusMeter: preventing your cellphone from spies," in *International Workshop on Recent Advances in Intrusion Detection*, pp. 244–264, Springer, 2009.

[29] L. Zhang, B. Tiwana, Z. Qian et al., "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *Proceedings of the 16th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS '16)*, pp. 105–114, Scottsdale, AZ, USA, October 2010.

[30] Z. Ramzan, V. Seshadri, and C. Nachenberg, "Reputation-based security: an analysis of real world effectiveness," *Symantec Corporation*, 2009.

[31] W. Enck, P. Gilbert, S. Han et al., "Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones," *ACM Transactions on Computer Systems*, vol. 32, no. 2, p. 5, 2014.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 5932183, 13 pages
https://doi.org/10.1155/2018/5932183

Research Article

A Secure Incentive Scheme for Vehicular Delay Tolerant Networks Using Cryptocurrency

Youngho Park,[1] **Chul Sur**,[2] **and Kyung-Hyune Rhee**[1]

[1]*Department of IT Convergence and Application Engineering, Pukyong National University, Busan, Republic of Korea*
[2]*Department of Information Security, Busan University of Foreign Studies, Busan, Republic of Korea*

Correspondence should be addressed to Kyung-Hyune Rhee; khrhee@pknu.ac.kr

Received 23 February 2018; Revised 8 May 2018; Accepted 20 May 2018; Published 8 July 2018

Academic Editor: Ilsun You

One remarkable feature of vehicular ad hoc networks is characterized by an opportunistic communications by means of store-carry-forward message relaying which requires the cooperation of vehicles on the networks. However, we cannot be sure that all vehicles willingly contribute their computing resources to the networks for message forwarding with no rewards for their efforts in real-world scenarios. In addition, unfortunately, there may exist some selfish and greedy node which may not help others but tend to take their own gain. To cope with this challenge, incentive mechanisms are generally considered as the promising solution. In this paper, we design a Bitcoin-based secure and reliable incentive scheme for cooperative vehicular delay tolerant networking services. Bitcoin is the well-known worldwide cryptocurrency and digital payment system whose implementation relies on cryptographic techniques, which makes it possible to develop a practical credit-based incentive scheme on the vehicular networks at a low cost. We also implement Bitcoin transaction scripts to handle our proposed incentive scheme.

1. Introduction

It is trend of modern vehicles to equip GPS-based navigation system with digital map and on-board unit (OBU) devices which allow vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications. Due to the advances of modern car technologies incorporating with wireless communication, vehicular communications have been an active research area over the last decade. In particular, we have also witnessed 5G connected vehicular communications testbed in South Korea recently. As a result, up to date, a variety of vehicular ad hoc network (VANET) applications have been researched to provide not only comfortable transportation services but also location-based infotainment services on the road. Another attractive vehicular network architecture is vehicular delay tolerant networks (VDTNs), focusing on the opportunistic connectivity among vehicles and road side units, in which vehicles contribute to a message relaying service by utilizing store-carry-forward paradigm [1, 2]. As shown in Figure 1, some information collected at a source location (S) can be stored, carried, and then forwarded to a destination location (D) by a vehicle passing through the roads. An example of such opportunistic networking applications is to deliver some location-aware information such as gas and parking around S to the display located at D. Another application is to forward messages to the Internet through the gateway server located at D when Internet connection is not possible at S.

However, because VANETs and VDTNs are autonomous and self-organized networks with the cooperation among vehicles, we cannot always expect that all vehicles voluntarily contribute their computing resources to the network. Moreover, some selfish vehicles would not help message relaying service for others while they enjoy the services provided by the network. One solution to this challenge is to provide incentives to stimulate vehicles to voluntarily participate in the networks by rewarding for their contribution with an actual money or credit. For example, when a sender asks a vehicle for help, the sender gives some incentive to the vehicle so that the vehicle is willing to store, carry, and forward sender's message to a destination.

Although, on the one hand, an incentive looks like an attractive approach to selfish behavior, another concern is how the sender can be convinced whether the message

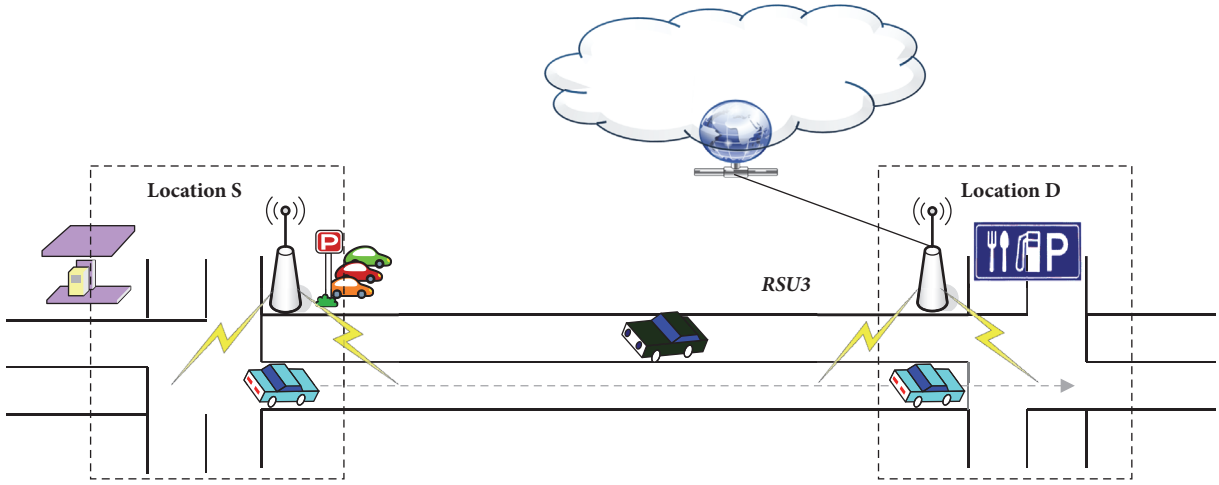


Figure 1: An example of store-carry-forwarding scenario in VANETs.

is actually delivered to the destination or not [3, 4]. If a source location server first gives credits to a vehicle, a malicious vehicle will not faithfully store-carry-forward the message to the specified destination after receiving the credit. The source location server may be concerned about such malicious behavior of a vehicle so-called dine and dash, such situation is unfair to the source location server. Therefore, it is also a critical challenge how to resolve such unfairness issue of incentive scheme on autonomous vehicular networks.

1.1. Related Work. While vehicular communications have received a great deal of attentions, the researches on secure vehicular communications have also widely carried out, and most of them are focusing on anonymous authentication schemes implemented by digital signature combined with pseudonyms of vehicles so as to guarantee nonrepudiation and anonymity of the communication activities simultaneously [5–12]. However, ordinary digital signature itself cannot deal with the fairness problem. To resolve the fairness problem in store-carry-forward message delivery, Lin et al. proposed a location-release signature [4] in which the credit signed by a source location server becomes valid if the credit reaches to a specific destination location server. However, they did not present how to actual incentives are rewarded to the vehicles.

Incentive schemes for cooperative VANET or VDTN environments can be categorized into reputation-based scheme and credit-based scheme. In reputation-based schemes, network nodes evaluate trust relationship with each other and vote on neighboring node's activity of message forwarding so that uncooperative nodes of bad reputation are excluded from the networks [13, 14]. Li et al. proposed an announcement scheme for VANETs based on a reputation system which allows for vehicles to evaluate message reliability [13]. In their scheme, the reliability of a message is evaluated by the reputation of the vehicle which generates the message, and the reputation score is collected, updated, and certified by a trusted third party.

Credit-based schemes employ some form of virtual currency for regulating message forwarding among different nodes and rewarding nodes for their helps [15–17]. When a source node needs help of other nodes for message forwarding, the source node should pay a certain amount of virtual coins to the helper nodes. To incentivize nodes for DTNs, Zhu et al. proposed a secure multilayer credit-based incentive scheme, named SMART [15], which provides nodes with virtual coins to charge for and reward the provision of data forwarding. They also briefly discussed several security issues in DTNs and countermeasures; however, they did not consider fairness issue. With regard to fairness issues in VDTNs, Lu et al. proposed a secure and practical incentive protocol Pi, which is a hybrid model combining reputation and credit, using verifiably encrypted signature technique.

However, those schemes additionally require implementing an application-dependent reputation management system or a virtual coin management system on VANETs. Furthermore, the existing incentive schemes entirely rely on a central trusted third party to assign some virtual coins to each node and to keep track of issued virtual coins in the system.

1.2. Contribution. In this paper, we present a secure credit-based incentive scheme for cooperative VDTNs integrating with a blockchain-based cryptocurrency system. Bitcoin is the most famous and practical cryptocurrency, whose implementation relies on cryptographic techniques and a distributed electronic payment system in which no trusted third party is required. By taking advantage of the Bitcoin system or Bitcoin overlay network, we can design a secure message delivery service and credit-based incentive scheme for VDTNs at a low cost.

As compared to the existing credit-based scheme, we do not need to be concerned about the reliability of virtual coin rewards on VDTNs. Instead, reliable virtual coin exchange transactions are shifted to the Bitcoin system. Moreover, we do not need to implement public key and pseudonym management system such as vehicular-PKI to authenticate

Table 1: Some remarkable cryptocurrencies based on blockchain.

project	launched	anonymity	remarks	ref.
Bitcoin	2009	pseudonymous	(i) first blockchain-based cryptocurrency (ii) uses proof-of-work timestamping	[20]
Namecoin	2011	censorship registance	(i) first altcoin that implemented Satoshi's BitDNS idea (ii) key-value pair registration and transfer	[21]
Litecoin	2011	pseudonymous	(i) technical detail is similar to Bitcoin (ii) aims to process a block every 2.5 minutes, rather than Bitcoin's 10 minutes (iii) uses scrypt in its proof-of-work algorithm	[22]
Monero	2014	pseudonymous, unlinkable	(i) implemented based on the CryptoNote protocol (ii) uses ring confidential transactions technique	[23]
Ethereum	2015	pseudonymous	(i) provides Turing-complete virtual machine, Ethereum Virtual Machine (EVM) (ii) distributed computing platform featuring smart contract functionality (iii) Ether is a cryptocurrency generated by the Ethereum platform	[24]
Zcash	2016	strong anonymity, shielded transactions	(i) decentralized cryptocurrency that provides strong privacy protections (ii) transactions can be controlled by a zero-knowledge proof called zk-SNARKs	[25]

vehicles participating in store-carry-forward communications because the private and public key pair of Bitcoin account owned by the vehicle/user can be used for vehicular communications on VDTNs as well as handling Bitcoin transactions rewarded as incentives. That is, the Bitcoin public key can be viewed as vehicle's pseudonym.

However, ordinary Bitcoin public keys are not sufficient to incorporate trustworthiness from real-world entities into the system, but vehicular networking is a cyberphysical system implemented in real world. We present authenticated vehicular communication using certified Bitcoin public key technique [18] to validate the legitimacy of the entity taking part in the system. Even though a trusted authority is involved in the proposed system for issuing certified public keys, we can more efficiently implement authentication system than the existing vehicular-PKI.

In our preliminary version of this work [19], we presented how Bitcoin can be used as incentive in VANET but did not show implementation details. In this paper, we show that how the fairness issues of incentive scheme on VDTNs can be resolved by handling Bitcoin transactions which transfer coins to a volunteer vehicle as incentives and implement Bitcoin locking and unlocking scripts to control the redemption conditions to spend the coins specified in the incentive transactions.

The rest of this paper is organized as follows: we first shortly introduce the technical concept of the Bitcoin needed to understand our proposed system in Section 2, then we design the proposed Bitcoin-based secure incentive system on VDTNs in Section 3. We discuss the security features of the proposed system in Section 4 and finally conclude this paper in Section 5.

2. Background

Cryptocurrency is a digital asset to support a medium of exchange based on cryptography to secure its transactions and to verify the transfer of assets. As opposed to centralized electronic cash and banking systems, cryptocurrencies are maintained by decentralized control through a blockchain functioning as a distributed ledger. Since the first implementation of decentralized cryptocurrency, Bitcoin, numerous alternative coins (altcoins) have been created. Table 1 summarizes some remarkable cryptocurrencies and their technological characteristics. Due to the cost effectiveness in validating transactions and the security of immutable ledgers on a distributed blockchain, the concept of blockchain is evolving to a platform beyond the cryptocurrency to develop decentralized applications and collaborative organizations to remove the need for a trusted third party.

In the followings, to understand blockchain-based cryptocurrency system, we briefly give a general overview of the Bitcoin on which our proposed incentive scheme is built.

2.1. Bitcoin System. Bitcoin is the first cryptocurrency which is a new form of a decentralized electronic cash system introduced by Satoshi Nakamoto [26]. Unlike traditional currency systems relying on a central authority such as a bank, Bitcoin is based on Peer-to-Peer (P2P) network and distributed consensus protocol without a trusted third party.

In Bitcoin system, each user has a private and public key pair to sign the transactions for coin transfers, and the address to uniquely identify a user is represented by a cryptographic hash of the public key for the respective user. The address is associated with user's account and the private key is used to sign transactions for spending coins.

Bitcoin payments are processed by generating transactions which transfer the values of coins from one user's account to another. Transactions are composed by senders and distributed to the Bitcoin P2P network, then the validity of the transactions is verified by Bitcoin network nodes called *miners*. After validating the transactions pended for a given time period, miners collect the transactions into a single unit called *block*. The new block accepted by the miners according to a consensus protocol is then added to the Bitcoin public ledger called *blockchain*.

2.2. Transaction. Bitcoin transaction is the record implying that transfers the value of coins from a sender to a recipient as shown in Figure 2. A transaction (TX) has a unique identifier and consists of a set of inputs and outputs which are key components of the transaction. Each input specifies unspent coins, belonging to a particular user, of the previous transaction identified by its hash code. Each output represents the amount transferred to the specified address of a recipient. To authorize spending the coins in the transaction input, the user should present his/her signature for the transaction and corresponding public key. Bitcoin uses ECDSA as a digital signature scheme.

Unlike traditional bank services in which transactions are verified and maintained by a central bank, transaction verification in Bitcoin system is distributed to P2P network nodes and only the valid transactions are recorded in the distributed ledger by means of a blockchain.

The concept of processing Bitcoin transactions, which spends outputs of previous transactions, is to manage transaction chain which transfers coin ownership from a sender to a recipient [27]. Therefore, sending some Bitcoin is creating an unspent transaction output (UTXO) locked to a specific public key owner, and a new transaction can consume one or more UTXOs as transaction inputs.

2.3. Script. Each UTXO has to be bound to a specific user eligible for spending the coins in it. Bitcoin system introduces a scripting language to describe the essential conditions (encumbrances) to claim the coins. That is, each transaction output can contain a locking script which defines the conditions that must be met to spend the coins associated with the UTXO. One dominant script supported by today's Bitcoin system is Pay-to-Public-Key-Hash (P2PKH) which encumbers the output with a public key hash known as address. An output locked by a P2PKH script can be unlocked by the user who can present a public key and a signature generated by the corresponding private key. A P2PKH transaction output would have the following script of form (we omit detailed operation of each script command):

```
scriptPubKey: OP_DUP OP_HASH160 <PubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
```

To authorize spending the output, the corresponding input specifies an unlocking script of the form:

```
scriptSig: <Signature> <PubKey>.
```

Another interesting transaction to us is MultiSig transaction which requires multiple signatures to unlock the encumbrance. MultiSig transaction outputs are usually denoted as M-of-N, where N is the total number of public keys and M is the minimum number of signatures required for redeeming the transaction output. For example, 2-of-3 MultiSig transaction of the following script

```
scriptPubKey: 2 <PubKey A> <PubKey B> <PubKey C> 3 OP_CHECKMULTISIG
```

can be redeemable by including any combination of two signatures of the private keys corresponding to the three listed public keys as follows:

```
scriptSig: OP_0 <Signature A> <Signature C>, or
scriptSig: OP_0 <Signature B> <Signature C>
```

2.4. Time-Locked Transaction. Bitcoin supports both transaction-level and script-level time-lock features which restrict the spending of outputs of the time-locked transactions by a certain time in the future. The functions of time-locks are useful for postdating transactions and withholding redemption of funds to a date in the future. We are interested in script-level time-locks.

There are two types of time-locks in the Bitcoin system: one is absolute time-lock and the other is relative time-lock. Absolute time-lock transactions use `CHECKLOCKTIMEVERIFY` opcode to specify a fixed date in the future when the output of the transaction can be spent, and relative time-lock transactions use `CHECKSEQUENCEVERIFY` opcode to establish amount of time far from the transaction publishing time.

2.5. Blockchain. Blockchain is a linked-list type data structure which maintains entire transaction history in terms of blocks. Figure 3 shows the blockchain structure used in the Bitcoin. When a transaction is generated and distributed to the Bitcoin network, some node called miner in the network

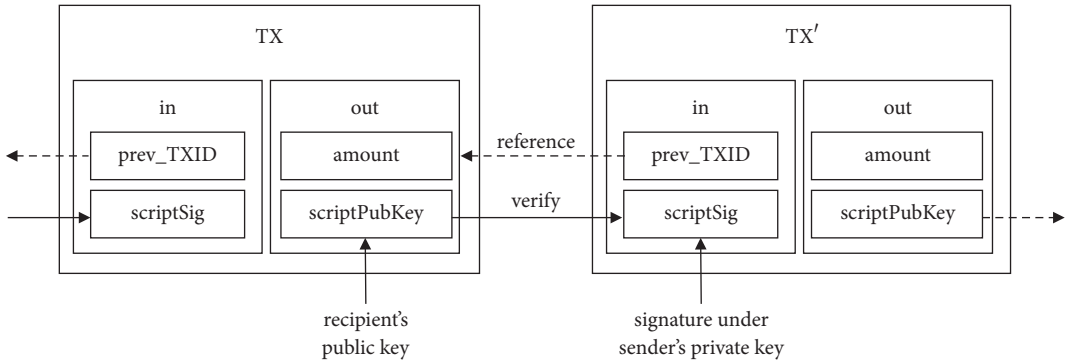


Figure 2: Overview of Bitcoin transaction structure and spending.

collects and verifies the pending transactions for a given time period to form a new block. Each block header contains the hash value pointing to the previous block and root of Merkle hash tree constructed from the transactions specified in the block. Once a block grouping some transactions is added to the blockchain, it means that a majority of miners verified the legitimacy of the transactions and validated the block through a probabilistic distributed consensus protocol with a Proof-of-Work (PoW) implemented by a complex cryptographic puzzle.

In order to agree on a common order of transactions and to ensure consistent state of the blockchain in a distributed system, Bitcoin is employing the PoW by varying a nonce value in the block until the hash value becomes lower or equal to the given difficulty target value, i.e., finding a random nonce such that $\texttt{Hash(header, nonce)} \leq \texttt{target}$. Bitcoin uses SHA-256 cryptographic hash function, and it is computationally difficult to find a desired hash value. If a majority of miners verify a block by solving a computationally hard PoW puzzle, then the new block is broadcasted to the network and successfully added to the blockchain. Other nodes in the Bitcoin network can easily verify the block by recalculating the hash value for the nonce given in the block header and comparing with target value. By making use of the PoW-based consensus protocol, Bitcoin system makes it hard to abnormally manipulate blockchain. Therefore, the blockchain can be viewed as a distributed immutable ledger.

3. Proposed System

In this section, we describe the proposed system architecture to design a Bitcoin-based secure and reliable incentive scheme for VDTNs. Anonymity of the vehicles participating in the communication is guaranteed by the use of Bitcoin public keys, and the vehicles are stimulated to help message store-carry-forward from one location to another location by paying them Bitcoin as incentives. Moreover, the fairness to a source location server is guaranteed by exploiting the 2-of-2 MultiSig transaction, in which the signature of the destination server is required as well as vehicle's, so that the forwarding vehicle can be allowed to spend the coins given as incentives once the vehicle actually arrives at the destination point.

3.1. System Model. Opportunistic networking applications on VDTNs can be characterized by store-carry-forward communications with the help of moving vehicles in the circumstance where a direct communication link is not always possible between source to destination. To design a Bitcoin-based incentive scheme, we consider the information dissemination service scenario as shown in Figure 4 where a vehicle helps forwarding some messages received from the source server to the destination point that displays the information such as commercial ad for the source server location.

(i) Service manager (SM) controls roadside units and authorizes vehicles participating in message dissemination service on VDTNs. SM issues certified public keys to the authorized roadside units and vehicles for authenticated vehicular communications and Bitcoin incentive transactions. The public key of SM for certified key generation is known to other entities in the system.

(ii) Vehicles participating in the system are equipped with OBUs embedding LTE/5G mobile communication for Internet connection, 802.11p for V2I and V2V communications [28] and navigation system with digital map. For handling Bitcoin transactions, Bitcoin client module is also installed in the vehicle.

(iii) Roadside units are under the control of SM and have 802.11p wireless link for communicating with vehicles passing by them, but not all roadside units have end-to-end or direct communication among them, so it is made in an opportunistic way with the help of moving vehicles.

We assume that the owners/users of both roadside servers and vehicles have their Bitcoin accounts to give and receive Bitcoin as incentives. When a source server asks for a vehicle to transfer a message bundle to a certain destination point, the source server publishes a Bitcoin transaction to the Bitcoin network for paying incentives to the vehicle. The source server's Bitcoin transaction is locked under the condition that the coins can be spent by the vehicle which forwards the message bundle to the destination roadside point. Therefore, if the vehicle faithfully transfers the message bundle and

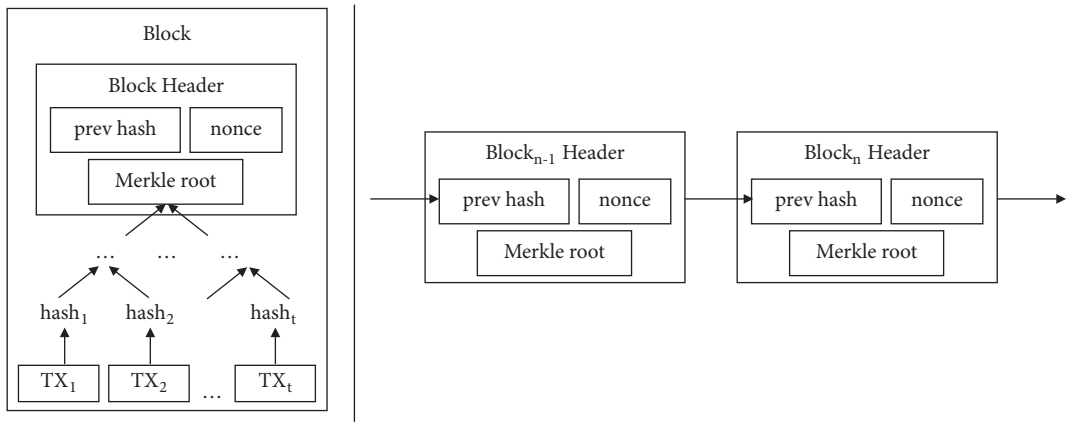


Figure 3: Blockchain structure in the Bitcoin system.

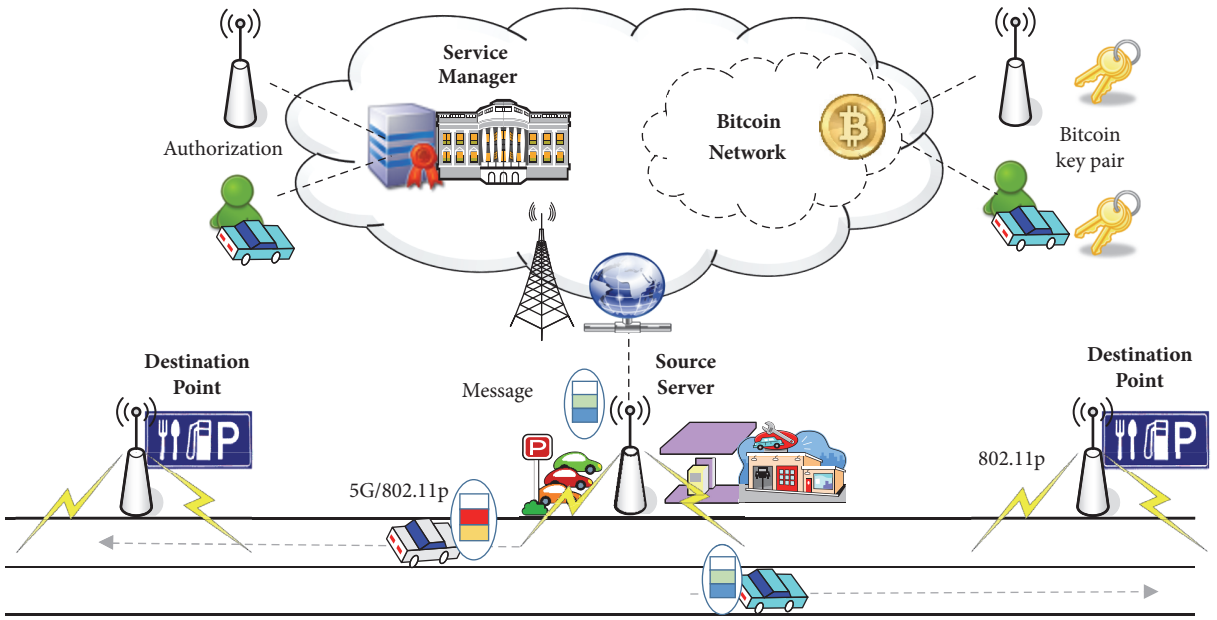


Figure 4: System architecture for Bitcoin-based incentives for VDTNs.

receives a confirmation from the destination point, the vehicle can spend the coins.

For such a scenario, in this paper, we focus on how to design Bitcoin-based secure incentive scheme for VDTNs by taking the following security goals into account.

(i) *Fairness*: actual incentives must be rewarded to the volunteer vehicles only if the vehicle completes to transfer a message bundle to the destination point, which is fair to the source server to prevent a sort of dine and dash.

(ii) *Authorization*: in cyberphysical vehicular networks, only the entities authorized by SM must be able to take part in the vehicular communications for preventing illegal entities from misusing the system.

(iii) *Anonymity of vehicles*: the identity of the vehicle participating in store-carry-forwarding should be hidden during vehicular communications and incentive protocol run for privacy preservation.

The proposed system achieves the security goals by leveraging the functionalities of Bitcoin transactions and adopts certified Bitcoin public key technique to authorize vehicles in the system. Table 2 describes the notations used in the proposed system.

3.2. Certified Bitcoin Public Key. For providing Bitcoin-based incentives, we assume that each roadside server RS_i and each vehicle v_i participated in message dissemination service on VDTNs has certified Bitcoin public key [18] issued by the SM. That is, RS_i and v_i have their Bitcoin private and public key pair $\langle bsk_{S_i}, bpk_{S_i} \rangle$ and $\langle bsk_{v_i}, bpk_{v_i} \rangle$, respectively. More specifically, let $\langle p, q, \mathbb{G}, P \in \mathbb{G} \rangle$ be the public parameters where p and q are two large prime numbers, $\mathbb{G}$ is an additive group with the order q consisting of all points on an elliptic

Table 2: Notations and descriptions.

notation	description
$\langle x, X\rangle$	master secret and public key pair of SM
$\langle bsk_i, bpk_i\rangle$	private and public key pair of entity i
C_i	certified public key of entity i
TX	Bitcoin transaction
$TX.in$	input of the transaction TX
$TX.out$	output of the transaction TX
$Sig\,(sk, M)$	signature for a message M under a private key sk
$Vrf\,(pk, \sigma)$	verification for a signature σ under a public key pk
$\rho : \mathbb{G} \to Z_q$	a mapping function
ts	timestamp

curve $E(\mathbb{F}_p)$, and P is a base point of $\mathbb{G}$. We can choose the parameters in accordance with secp256k1 ECDSA curve specification used in Bitcoin.

Let $\langle x, X\rangle$ be the secret and public key pair of SM where $x \in Z_q$ and $X = x \cdot P$. Certified public keys for each entity U_i authorized by SM are generated as follows:

(1) U_i chooses a random $k_i \in Z_q$ and computes $K_i = k_i \cdot P$, then requests its certified public key by sending K_i to SM.

(2) Assuming that the legitimacy of U_i was validated, SM chooses a random $r_i \in Z_q$, computes $C_i = K_i + r_i \cdot P$ and $y_i' = r_i + \rho(C_i) \cdot x \pmod q$ where ρ is a function encoding an element of $\mathbb{G}$ as a positive integer. Then SM provides $\langle C_i, y'\rangle$ to U_i.

(3) U_i computes $y_i = y_i' + k_i \pmod q$, $Y_i = y_i \cdot P$ and checks if $Y_i \stackrel{?}{=} C_i + \rho(C_i) \cdot X$. If it holds, U_i sets $\langle bsk_i \leftarrow y_i, bpk_i \leftarrow Y_i\rangle$ as its ECDSA key pair and C_i as certified public key.

When U_i's certified public key C_i is given, we can derive the public key Y_i from C_i by using SM's public key X as $Y_i = C_i + \rho(C_i) \cdot X$ so as to verify the signature generated under y_i. In [18], the certified public key itself is encoded in the Bitcoin transaction output in the form of scriptPubKey script to designate the recipient. On the other hand, in our proposed system, the certified public key C_i is only used in authentication between a vehicle and a raodside server, and then the public key Y_i derived from C_i is encoded in the Bitcoin transaction rewarded as incentives.

3.3. Bitcoin-Based Incentive. In this section, we describe Bitcoin-based incentive scheme to reward a vehicle for the effort of message forwarding. Suppose that a source roadside server RS_s wants to send a message M to a destination point RS_d by means of store-carry-forward with the help of a vehicle v_i. When RS_s requests v_i to deliver a message to the destination point RS_d, the incentive transaction TX of RS_s is published to the Bitcoin network under the condition that the output of TX can be redeemed by v_i if v_i completes the message forwarding to RS_d by using MultiSig transaction. However, if v_i does not forward the message to the destination, RS_s is likely to lose its coins without taking advantage of message delivery service from v_i because the RS_s's input of TX is treated as spent in the Bitcoin system once TX is published. To cope with this situation, we put time-locked condition together with MultiSig so as for RS_s to withdraw the coins from TX if v_i does not forward the message nor redeem the output before the time-lock expires.

The details of the proposed scheme are described in the following:

(1) A source roadside server RS_s broadcasts a request message including the identity of the destination point RS_d and the location information loc_d to ask for a volunteer vehicle which will help carrying a message to RS_d.

(2) A vehicle v_i, which will pass by RS_d's location and be willing to help message forwarding, responds to RS_s by giving $\sigma = Sig(bsk_{v_i}, RS_d \| loc_d)$ with its certified public key C_{v_i}.

(3) RS_s verifies the signature σ as $Vrf(bpk_{v_i}, \sigma)$ by deriving v_i's public key bpk_{v_i} from C_{v_i} as described in Section 3.2. If the signature is valid, RS_s prepares a Bitcoin transaction TX_1 and composes a message bundle $msg_1 := \{M \| ts \| Sig(bsk_{RS_s}, M \| ts), C_{RS_s}, TX_1\}$. Figure 5 shows the transactions for transferring Bitcoin as incentives. $TX_1.in$ specifies unspent coins retrieved from RS_s's UTXO pool and it includes RS_s's signature for the transaction. The amount of coins given to v_i as incentives is recorded in $TX_1.out$ and the redemption condition for $TX_1.out$ is written by using locking script consisting of 2-of-2 MultiSig for v_i and time-lock constraint for RS_d. Implementation details of the locking and unlocking scripts are presented in the next sections. RS_s publishes the TX_1 to the Bitcoin network and provides msg_1 to v_i. Note, at this phase, that TX_1 does not mean a prompt coin transfer but functions as a deposit which will be spent by someone who satisfies the unlocking condition.

(4) Upon receiving msg_1, v_i derives RS_s's public key from C_{RS_s} and verifies RS_s's signature. Then, v_i stores and carries the message bundle to the destination point RS_d. In addition, v_i checks the validity of TX_1 through the Bitcoin network and partially prepares a transaction TX_2 to redeem the coins specified in $TX_1.out$ while moving to the destination. At this phase, v_i fills in all other forms of TX_2 except MultiSig unlocking script for $TX_2.in$.

(5) If v_i is an honest volunteer vehicle, v_i will faithfully store, carry, and forward the message bundle. Hence, when v_i arrives at the destination location and recognizes RS_d, v_i composes the message $msg_2 := \{M \| ts \| Sig(bsk_{RS_s}, M \| ts), C_{RS_s}, TX_2\}$ and sends to RS_d.

(6) RS_d parses the msg_2 and verifies the signature of $Sig(bsk_{RS_s}, M \| ts)$ by using C_{RS_s}, then accepts the message M if the signature is valid. Then, as a witness to the effort of message delivery of v_i, RS_d generates a partial signature for TX_2 to unlock 2-of-2 MultiSig

```
scriptPubKey:   OP_IF
                    2 < bpk_vi >< bpk_RSd > 2 OP_CHECKMULTISIG
                OP_ELSE
                    < time-lock > OP_CHECKSEQUENCEVERIFY OP_DROP
                    < bpk_RSs > OP_CHECKSIGVERIFY
                OP_ENDIF
```

Algorithm 1: Locking script of $TX_1.out$ for giving incentives.

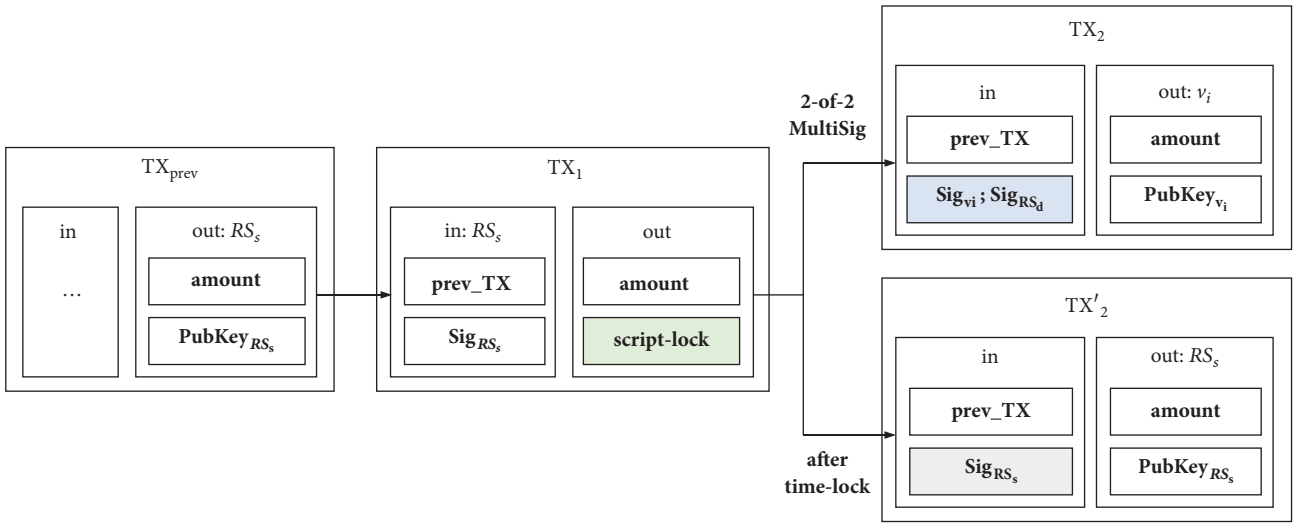


Figure 5: Transactions to give incentive or withdraw.

script which is needed for v_i to spend the coins specified in the previous transaction $TX_1.out$. RS_d provides $\{TX_2, Sig(bsk_{RS_d}, TX_2), C_{RS_d}\}$ to v_i.

(7) v_i derives RS_d's public key from C_{RS_d} and verifies the signature. If it holds, v_i completes 2-of-2 MultiSig unlocking script by adding v_i signature for TX_2 and finally publish TX_2 to the Bitcoin network to transfer the incentive given by TX_1 to v_i's another Bitcoin account.

If all the above steps are correctly processed, the transactions TX_1 and TX_2 will be validated over the Bitcoin network and successfully appended to the blockchain, then v_i can gain Bitcoin incentives as a reward for its contribution to message delivery on VDTNs. In other words, v_i will not be rewarded if it ceases from forwarding the message even though TX_1 is published to the Bitcoin network in step 3 because v_i alone cannot fulfill 2-of-2 MultiSig locking script. Therefore, when RS_s finds that TX_1 is not redeemed by v_i after the time-lock expired, RS_s withdraws the coins by publishing the transaction TX_2' as regarding that v_i did not forward the message faithfully.

3.4. Implementing Transaction Scripts. Validating a Bitcoin transaction relies on two types of scripts, a locking script and an unlocking script. For guaranteeing the fairness to the source server RS_s in our incentive scheme, we make use of MultiSig script and time-lock script in the Bitcoin transactions. Hence, when the source server RS_s publishes the transaction TX_1, RS_s can specify the proper recipient for $TX_1.out$ (i.e., v_i for incentive or RS_s for withdraw) by using MultiSig and time-lock script as shown in Algorithm 1. In our implementation, we consider relative time-lock which implies that $TX_1.out$ can be spent after the specified time has elapsed starting from the TX_1 publishing time. For example, RS_s can specify one day amount of time in $< time - lock >$ if RS_s allows for v_i to deliver the message within a day so that RS_s does not withdraw the coins for the day.

Once RS_s published the TX_1 to the Bitcoin network for providing incentives to v_i, the redemption condition for v_i to spend the coins of $TX_1.out$ is constrained under 2-of-2 MultiSig script fulfilled by both v_i's and destination point RS_d's signatures. Hence, if v_i correctly delivers the message requested by RS_s to the destination point RS_d and receives a partial signature for TX_2 from RS_d, v_i can spend the coins of TX_1 by completing the unlocking script of $TX_2.in$ as shown in Algorithm 2.

The Bitcoin transaction script is a stack-based execution language which uses a stack data structure to store input parameters and a return value of each operation. To execute an operation, the arguments for the operation are first pushed onto a stack and its calculation is performed by reading these arguments directly from the stack. The unlocking script contained in the output and the locking script in the

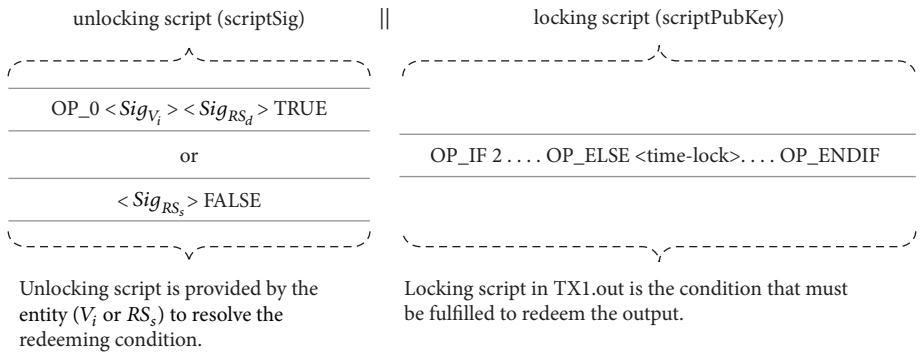


Figure 6: Combining unlocking script and locking script to evaluate a transaction script.

scriptSig:	OP_0 < Sig_{vi} > < Sig_{RS_d} > TRUE

Algorithm 2: Unlocking script in $TX_2.in$ for v_i to redeem the incentive.

scriptSig:	< Sig_{RS_s} > FALSE

Algorithm 3: Unlocking script in $TX_2'.in$ for RS_s to withdraw the coins.

referenced input are combined as shown in Figure 6, and the combined script is executed from left to right in sequence by every Bitcoin validating node.

By combining those scripts of $TX_2.in$ and $TX_1.out$, the execution path of the first IF clause is selected by putting TRUE, then it would be evaluated in the script execution stack as the following form:

$$\texttt{OP\_0} < Sig_{v_i} > < Sig_{RS_d} > 2 < bpk_{v_i} > < bpk_{RS_d} > \texttt{2OP\_CHECKMULTISIG}$$

Figure 7 shows the stack-based script execution to validate v_i's redemption condition by using MultiSig operation.

On the other hand, in order to withdraw the coins of TX_1 after the time-lock expired, RS_s publishes the transaction TX_2' containing the unlocking script as shown in Algorithm 3.

Like the preceding, by putting FALSE at the end of unlocking script, the execution path of ELSE clause is selected, but this execution is only used after the amount of $< time - lock >$ has elapsed from the creation of TX_1. If so, RS_d's script would be evaluated in the script execution stack as the following form:

$$< Sig_{RS_s} > < bpk_{RS_s} > \texttt{OP\_CHECKSIGVERIFY}$$

Figure 8 shows the stack-based script execution to validate RS_s's redemption condition by using time-lock restriction.

4. Discussion

4.1. Security of the Proposed System. As presented so far, our incentive scheme for VDTNs is designed by making use of Bitcoin system which is a cryptographically secure and practical decentralized virtual currency system. In this section, we discuss the security properties of the proposed system in terms of fairness, authorization, and anonymity of vehicular communications.

4.1.1. Fairness. When we design an incentive scheme based on virtual currency for VDTN environments in this paper, one of the important issues is fairness to the source server because a malicious vehicle might not follow the protocol run if the source server provides incentives first.

In the proposed system, providing incentives to a vehicle contributed to message forwarding is processed by the Bitcoin transaction TX_1 which conceptually transfers coins from the source server RS_s's Bitcoin account ($TX_1.in$) to the forwarding vehicle v_i's account ($TX_1.out$). Since the $TX_1.out$ for v_i is locked by 2-of-2 MultiSig script when RS_s publishes TX_1 to the Bitcoin network, the coin amount specified in $TX_1.out$ is ineffective for v_i to redeem it by $TX_2.in$ at this moment unless the destination point RS_d confirms the message receiving by giving its signature for TX_2 to unlock 2-of-2 MultiSig combined with v_i's signature. v_i cannot but complete message forwarding to RS_d in order to redeem the incentive. Therefore, the source server RS_s does not need to worry about dine and dash of v_i, and v_i also can be rewarded for its contribution to message delivery if v_i honestly follows the protocol run.

Moreover, as considering the problematic situation where v_i ceases delivering the message to the destination once RS_s published TX_1 incentive transaction, RS_s is allowed to make the transaction TX_2' to withdraw the coins from TX_1 by putting time-locked script. When RS_s finds that TX_1 is not redeemed by v_i after the time-lock expired, it is regarded that v_i did not follow the protocol and could not redeem the incentive, so RS_s withdraws the coins. Such time-lock condition also provides another feature that RS_s cannot withdraw the coins earlier than the time-lock. As an example for 12-hour time-lock, v_i can acquire the coins of TX_1 if v_i

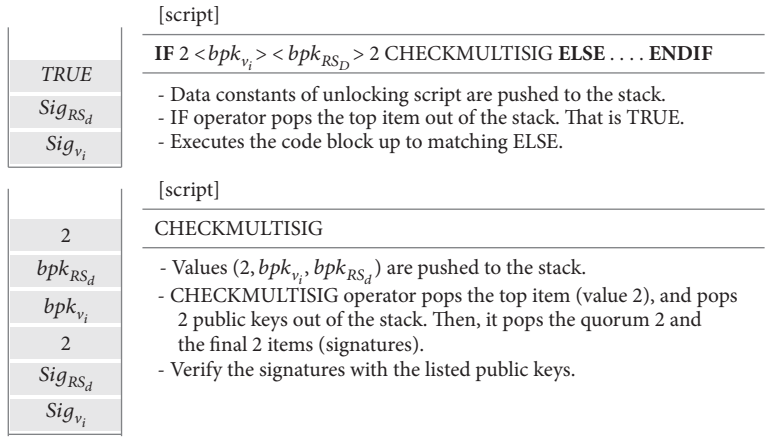


Figure 7: Script execution for validating v_i's redemption condition.

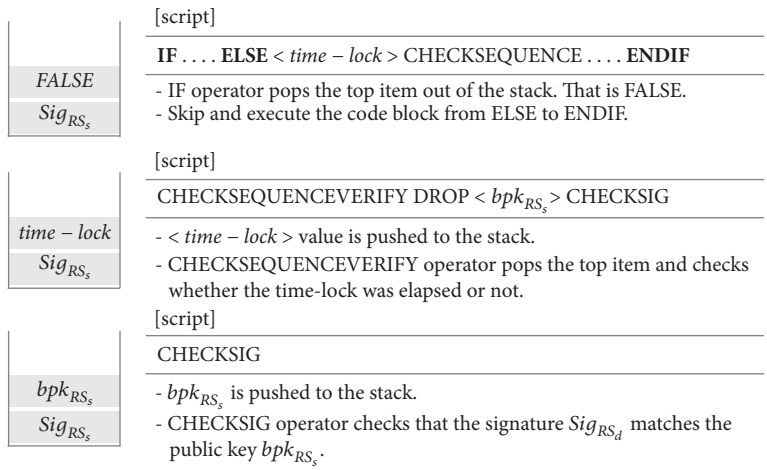


Figure 8: Script execution for validating RS_s's redemption condition.

arrives at the destination point within 12 hours while RS_s cannot withdraw the coins, which guarantees kind of fairness to the vehicle.

4.1.2. Authorization. To deploy a practical VDTNs application of good quality of service in the real-world scenarios, it is needed to allow only authenticated users to take part in the system. We consider using Bitcoin public key cryptography based on ECDSA for our VDTN scenario instead of adopting vehicular-PKI for authenticated vehicular communications. However, ordinary Bitcoin public keys are not sufficient to incorporate trustworthiness from real-world entities into the system. Hence, in the proposed system, each vehicle v and each roadside unit RS are authenticated by using certified Bitcoin public keys (C_v and C_{RS}, respectively) issued by SM based on the technique of [18].

When a vehicle and a roadside unit communicate to forward a message and give incentives, they exchange their certified public keys, then the corresponding Bitcoin public keys are derived from SM's public key. Because the exchanged message and Bitcoin transactions include signatures verified under the derived public keys, any other entities unauthorized by SM cannot join the system.

4.1.3. Anonymity of Vehicles. For secure vehicular communications, another security aspect is anonymity of vehicles which voluntarily take part in message store-carry-forwarding communications on VDTNs. The only thing required to the vehicles in the system is their valid Bitcoin public key to uniquely identify the vehicle and handle Bitcoin transactions for incentives. Those public keys used in vehicle to roadside unit communications can be viewed as vehicle's pseudonyms and do not contain any identity information of the vehicle even though the proposed scheme employs certified public keys.

When we say privacy preservation, we should consider not only anonymity of user identity but also unlinkability. However, the proposed system does not fully satisfy unlinkability requirement because we just assumed that each vehicle has a single public key, so that any outside observer can decide any two messages or two transactions originated from the same user by tracing the fixed same public key of the user.

Table 3: Comparison of the characteristics of the proposed system with the previous systems.

	Lee et al. [3]	Zhu et al. [15]	Lu et al. [16]	proposed
network service	simple broadcast	dtn	dtn	vdtn
trust model	centralized authority	centralized virtual bank (VB)	centralized (VB)	distributed (blockchain)
fault tolerance	weak	weak	weak	strong
fairness	escrow held by the authority	escrow held by the VB	escrow held by the VB	multisig and timelock script on a blockchain
incentive means	virtual coin, application dependent	virtual coin, application dependent	reputation, virtual coin, application dependent	Bitcoin, worldwide
payment contract	signed receipt by a receiver	layered credits by sender, forwarder, receiver in sequence	layered credits	Bitcoin transaction script
rewarding	exchange receipt to virtual cash through the authority	credit clearance through the VB	credit clearance through the VB	Bitcoin payment
identity	certificate	certificate	certificate	Bitcoin public key (address)
anonymity	low	n/a	n/a	medium

One simple solution to this challenge is for a vehicle to generate multiple Bitcoin public keys and use a different public key whenever a message forwarding and incentive protocol is performed on VDTNs. However, in this case, it is required to securely maintain lots of private keys as many as the number of public keys. Another flexible solution is to use a one-time public key technique such as CryptoNote in which an ephemeral key pair for each protocol run can be computed from a fixed long-term key pair[29]. Hence, the proposed scheme can be modified by using one-time public key technique to enhance the anonymity of vehicular communications and incentive transactions.

4.2. Comparison. To highlight the novelty of the proposed Bitcoin-based incentive system for VTDNs, we evaluate and qualitatively compare our system with previous systems. Table 3 summarizes the different characteristics of the proposed system as compared with Lee et al.'s [3], Zhu et al.'s [15], and Lu et al.'s [16], respectively.

The key difference of the proposed system results from the use of Bitcoin which is a decentralized cryptocurrency and a worldwide payment system whose transactions are verified by means of a blockchain, while each previous system implements its own application-dependent virtual coin relying on a centralized trusted authority or a bank to guarantee the validity of payment transactions. Hence, for the previous system, we cannot help but depend on the central authority to enjoy reliable payment service. However, it is widely believed that the distributed structure of blockchain network performs better robustness under the single point of failure, so the proposed system can provide strong fault tolerance.

In addition, the previous systems make use of public key certificate to identify the entities participating in the network service and to verify the layered credits, but they do not focus on the anonymity of users. On the other hand, in our system, the Bitcoin public key used in a payment contract as the form of Bitcoin transaction script can be viewed as a pseudonym and we can generate multiple keys or adopt one-time public key technique to enhance the anonymity to some degree.

We also compare the incentive procedures of the proposed system to reward a node for message forwarding and briefly shows the processes assigned to each entity relating to incentive scheme in Table 4. In the previous systems of [15, 16], the incentive for message forwarding is endorsed by means of the layered credits which are sequentially generated by message sender, forwarder, and receiver during the message delivery phase. To securely handle incentive scheme, forwarder and receiver must check the validity of the given credit by themselves and add their own credit layers in sequence. Then, the VB verifies the collected credits and records amount of virtual coin in forwarder's account if the credits are valid.

On the other hand, in our system, the incentive is handled by means of Bitcoin transactions to pay the coin from the sender to the forwarder. Those Bitcoin transactions are validated by Bitcoin network in a distributed manner and added to a blockchain which serves as immutable distributed ledgers. The message forwarder just queries the validity of sender's payment transaction to Bitcoin network, instead of verifying sender's payment transaction by forwarder itself. Message sender can control that the payment would be redeemed by the honest forwarder which delivers the message to the receiver by putting MultiSig locking script to the payment transaction which must be resolved by both forwarder's and receiver's signatures.

As compared to the previous system, the processes of validating incentive transactions to reliably pay the coins from the sender to the forwarder as an incentive are not burdened to VANET but shifted to Bitcoin network. The required processes for the sender and the forwarder are just to publish Bitcoin transactions which will be validated through a blockchain network. Therefore, we can develop a practical credit-based incentive scheme on VANETs at a low

Table 4: Comparison of the incentive procedures to perform virtual coin rewarding.

	VANET			VB / Blockchain
	sender	forwarder	receiver	
Zhu et al. & Lu et al.	(1) generate a base layer credit	(2) check the validity of the base layer credit (3) generate and add its endorsed layer credit	(4) check the validity of the layered credits (5) generate and add its endorsed layer credit (6) send and request clearance of the credits to VB	(7) check the validity of the all layered credits (8) check sender's balance (9) record amount of coin in forwarder's account
proposed	(1) publish incentive transaction to blockchain network (7) publish withdraw transaction to blockchain network	(4) query the validity of the incentive transaction to Bitcoin network (7) publish redemption transaction to blockchain network	(6) sign forwarder's redemption transaction	(2), (8) validate the incoming transaction (5) respond to the transaction query (3), (9) add the transaction to a blockchain

cost by removing the necessary of implementing application-dependent virtual coin system but by taking advantage of the functionalities of the existing cryptocurrency system.

5. Conclusion

In this paper, we proposed a secure incentive scheme incorporating with Bitcoin for VDTNs to stimulate vehicles positively cooperating with other nodes and to reward their efforts. Based on the security features of the Bitcoin system, the incentives for volunteer vehicles are rewarded by means of Bitcoins which can be worldwide used as virtual cash, and the fairness to the source server is guaranteed by using MultiSig transaction so that a message relaying vehicle can redeem the coins of incentive transactions only if the vehicle correctly completes the message relaying to a destination. To achieve the fairness goal, we also implemented transaction scripts to deal with reliable incentive rewarding based on locking and unlocking scripts consisting of 2-of-2 MultiSig and time-lock condition. Especially, as compared to the previous systems, the proposed incentive scheme can be developed at a low cost because we do not need to implement our own virtual currency system on VDTNs.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korea Government (MSIT) (2017-0-00156, The Development of a Secure Framework and Evaluation Method for Blockchain).

References

[1] J. Zhou, X. Dong, Z. Cao, and A. V. Vasilakos, "Secure and privacy preserving protocol for cloud-based vehicular DTNs," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1299–1314, 2015.

[2] J. A. F. F. Dias, J. J. P. C. Rodrigues, and L. Zhou, "Cooperation advances on vehicular communications: a survey," *Vehicular Communications*, vol. 1, no. 1, pp. 22–32, 2014.

[3] S.-B. Lee, J.-S. Park, M. Gerla, and S. Lu, "Secure incentives for commercial ad dissemination in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 6, pp. 2715–2728, 2012.

[4] X. Lin, R. Lu, and X. Shen, "Location-Release Signature for Vehicular Communications," in *Proceedings of the 2009 Proceedings of 18th International Conference on Computer Communications and Networks - ICCCN 2009*, pp. 1–7, San Francisco, CA, USA, August 2009.

[5] M. Raya and J.-P. Hubaux, "Securing vehicular ad hoc networks," *Journal of Computer Security*, vol. 15, no. 1, pp. 39–68, 2007.

[6] X. Lin, X. Sun, P.-H. Ho, and X. Shen, "GSIS: a secure and privacy-preserving protocol for vehicular communications," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 6 I, pp. 3442–3456, 2007.

[7] R. Lu, X. Lin, H. Zhu, P.-H. Ho, and X. Shen, "ECPP: efficient conditional privacy preservation protocol for secure vehicular communications," in *Proceedings of the 27th IEEE Communications Society Conference on Computer Communications (INFOCOM '08)*, pp. 1229–1237, April 2008.

[8] R. Lu, X. Lin, H. Zhu, P.-H. Ho, and X. Shen, "A novel anonymous mutual authentication protocol with provable link-layer location privacy," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 3, pp. 1454–1466, 2009.

[9] Y. Park, C. Sur, C. D. Jung, and K.-H. Rhee, "An efficient anonymous authentication protocol for secure vehicular communications," *Journal of Information Science and Engineering*, vol. 26, no. 3, pp. 785–800, 2010.

[10] Y. Sun, R. Lu, X. Lin, X. Shen, and J. Su, "An efficient pseudonymous authentication scheme with strong privacy preservation for vehicular communications," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 7, pp. 3589–3603, 2010.

[11] Y. Park, C. Sur, S. Shin, K.-H. Rhee, and C. Seo, "A privacy preserving message delivery protocol using identity-hidden index in VDTNs," *Journal of Universal Computer Science*, vol. 19, no. 16, pp. 2385–2403, 2013.

[12] X. Zhu, S. Jiang, L. Wang, and H. Li, "Efficient privacy-preserving authentication for vehicular Ad Hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 2, pp. 907–919, 2014.

[13] Q. Li, A. Malip, K. M. Martin, S.-L. Ng, and J. Zhang, "A reputation-based announcement scheme for VANETs," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 9, pp. 4095–4108, 2012.

[14] J. A. F. F. Dias, J. J. P. C. Rodrigues, L. Shu, and S. Ullah, "Performance evaluation of a cooperative reputation system for vehicular delay-tolerant networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2014, no. 1, pp. 1–13, 2014.

[15] H. Zhu, X. Lin, R. Lu, Y. Fan, and X. Shen, "Smart: a secure multilayer credit-based incentive scheme for delay-tolerant networks," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 8, pp. 4628–4639, 2009.

[16] R. Lu, X. Lin, H. Zhu, X. Shen, and B. Preiss, "Pi: a practical incentive protocol for delay tolerant networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 4, pp. 1483–1493, 2010.

[17] J. Zhou and Z. Cao, "Secure incentive-based architecture for vehicular cloud," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM'12)*, IEEE, Anaheim, USA, December 2012.

[18] G. Ateniese, A. Faonio, B. Magri, and B. de Medeiros, "Certified bitcoins," in *Proceedings of the International Conference on Applied Cryptography and Network Security*, vol. 8479 of *Lecture Notes in Computer Science LNCS*, pp. 80–96, Springer, 2014.

[19] Y. Park, C. Sur, H. Kim, and K.-H. Rhee, "A reliable incentive scheme using bitcoin on cooperative vehicular ad hoc networks," in *Proceedings of the 2017 International Symposium on Mobile Internet Security (MobiSec'17)*, pp. 1–8, Jeju, Republic of Korea, October 2017.

[20] "Bitcoin," https://bitcoin.org.

[21] "Namecoin," https://namecoin.org.

[22] "Litecoin," https://litecoin.org.

[23] "Monero," https://getmonero.org.

[24] "Ethereum," https://ethereum.org.

[25] "Zcash," https://z.cash.

[26] S. Nakamoto, *Bitcoin, A peer-to-peer electronic cash system*, 2008, https://bitcoin.org/bitcoin.pdf.

[27] A. M. Antonopoulous, *Mastering Bitcoin - Programming the open blockchain*, O'Reilly Media, 2017.

[28] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[29] N. van Saberhagen, *Cryptonote v2.0*, 2013, https://cryptonote.org/whitepaper.pdf.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 6135845, 12 pages
https://doi.org/10.1155/2018/6135845

Research Article

Security Evaluation Framework for Military IoT Devices

Sungyong Cha, Seungsoo Baek, Sooyoung Kang, and Seungjoo Kim

Center for Information Security Technologies (CIST), Korea University, Seoul 02841, Republic of Korea

Correspondence should be addressed to Seungjoo Kim; skim71@korea.ac.kr

Received 6 March 2018; Revised 6 May 2018; Accepted 29 May 2018; Published 3 July 2018

Academic Editor: Ilsun You

IoT is gaining importance in our lives and in the military too. With the application of IoT paradigm in the military and the weapon system's connectivity to the network, this facilitates the commanders to make real-time decisions. However, cybersecurity threats to weapon systems intensify along with the growing of IoT's benefits. Coping with these cybersecurity threats nowadays, we require the implementation of "security by design" concept during weapon system development throughout the system lifecycle, but not traditional security solutions. Since only developed countries are capable of developing systems on their own, they adopt "security by design" when developing new weapon systems; another approach to acquire weapon systems is through import if a country cannot develop the whole weapon system. However, few studies have been done on the security evaluation framework that could be used upon purchase and integration of the developed weapon system. In this paper, we proposed a novel security evaluation framework that could be used to integrate IoT devices and components into the weapon system and a method to address cybersecurity requirements using international standard security control.

1. Introduction

With the advent of the IoT paradigm, all devices connected to the network began exchanging data among each other. The military is aware of the benefits of IoT and has continuously applied it to the weapon system, which ultimately becomes an essential to achieve military goals [1]. IoT technology connects and integrates into a myriad of devices and systems, but this will also increase attack surface. Until recently, cyber-attacks have been spreading beyond PCs to devices which of those have been connected to the Internet and smartphones [2]. But, the security threats of the network-centric weapon systems, including the IoT devices, are growing together with the advantages of IoT too [3]. What is more, the military domain is targeted by hackers. There is a recent experiment conducted in 2015 purported to hack into a Jeep Cherokee, the symbol of military mobility in the U.S., and it turned out to be a successful hacking later [4]. In fact, even if the network is far from another network physically, such as the Internet and Intranet, hacker attacks are still inexorable. According to [5], there is a possibility that the Trident system in partitioned network will be hacked through the air-gap from an external network, which will seriously affect operation and reliability. Therefore, some argue that the protection against cyber-attacks should be placed as the top priority. Simply, the usual way to enhance cybersecurity of the existing systems by introducing security solutions, e.g., firewall and cryptographic devices, is no longer effective. If a security vulnerability is found in a system, denoting that the cybersecurity factor has not been embedded in the system design at first, it would be difficult and costly to fix it; in some extreme cases, it would be impossible to handle the problem. In order to overcome these defects, researches are conducted by implementing 'security by design' concept as displayed in [6–8]. Representative examples are HACMS (High-Assurance Cyber Military Systems) project in DARPA (Defense Advanced Research Projects Agency) and the Defense Acquisition System in Department of Defense (DoD) which coordinately takes cybersecurity into account during weapon system acquisition [9, 10].

However, [9, 10] only confine the consideration for cybersecurity to weapon systems which are developed by the U.S.; in this manner when one country imports part or all of a weapon system from another country, its cybersecurity could not be checked. To remedy these drawbacks, a composite security evaluation method for assessing the cybersecurity

of the whole system in system integration has been studied [11, 12, 15]. Such composite security evaluation method is, yet, difficult to be applied to weapon systems because of the differences in the evaluation target and assurance levels for these composite security assessments. Therefore, according to [13], if the composite security evaluation cannot be implemented and the security of the system cannot be proved mathematically, the system should be developed along a well-structured process; hence we require a framework to evaluate cybersecurity when we buy parts from other countries and integrate them. To settle cybersecurity requirements, acquirer makes and utilizes security controls. For example, there are 253 security controls composed of 18 families being used in the United States and 5 families and 76 security controls utilized in Republic of Korea, respectively [16]. Nevertheless, security controls differ from country to country in terms of the criteria and description; on top of that, there are differences in interpretation between acquirer and provider, which may result in missing some security functions of the weapon system; hence, a universal security control is imperative.

In this paper, we propose a novel evaluation framework that could be used to evaluate cybersecurity requirements, not upon weapon system development, but the import of part or all of a weapon system from other countries. By using the international cybersecurity standards, security controls, as well as the understanding between acquirer and provider, are then unified in order to achieve mutual trust and strengthen cybersecurity along the arms import process.

The contents of this paper are as constructed as follows: we discuss the research about the background of this study in Section 2; in Section 3, we introduce the framework to improve the cybersecurity when purchasing weapon systems, in which our proposal also maps out domestic security controls with international cybersecurity standards. Following that, we examine how the proposed process can be applied in Section 4; we finalize the paper in Section 5.

2. Related Work

2.1. Cybersecurity Evaluation in Domestic and International. Reliability and security are verified through systems' cybersecurity evaluation. That is, system users can make use of evaluation certification methods to upgrade information protection level of an organization and their asset and contribute to credibility. Countries establish their standards to undergo cybersecurity evaluation and requirements based on their own circumstances. To name a few, BS 7799 part 2 [17] operated by United Kingdom and TCSEC (Trusted Computer System Evaluation Criteria) in the U.S. [18] are the representatives of domestic evaluation standards. NIST (National Institute of Standards and Technology) documents are also references to many countries, besides the U.S. government [19].

ISO / IEC 15408 [20], as known as the Common Criteria (CC), is an international standard established to coordinate different information protection system evaluation standards by country and to mutually certify the evaluation results. CC consists of 11 Security Functional Requirements (SFRs) which defines the required structure and content of security functional components; and 10 Security Assurance Requirements (SARs) define a scale for measuring assurance for component target of evaluations. However, the limited ability to evaluate a single product but not the whole system and disability to provide an evaluation of environments elements (physical, human, and operational security) discern the disadvantage of CC; to this end, the composite evaluation was introduced. Yet, composite ToE (Target of Evaluation) of CC-CAP (Composite Assurance Packages) is subjected to products only with CC certification. The highest evaluation level of CC-CAP, CAP-C, manifests a similar evaluation level of EAL (Evaluation Assurance Level) 4 in CC to which weapon systems requesting a level higher than EAL 5 are not applicable. CCDB-CPE (Common Criteria Development Board-Composite Product Evaluation) approach could be deployed to weapon systems demanding a level higher than EAL 5; still this approach is yet to be claimed ideal; design documents and other sensitive information are required to be disclosed; added to this disadvantage, it is not suitable for implementation in weapon systems as it was originally designed for smartcard evaluation. EURO-MILS (multiple independent levels of security), another evaluation approach appropriate for those security levels equivalent to EAL 5, is also limited to the use of designated platforms only. Table 1 concludes the characteristics of these composite security evaluation methods.

ISO/IEC 27001 [21], as known as the Information Security Management System (ISMS), consists of 114 controls in a total of 14 clauses. It also carries out verification for information security policy management, human resources security, physical environment security, communication and operation management, information system construction, and maintenance. The security controls of ISO/IEC 27001 could be indicators of whether the product (or information) is managed properly based on the established procedure. There is a shortcoming that the product security itself, still, could not been substantiated. Therefore, it is necessary to verify that the security of a product in general. Since the SFRs of ISO/IEC 15408 is initiated to confirm only the implemented security functions of the product but not the environment related to the development, the evaluation of functional and nonfunctional parts of the weapon system could be done by complementing ISO/IEC27001 mentioned previously.

2.2. Cybersecurity Evaluation in Defense. The evaluation of security functions and assurance of a to-be-imported system is of considerable importance. Despite that, to achieve high level of security and assurance of weapon systems as well as security functional evaluation, introducing formal verification is necessary. Nevertheless, a formal verification entails the combination of all the functions in the system to be proved mathematically to validate the logic. A framework should be followed by a well-defined development process if it cannot be proved on a mathematical basis [13]. Products are developed according to the System Development Life Cycle (SDLC); similarly, weapon systems are developed based on SDLC, considering whether they are suitable to the acquisition system of the respective country. However, these acquisition systems have traditionally focused on performance

Table 1: Composite security evaluation approach.

	CC-CAP [11]	CCDB-CPE [12]	EURO-MILS [13]
Operation area	CCRA Members (28 countries)	CCRA Members (28 countries)	European countries
key features	a composite Target of Evaluation (TOE) is composed of components that have already passed CC evaluation	enable installing applications onto an already passed CC evaluation platform	reuse component certificates
Evaluation subject	information product	smartcard	avionic and automotive
Assurance Level	CAP-(A, B, C) ∗ CAP-C is comparable to EAL4	EAL1 ~ EAL7	comparable to EAL 5

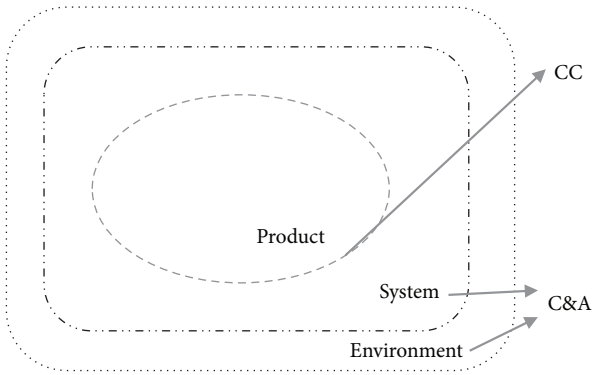


Figure 1: Scope of security evaluation by CC and C&A.

and operability rather than cybersecurity aspects of weapon systems. Instead of changing the acquisition system to reinforce cybersecurity, processes such as C&A (Certification and Accreditation) are blended into the existing systems to enhance cybersecurity level, of which this practice is shown in Figure 1.

The C&A process was introduced in the Trusted Computer System Evaluation Criteria (TCSEC) [18], known as the Orange Book, and was then recognized as an international standard under the name of Common Criteria (CC) [20]. However, as described in Section 2.1, there are drawbacks that make it difficult to apply these evaluation methods directly to weapon systems. Thus, the U.S. Department of Defense (DoD) created the Department of Defense Information Technology Security Certification and Accreditation Process (DITSCAP) in 1997 for environmental and system security evaluation as well as product security [23]. Later on, the DoD released Information Assurance Accreditation Process (DIACAP) [24] to overcome the drawbacks of DITSCAP and is currently operating Risk Management Framework (RMF) [25, 26]. At present, cybersecurity-enhanced defense acquisition system with RMF employed in the U.S. is shown in Figure 2 [22]. However, [22] focuses only on weapons systems development and, hence, is difficult to use when purchasing and integrating parts of weapons systems. It is not easy to apply the above-mentioned system in other countries because most countries would not develop weapons systems by themselves but import weapons systems from other countries. For example, some countries purchase critical parts, e.g., stealth functions, in developed countries; in essence, it is problematic to verify security by using different cyber security processes and standards for acquirers and providers. Besides, the provider developed the system of which its risk management was based on the assumption of an environment totally different from that of the acquirer. Therefore, it is not appropriate for the acquirer to assess risk based on the same environment in this respect. Therefore, we need a framework to design, verify, and test cyber security in the weapon system import process. Figure 3 is a general weapon system import process where we must combine some of the frameworks to enhance cybersecurity.

An analysis of the defense acquisition system shows that the Risk Management Framework and Cybersecurity Test & Evaluation are carried out throughout the lifecycle of weapon system development. In addition, it applies not only to risk management in functional part of the weapon system, but also to the nonfunctional aspects, such as the development environment and human resources. Therefore, it is essential to apply risk management in both functional and nonfunctional parts of the weapon system during the purchasing process shown in Figure 3.

2.3. Weaknesses in Current Evaluation Framework. To summarize the issues discussed earlier, first of all, among the existing import process for weapons systems, there is no framework to enhance cybersecurity. Secondly, there is a possibility of misinterpretation of weapon security requirements that arises from differences in security controls adopted in various countries. For that reason, the requirements for the weapon system import process are summarized as follows.

(i) To comprehensively evaluate both security and nonsecurity functions, processes for enhancing cybersecurity such as RMF and Cybersecurity T&E should be integrated into existing import processes.

(ii) Security controls of the provider and the acquirer should be matched so that they can be used in all instances, not limited to specific countries.

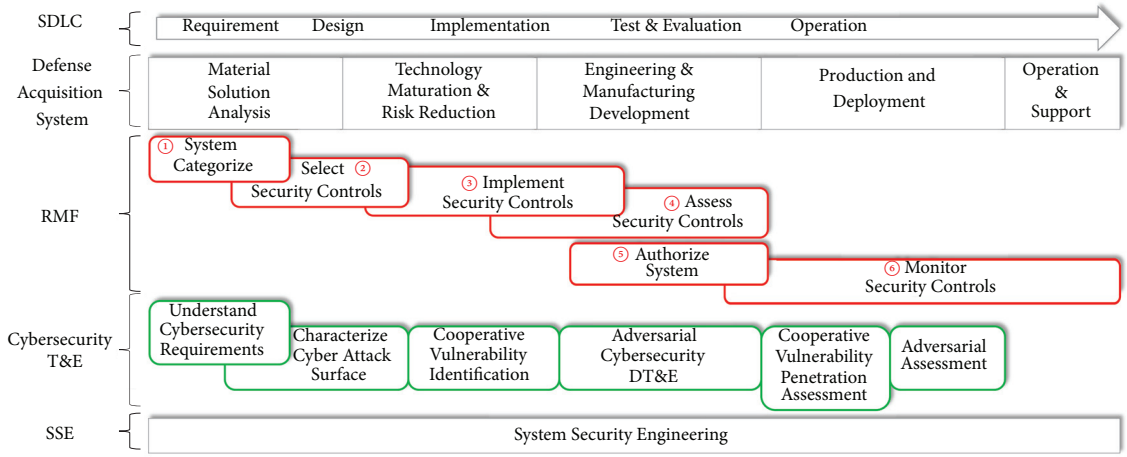


Figure 2: Defense Acquisition System in the U.S. for enhancing cybersecurity [22].

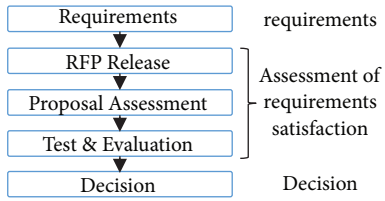


Figure 3: General weapon system purchasing process.

3. Proposed Security Evaluation Framework

3.1. Security Evaluation Framework. In this chapter, we propose a new security evaluation framework that combines RMF and Cybersecurity Test & Evaluation in the existing weapons import process with cybersecurity international standards. The framework for improving cybersecurity is shown in Figure 4: ① ~ ⑥ relate to each RMF step in Figure 2 and the blue boxes indicate the purchase process in Figure 3; the red and green boxes represent RMF and Cybersecurity T&E in Figure 2, respectively. Also, the indicators of italicized alphabets for flow of steps are illustrated in detail below.

(A) Defining Requirements with Risk Identification (System Categorization). At this stage, the same method as the first step in RMF [25, 26] is used to identify the risks for the product (or system) to be acquired. And the provisional impact value for each risk is calculated. Provisional impact values are classified into three levels: high, moderate, low for confidentiality, integrity, and availability. By defining the cybersecurity requirements for the product to be acquired with reference to the impact value of the identified risk, the unnecessary cost can be reduced and accurate security functions can be implemented. These cybersecurity requirements are then integrated with other functions, operating requirements and then passed on to the next step.

(B) Selecting Appropriate Security Controls from the Requirements. In this step, we choose security controls as a countermeasure to achieve the cybersecurity requirements defined in step (A). To select security controls for the product to be imported, we should review the security controls in the parent systems first and determine if there are any security controls that could be inherit from the system. Those are inherent in the system should be tailored out from the product list of security controls and the remaining list would be the security controls we should implement. At this point, the security control set made by the acquirer comes into effect. The reason is to reduce time to select security controls and to remove missing parts by using new sets of security controls that are not familiar with. If there is no set of security controls available from the acquirer, they may skip this step and the provider may bring up a set of security controls or a set of international standard security controls. Once the security controls have been selected, one can proceed to the next step following by review.

(C) Converting the Security Controls into the International Standards. The selected security controls are converted into international standard (ISO/IEC 15408, ISO/IEC 27001) security controls at this stage. The reason for this standardization is that it is difficult for the provider to clearly understand the security controls of the acquirer, so the misunderstanding ensued from such environmental differences can be reduced. However, the problem is to what extent the set of security controls of the acquirer and the provider can be converted and expressed under the set of international standard security controls. This will be covered in detail in Section 3.2, and briefly, the mapping table and definition of addition subjects are provided in Appendix H [16].

Concurrently, the part corresponding to the function of product security is converted into the Security Functional Requirements (SFR) of ISO/IEC 15408, and that of operation and environment is converted into ISO / IEC 27001. Nonetheless, there are real cases where ISO/IEC 15408 does not cover

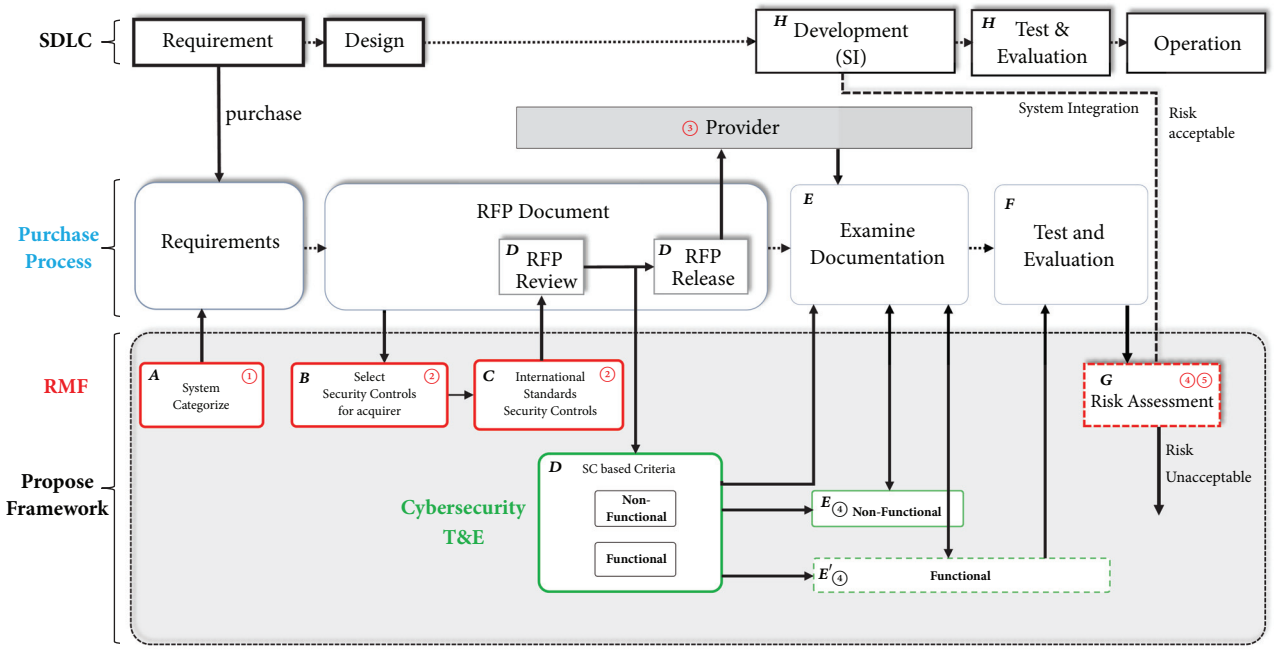


Figure 4: Proposed security evaluation framework.

all the conversions for product security function, then ISO / IEC 27001 is used for the conversion in such cases. Using international standard security controls in this construct, we can generate a set of security requirements similar to the Profile Protection used in the Common Criteria.

(D) Completion of RFP Evaluation Criteria and Preparation for Proposal Evaluation. It is a step for a comprehensive assessment of those security controls, functions, and operating requirements that has been converted into international standards which reflect cyber security requirements. The main objective at this stage is to gather all stakeholders and coordinate any conflicting interests in between, in particular, to adjust requirements that may be confined to funding. If resource constraints and other issues do not reflect all requirements, they should be broken down into mandatory requirements and optional requirements on the foundation of distinguished risk and impact values identified in the first step [27]. In other words, risk-based decisions are made so that risks can be handled according to priorities. Once everything is done consistently, it is followed by the announcement of RFP (Request for Proposal).

While the provider is in the middle of RFP release and proposal preparation, the acquirer constructs the evaluation list by dividing the nonsecurity functional part and the security functional part based on the cybersecurity requirements in Figure 4. The rationale behind this is to insure the evaluation of security-related means and environments for required security functions and product development.

(E) Evaluation of Submitted Proposals and Preverification. The provider submits the proposal of the announced RFP to the acquirer based on the international standard security controls. Since the security nonfunctional parts are difficult to verify directly, it is advised to submit proposal results which has been verified by an authorized third party. Despite the fact that the nonfunctional parts could be compromised by certifications such as ISO/IEC 27001, it is only possible when the scope of evaluation is identical. The security functional parts are the main components to be verified in the next step "Real verification"; nevertheless, it should also engage in assessing whether the document evaluation comprises the functionality to confirm the satisfactory level of security functions. When examining the satisfactory level of security functions (for the security functional parts of the document evaluation), it is necessary to deploy advance preparation to shorten the evaluation time for "Real verification". Advance preparation refers to the identification of the attack surface and the identification of known vulnerabilities [28] which embody the penetration test, and Figure 5 is a detailed description of E′ in Figure 4.

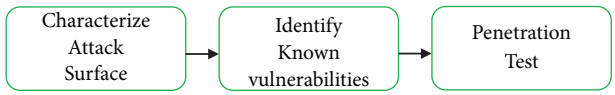


Figure 5: Security functional test flow.

(F) Functional Requirement Verification. Products that passed documentation undergo another test to gauge whether they can function exactly as what have been stated in the proposal. The focus of the verification is not only to validate the

Table 2: Number of security controls that do not provide mapping table in NIST 800-53.

Security Controls		Total	Not Provides
AC	Access Control	25	8
AT	Awareness and Training	5	1
AU	Audit and Accountability	12	4
CA	Security Assessment and Authorization	9	5
CM	Configuration Management	11	2
CP	Contingency Planning	13	1
IA	Identification and Authentication	11	4
IR	Incident Response	10	5
MA	Maintenance	6	3
MP	Media Protection	8	1
PE	Physical and Environmental Protection	20	2
PL	Planning	9	1
PS	Personnel Security	8	1
RA	Risk Assessment	6	1
SA	System and services Acquisition	22	8
SC	System and Communications Protection	44	31
SI	System and Information Integrity	16	11
PM	Program Management	9	5

Figure 6: Risk assessment flow.

described requirements, but also to confirm if cybersecurity is implemented properly by carrying out known vulnerabilities check and penetration test. The actual evaluation is based on a block-box test. However, if the acquirer is in agreement with the provider on white-box testing, the acquirer could verify the product in detail. In case of lack of time and cost, this might be an optional step for acquirers to follow.

(G) Risk Assessment. The product is cleared if it has passed through all the evaluations at "verification" stage; products that fail to meet the requirements criterion are directed to undergo risk assessment. The first risk assessment was a provisional impact value done under a proposed situation, whereas the risk assessment at this step, will be revealed using real data input. The risk assessment flow is shown in Figure 6, and this procedure is applied in accordance with the procedure in [29].

At this stage, an actual and reliable risk assessment is feasible as outcome is collected from real data. If the risk level result is acceptable, acquirer can advance to the next step; if not, the parts would be taken as inappropriate and acquirer may consult with provider.

(H) Secure System Integration and Operational Test and Evaluation. Products that survived "verification" ((F), (G) Step) are incorporated into the whole system and are, finally, reevaluated. After integrating into the system, similar evaluation on functionality of a single product goes through, except that the reevaluation for the time being concentrates on the product operability. The "Assurance" part that is designed to guarantee the function performance of the product is excluded from this paper.

3.2. Mapping Security Controls to ISO/IEC 15408 and 27001. The security controls of acquirer have to be revised to adapt to international standardized security requirements with a purpose to guarantee a successful flow of Stage (C) in Section 3.1. This section illustrates the amended international security controls with the most concrete and detailed NIST 800-53. According to the analyzing result from the conversion table provided in Appendix H, 73 out of 252 security controls in 18 categories are not provided with mapping as security controls of ISO/IEC 15408 and 27001 [Table 2].

Those security controls without mapping are analyzed and converted into international standard security controls of which the results are shown in Table 3.

The conversion of security controls demands considerable of technical controls; in view of this, it is recommended to proceed conversion using ISO/IEC 15408. The definition of the class regarding ISO/IEC 15408 Security Functional Requirements is listed in Table 4. Also, security controls that are not related to weapons purchasing, IR (Incident Response), AT (Awareness and Training), and PM (Program Management) are removed from conversion of security controls.

For example, the *IA-3 (Device Identification and Authentication)* relates to the identification and authentication of devices connected to the information system. This control can be converted into the UAU (User Authentication) and UID

Table 3: Security controls that are converted on component and family level.

NIST	Security Controls	ISO/IEC15408 Common Criteria Component or Family
AC-21	Information Sharing	FPT_TDC.1
AC-22	Publicly Accessible Content	FPT_TDC.1
AC-23	Data Mining Protection	FTA_LSA.1
AU-13	Monitoring for Information Disclosure	FAU_SAR.1, FDP_ETC.1
AU-16	Cross-Organizational Auditing	FAU_SAR.1
CM-2	Baseline Configuration	EAL package
IA-3	Device Identification and Authentication	FIA_UAU.1, FIA_UAU.2, FIA_UAU.1, FIA_UAU.2
IA-9	Service Identification and Authentication	FIA_UAU.1, FIA_UAU.2, FIA_UAU.1, FIA_UAU.2
IA-10	Adaptive Identification and Authentication	FIA_UAU.1, FIA_UAU.2, FIA_UAU.1, FIA_UAU.2
MP-8	Media Downgrading	ALC_CMC, ALC_CMS
PE-6	Monitoring Physical Access	FPT_PHP.1, FPT_PHP.2, FPT_PHP.3
PE-8	Visitor Access Records	ALC_DVS
PS-2	Position Risk Designation	FPT_PHP.1, FPT_PHP.2, FPT_PHP.3
RA-6	Technical Surveillance Countermeasures Survey	AVA_VAN
SA-2	Allocation of Resources	FRU_RSA
SA-13	Trustworthiness	EAL package
SA-16	Developer-Provided Training	ALC_DVS
SA-20	Customized Development of Critical Components	ALC_CMC, ALC_CMS
SC-2	Application Partitioning	FIA_ATD.1
SC-18	Mobile Code	FMT_MSA.1, FMT_MSA.2
SC-19	Voice Over Internet Protocol	FMT_MSA.1, FMT_MSA.2
SC-20	Secure Name/Address Resolution Service (Authoritative Source)	FMT_MSA.1, FMT_MSA.2
SC-21	Secure Name/Address Resolution Service (Recursive or Caching Resolver)	FMT_MSA.1, FMT_MSA.2
SC-22	Architecture and Provisioning for Name/Address Resolution Service	FMT_MSA.1, FMT_MSA.2
SC-29	Heterogeneity	FDP_IFF
SC-32	Information System Partitioning	ADV_ARC
SC-37	Out-of-Band Channels	FPT_PHP.1, FPT_PHP.2, FPT_PHP.3
SC-39	Process Isolation	ADV_ARC
SC-42	Sensor Capability and Data	FDP_ETC.1, FDP_ETC.2
SC-43	Usage Restrictions	FDP_IFF.3
SI-8	Spam Protection	FDP_ACC.1, FDP_ACC.2 FDP_IFC.1,FDP_IFC.2
SI-11	Error Handling	FDP_ACC.1, FDP_ACC.2, FIA_AFL.1
SI-15	Information Output Filtering	FDP_ACC.1, FDP_ACC.2 FDP_IFC.1,FDP_IFC.2
SI-16	Memory Protection	ADV_ARC
SI-17	Fail-Safe Procedures	FPT_RCV, ADV_ARC

(User Identification) families in Functional Identification and Authentication (FIA) class. The selected components are shown in Table 5.

The reason is that the FIA_UID and FIA_UAU families are related to user authentication and identification but can also be described equivalently under the condition that the user of the CC is conceived as the same subject as the device in the NIST security controls. However, as shown in Table 6, security controls of some specific technologies that are not directly converted to the component or family unit can be mapped at the class level and organizational security policy pursued by illustration.

3.3. Comparison Analysis. Different from other evaluation frameworks, our proposed framework is intended for the integration of imported systems. It is set up based on RMF and Cybersecurity T&E of which their properties are inherited. However, as shown in Table 2, RMF does not carry

Table 4: Families of security functional requirements in ISO/IEC15408.

Family	Description
FAU	Security Audit
FCO	Communication
FCS	Cryptographic Support
FDP	User Data Protection
FIA	Identification and Authentication
FMT	Security Management
FPR	Privacy
FPT	Protection of the TSF
FRU	Resource utilization
FTA	TOE access
FTP	Trusted path/channels

Table 5: Example of security control conversion into CC component.

Components	Description
FIA_UAU.1	Timing of authentication, allows a user to perform certain actions prior to the authentication of the user's identity.
FIA_UAU.2	User authentication before any action, requires that users authenticate themselves before any action will be allowed by the TSF.
FIA_UID.1	Timing of identification, allows users to perform certain actions before being identified by the TSF.
FIA_UID.1	User identification before any action, require that users identify themselves before any action will be allowed by the TSF.

Table 6: Security controls that are converted on class level.

NIST	Security Controls	ISO/IEC15408 Family
SA-22	Unsupported System Components	FMT
SC-25	Thin Nodes	FRU
SC-26	Honeypots	FDP
SC-27	Platform-Independent Applications	FDP, FIA
SC-30	Concealment and Misdirection	FDP
SC-34	Non-Modifiable Executable Programs	FMT
SC-35	Honey clients	FDP
SC-36	Distributed Processing and Storage	FMT
SC-40	Wireless Link Protection	FIA,, FPT, FTA FTP
SC-44	Detonation Chambers	FDP
SI-14	Non-Persistence	FDP

all the mapping tables required for international standard; therefore, we propose an addition conversion method in Section 3.2. Table 7 is the comparison table of our framework with RMF, Cybersecurity T&E, and CC-based approaches.

4. Use-Case: Adoption of the Inertial Navigation System Import for Unmanned Aerial Vehicle

A formal evaluation upon our framework would be an ideal proof of framework's novelty. The framework we proposed however would be difficult to be proved formally as such evaluation is out of scope of this proposal. In lieu of a formal proof, we demonstrate use-case to explain how our proposed framework in Section 3 is applied to the weapon system purchasing process, an example of an inertial navigation system(INS) built in an unmanned aerial vehicle (UAV). It is noted that owing to confidentiality issue the components of military UAV are not disclosed; hence we use a general UAV instead for illustration in the use-case. An inertial navigation system is a navigation aid that uses a computer, motion sensors (accelerometers), rotation sensors (gyroscopes), and occasionally magnetic sensors (magnetometers), to continuously calculate by dead reckoning the position, the orientation and the velocity (direction and speed of movement) of a moving object without the need for external references. The INS is a critical part of computing data from sensors in UAV; thus, security is assured.

(A) Defining Requirements with Risk Identification (System Categorization). As presented in [30], the risk of INS in UAV should be evaluated by the parameter, i.e., velocity, attitude,

Table 7: Comparison with other approaches.

	Our framework	RMF	Cybersecurity T&E	CC based approaches
Purpose	Integrated into arms import process	Integrated into acquisition system	Eliminate vulnerabilities	Security evaluation of information products
For international use	○	△	X	○
Functional test	○	○	○	○
Operational test	○	○	○	X
Risk management	○	○	X	X

Table 8: Example of system categorization of INS.

Elements	Confidentiality	Integrity	Availability
		Provisional Impact Value	
Velocity	M	H	H
Attitude	M	H	H
Horizontal position	M	H	H
Depth	M	H	H
Total	M	H	H

horizontal position, and depth, which factor out the system categorization. When the confidentiality of the parameters does not impact much on duty, the system is graded as low; when the values of parameters are vague and calculation of position becomes infeasible, availability is classified as high; and integrity is marked as high too where inaccurate values deter an anticipation of position. Table 8 shows the risk evaluation result of INS.

(B) Selecting Appropriate Security Controls from the Requirements. Based on the result from Stage (A), the security control of INS brings the focus on the assurance of integrity and availability. The security controls of INS are compared with that of UAV for analysis to derive the necessary security controls of INS for later integration. The security controls inherent in UAV should then be excluded from the INS security control list. Since INS is a part of UAV, physical and environmental security controls against direct approach and accident could be inherited from UAV together with the security controls of management and operation security. Table 9 lists out the potentially inheritable security controls from UAV.

After tailoring out these inheritable security controls, the remaining INS security controls are concluded in Table 10.

AC-17, 18 are chosen because they carry Ethernet port; IA-2 is used for the identification of user and authentication of device to notify UAV of the identity of INS; IA-3 is to verify the authenticity of signal if it comes from INS. After selecting the proper security controls, acquirer should proceed to the next step.

(C) Converting the Security Controls into the International Standards. We convert selected elements in the security control into the international standards with Table 2 and [25]. Table 11 shows that result of converting security controls to international standards.

(D) Completion of RFP Evaluation Criteria and Preparation for Proposal Evaluation. We review the result of conversion whether it is reasonable. If, due to certain reason, the chosen security control is found inapplicable amid the evaluation process, its property should be further distinguished from compulsory to optional. For example, in the manner where IA-3 cannot be enacted unless it executes along with GPS (Global Positioning System), it is considered as an optional security control whereas IA-3 becomes compulsory when it is to be implemented in the absence of assistive device. Then, we prepare security evaluation in cases of function elements and nonfunction elements.

(E) Evaluation of Submitted Proposals and Preverification. If the security evaluation is ready, acquirer verifies documents from provider whether the documents meet the criteria of the security evaluation on the international standards. Then, we prepare the functional test by using open vulnerabilities on Common Vulnerabilities Exposures (CVE) and Common Weakness Enumeration (CWE) in order to save evaluation time. For instance, because Ethernet is adhered to INS, potential attacks arising from Ethernet are predictable and hence, Ethernet contributes to one part of the attack surface in this regard. Table 12 shows some of the common vulnerabilities found on Ethernet, and given the fact that time is limited in most cases, acquirer should opt for the most likely and influential ones. In situations where provider has already attained the approved ISO/IEC 15408 or ISO/IEC 27001, it is at acquirer's discretion to adopt the system as embedded or not.

(F) Functional Requirement Verification. This part is highly reliant on external factors such as the test time and budget given. Because of the adequate availability of papers in the respect of test method, we will not discuss in detail here.

Table 9: Example of potentially inheritable security controls from UAV and above.

SC	Description	SC	Description
AC-1	Access Control Policy and Procedures	PE-9	Power Equipment and Cabling
AC-2	Account Management	SC-1	System and Communications Protection Policy and Procedures
AU-1	Audit and Accountability Policy and Procedures	SC-7	Boundary Protection
AU-2	Audit Events	SC-8	Transmission Confidentiality and Integrity
PE-1	Physical and Environmental Protection Policy and Procedures	SC-12	Cryptographic Key Establishment and Management
PE-2	Physical Access Authorizations	SC-13	Cryptographic Protection
PE-3	Physical Access Control	SC-20	Secure Name /Address Resolution Service
PE-6	Monitoring Physical Access	SC-38	Operations Security

Table 10: Example of selected INS security controls.

SC	Description	SC	Description
AC-17	Remote Access	IA-2	Identification and Authentication
AC-18	Wireless Access	IA-3	Device Identification and Authentication

Table 11: Example of converting the security controls to ISO/IEC 15408 and 27001.

NIST	Security Controls	ISO/IEC15408 or ISO/IEC 27001 Requirements
AC-17	Remote Access	A.6.2.1: Mobile device policy: A.6.2.2: Teleworking A.13.2.1: Information transfer policies and procedures
AC-18	Wireless Access	A.6.2.1: Mobile device policy: A.13.1.1 Network controls A.13.2.1: Information transfer policies and procedures
IA-2	Identification and Authentication	FIA_ATD.1: User Attribute Definition FIA_UAU.1: User Authentication (Timing of Authentication) FIA_UAU.2: User Authentication before any action) FIA_UID.1: Timing of identification FIA_UID.2: User Identification before any action
IA-3	Device Identification and Authentication	FIA_UAU.1: User Authentication (Timing of Authentication) FIA_UAU.2: User Authentication before any action) FIA_UID.1: Timing of identification FIA_UID.2: User Identification before any action

Table 12: Example of selected CVE on Ethernet vulnerabilities.

Name	Description
CVE-2017-9628	An Information Exposure issue
CVE-2017-9945	a Denial-of-Service condition could be induced by a specially crafted PROFINET DCP packet sent as a local Ethernet (Layer 2) broadcast.
CVE-2017-3726	a privilege escalation vulnerability
CVE-2016-8106	vulnerable to a denial of service in certain layer 2 network configurations.

Table 13: Example of assessment scale-level of risk [14].

Overall Likelihood	Impact Level of Risk				
	Very Low	**Low**	**Moderate**	**High**	**Very High**
Very High	Very Low	Low	Moderate	High	Very High
High	Very Low	Low	Moderate	High	Very High
Moderate	Very Low	Low	Moderate	Moderate	High
Low	Very Low	Low	Low	Low	Moderate
Very Low	Very Low	Very Low	Very Low	Low	Low

The requirement verification lies in the test method, which is decided by acquirer or settled between acquirer and provider.

(G) Risk Assessment. If the IA-3 security control failed in the functional test, we should reevaluate the impact level of the risk about IA-3 failure by referring to Table 13 [14]. As demonstrated in Table 12 the possibility of direct attacks to communication between INS and the rotation sensor is low, but the resulting damage can result in catastrophic consequences of UAV failing to fly properly (high). As a result, owing to the fact that the overall risk is low, acquirer still manages to continue the process even if IA-3 declines)

(H) Secure System Integration and Operational Test & Evaluation. If the whole evaluation ends successfully, acquirer could commence the integration of INS into UAV, after which the development and evaluation of UAV are to be deployed. The postintegration process follows the existing procedures of the acquirer.

5. Conclusion

Provided that the evaluation framework is system-specific, variations emerge in the midst of negotiations between countries; it is therefore difficult to employ the same approach to every scenario in reality. Notwithstanding, an existence of a well-established framework to guide, in part, is beneficial upon the trading of systems. We proposed the security-enhanced framework based on the Risk Management Framework and Cybersecurity Test & Evaluation, and this framework has been designed to be integrated into import process of weapon systems, but not developmental process. Also, we have raised the assurance level of the weapon system via a well-structured framework instead of mathematical (formal) verification. Within this framework, the cybersecurity of weapon systems is ensured throughout the lifecycle of weapon systems, from the development stage of the provider to the operation stage of the acquirer. In addition, by employing international standards rather than using domestic security controls under the proposed framework, it facilitates the weapon system acquisition communication between the acquirer and provider. We reinforced the construction of this framework with reference to NIST documents; however, we did not recommend an evaluation methodology relevant to this framework. Some nations might encounter difficulties in applying our framework into their weapon system acquire process. Therefore, future studies would require a more detailed evaluation method for this framework and for compositing security evaluation in order to address the assurance level of weapon systems.

Data Availability

The datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2018-2015-0-00403) supervised by the IITP (Institute for Information & Communications Technology Promotion).

References

[1] L. Yushi, J. Fei, and Y. Hui, "Study on application modes of military internet of things (miot)," in *Proceedings of the IEEE International Conference*, vol. 3, pp. 630–634, Computer Science and Automation Engineering (CSAE), 2012.

[2] J. P. Farwell and R. Rohozinski, "Stuxnet and the future of cyber war," *Survival*, vol. 53, no. 1, pp. 23–40, 2011.

[3] A. Kim, B. Wampler, J. Goppert, I. Hwang, and H. Aldridge, "Cyber attack vulnerabilities analysis for unmanned aerial vehicles," in *Proceedings of the AIAA Infotech at Aerospace Conference and Exhibit 2012*, USA, June 2012.

[4] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," in *Proceedings of the Black Hat Briefings*, USA, 2015.

[5] A. Futter, *The Politics of Nuclear Weapons*, SAGE Publications Ltd, 1 Oliver's Yard, 55 City Road London EC1Y 1SP , 2015.

[6] C. Dougherty, K. Sayre, R. C. Seacord, D. Svoboda, and K. Togashi, "Secure design patterns," Tech. Rep., Carnegie Mellon University, Pittsburgh, Pa, USA, Software Engineering Institute, 2009.

[7] N. Davis, "Secure software development life cycle processes: A technology scouting report," Tech. Rep., Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, A technology scouting report, 2005.

[8] C. Mundie, P. de Vries, P. Haynes, and M. Corwine, "Trustworthy computing," Technical Report, 2002.

[9] K. Fisher, "Using formal methods to enable more secure vehicles," *ACM SIGPLAN Notices*, vol. 49, no. 9, pp. 1-1, 2014.

[10] M. Schwartz, *Defense Acquisitions: How Dod Acquires Weapon Systems And Recent Efforts to Reform The Process*, Library of Congress Washington Dc Congressional Research Service, 2010.

[11] H.-G. Albertsen and F. Forge, "The modular approach: A composite product evaluation for smart cards," in *Proceedings of the 3rd International Common Criteria Conference-Common Criteria: Delivering Information Assurance Solutions*, 2002.

[12] K. Muller, M. Paulitsch, S. Tverdyshev, and H. Blasum, "MILS-related information flow control in the avionic domain: A view on security-enhancing software architectures," in *Proceedings of the 2012 IEEE/IFIP 42nd International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pp. 1–6, Boston, Mass, USA, June 2012.

[13] B. S. Blanchard and W. J. Fabrycky, "System engineering and analysis," 5th ed. Upper Saddle River, N. J. : Pearson, Prentice-Hall international series in industrial and systems engineering, 2011.

[14] S. Ross Ronald, "Guide for conducting risk assessments," No. Special Publication (NIST SP)-800-30r1. September, 2012.

[15] A. B. Jeng and Y.-M. Yu, "Analysis of the composition problems in cc v3. 1 rev. 1 with some suggested solutions, ICCC, 2006".

[16] R. S. Ross, "Security and privacy controls for federal information systems and organizations," Tech. Rep., 2013.

[17] R. Von Solms, "Information security management (3): The Code of Practice for Information Security Management (BS 7799)," *Information Management & Computer Security*, vol. 6, no. 5, pp. 224-225, 1998.

[18] S. L. Brand, *Dod 5200.28-std Department of Defense Trusted Computer System Evaluation Criteria (orange book)*, National Computer Security Center, 1985.

[19] C. E. Landwehr, "Computer security," *International Journal of Information Security*, vol. 1, no. 1, pp. 3–13, 2001.

[20] I. ISO and I. Std, "Iso 15408-2: 2009, Information technology-Security techniques-Evaluation criteria for IT security-Part 2".

[21] G. Disterer, "ISO/IEC 27000, 27001 and 27002 for Information Security Management," *Journal of Information Security*, vol. 04, no. 02, pp. 92–100, 2013.

[22] R. Sandoval, "Information systems development (isd) and the national institute of standards and technology (nist) risk management framework," 2017.

[23] D. Instruction, "5200.40, dod information technology security certification and accreditation process (ditscap)," December, 1997.

[24] D. Instruction, "8510.01, dod information assurance certification and accreditation process (diacap), november 28, 2007".

[25] NIST, *Guide for Applying the Risk Management Framework to Federal Information Systems: A Security Life Cycle Approach*, February, 2010.

[26] D. Instruction, 8510.01, Risk Management Framework (RMF) for DoD Information Technology (IT), 2014.

[27] K. F. Joiner, S. R. Atkinson, and E. Sitnikova, *AUSTRALIA'S FUTURE SUBMARINE: Cybersecurity Challenges and Processes*, 2017.

[28] D. Instruction, DoD Cybersecurity Test and Evaluation Guidebook, June 26, 2015.

[29] Y. Y. Haimes, *Risk Modeling, Assessment, and Management*, John Wiley & Sons, 2015.

[30] M. Kevin, R. L. Kissel, W. C. Barker et al., *Guide for Mapping Types of Information and Information Systems to Security Categories (2 Volume)*, NIST, 2008.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 7202598, 13 pages
https://doi.org/10.1155/2018/7202598

Research Article

A Secure and Privacy-Aware Smart Health System with Secret Key Leakage Resilience

Yinghui Zhang[1,2,3] **Pengzhen Lang**[1] **Dong Zheng**[1,2] **Menglei Yang**[1] **and Rui Guo**[1]

[1]*National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications, Xi'an 710121, China*
[2]*Westone Cryptologic Research Center, Beijing 100070, China*
[3]*State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China*

Correspondence should be addressed to Yinghui Zhang; yhzhaang@163.com

Received 30 January 2018; Revised 16 April 2018; Accepted 30 May 2018; Published 24 June 2018

Academic Editor: Karl Andersson

With the development of the smart health (s-health), data security and patient privacy are becoming more and more important. However, some traditional cryptographic schemes can not guarantee data security and patient privacy under various forms of leakage attacks. To prevent the adversary from capturing the part of private keys by leakage attacks, we propose a secure leakage-resilient s-health system which realizes privacy protection and the safe transmission of medical information in the case of leakage attacks. The key technique is a promising public key cryptographic primitive called leakage-resilient anonymous Hierarchical Identity-Based Encryption. Our construction is proved to be secure against chosen plaintext attacks in the standard model under the Diffie-Hellman exponent assumption and decisional linear assumption. We also blind the public parameters and ciphertexts by using double exponent technique to achieve the recipient anonymity. Finally, the performance analysis shows the practicability of our scheme, and the leakage rate of the private key approximates to 1/6.

1. Introduction

With the development of information technology, the Internet of Things (IoT) has become a very important technology for government departments, businesses, and academic circles in various countries with its huge application scenes. The technology of IoT is based on the Internet to achieve the communication between information and terminal equipment, information, and real goods. At present, IoT has been widely used in many fields, like food safety, smart health (s-health), urban construction, cloud storage, etc. [1–4].

The cloud-based s-health systems play an important role in our daily life. At present, doctors use computers to store and retrieve patients' electronic health records (EHRs). EHRs systems replace paper systems and thus increase efficiency in the recording, storage, and retrieval of patients information [5–7]. However, EHRs which reveal highly confidential personal information stored in the cloud and exchanged over the Internet can be vulnerable to attack [8–10], such as data loss, hackers hijacking information, and integrity. In recent years, a large number of medical information leaks in cloud storage have attracted more and more people's attention. Data privacy and security is believed to be the major challenge in the deployment of EHR-based healthcare system. There has been a lot of work that deals with data privacy and security problems [11–15].

However, the traditional cryptographic schemes may not be secure under various forms of leakage attacks, such as side-channel attacks [16] and cold-boot attack [17]. Such attacks exploit various forms of information leakage by observing physical implementations of cryptosystems, such as running time [18], electromagnetic radiation [19], power consumption, and fault detection [20]. IoT generally adopts discrete network structure, and most of the nodes are located outdoors, which makes it easy for attackers to obtain sensitive information [21–23]. Therefore, it is very meaningful to construct a secure and privacy-aware smart health system with secret key leakage resilience in the case of leakage attacks. Based on the paper of Zhang et al. [24], we construct

a leakage-resilient anonymous HIBE scheme in s-health data sharing [25–27] scenarios.

1.1. Our Contributions. To address the medical information security and privacy of the patients issues in s-health, we propose a secure s-health system which allows a medical information owner to securely share data in the case of leakage attacks. The main contributions of this paper as follows.

(i) Firstly, we present the system model of a secure s-health system based on a leakage-resilient anonymous HIBE scheme. Our system addresses the problem of key management and reduces the pressure of PKG.

(ii) Secondly, we blind the public parameters and ciphertexts using double exponent technique to achieve the anonymity, so as to achieve the effect of protecting privacy.

(iii) Finally, our scheme is built in prime order groups that are more computationally efficient than composite order groups. The proposed scheme is proved to be secure against chosen plaintext attacks in the standard model.

1.2. Related Work. More attention to identity-based encryption (IBE) has been attracted since the notion of IBE was introduced by Shamir [28]. The private key for the user in traditional IBE schemes [29] is generated by the Private Key Generation Center (PKG). However, in a large scale network, such as s-health, the number of users is huge, and the task of PKG is too heavy and it is difficult to manage the user's private key. In order to solve this problem, the notion of Hierarchical Identity-Based Encryption (HIBE) was proposed [30] and then many HIBE schemes were proposed [31, 32]. What is more, many anonymous HIBE schemes were proposed [33–36] where the ciphertext does not leak the identity of the recipient. However, their sizes of private keys and ciphertexts increase with the depth of identity hierarchy. Zhang et al. [24] proposed an anonymous HIBE scheme over prime order groups where both private keys and ciphertext have a constant size.

Dillema et al. [37] proposed a simple cryptographic access control method in the prehospital environment. However, this system provides access privilege if and only if patient and health worker meet in the physical world. Zhang et al. [38] proposed a reference model of the security and privacy issues in the EHR cloud and requirements for secure access of EHR data. A secure EHR system demonstrated to be resilient to various attacks to protect patient privacy and enable emergency healthcare was proposed by Sun et al. [39]. In e-Health and Mobile Health network, Guo et al. [40, 41] proposed a privacy-preserving attribute-based authentication system, which leverages users verifiable attributes to authenticate users while preserving their privacy issues. Kumar et al. [42] proposed a biometric based authentication scheme which is lightweight and solely uses symmetric key based operations. A secure data sharing using IBE scheme for the implementation of data sharing in the e-Healthcare system was proposed by Sudarson et al. [43]. Zhang et al. [44] proposed a system architecture and adversary model of a secure s-health system which realizes fine-grained access control on s-health cloud data and hence ensures users privacy protection. Dawoud et al. [45] defined different scenarios for the integration of the e-health systems with the cloud computing systems and these scenarios discussed the authentication and data processing in the different parts of the system. Zhang et al. [46] proposed a three-factor authenticated key agreement scheme based on a dynamic authentication mechanism to protect the users privacy using for e-health systems, and it was proved to be semantically secure under the real or random model. Sahi et al. [47] reviewed the latest research with regard to privacy preservation in e-Healthcare and explored whether this research offers any possible solutions to patient privacy requirements for e-Healthcare. However, these schemes may not be secure under various forms of leakage attacks.

The first leakage-resilient cryptographic scheme was proposed by Dziembowski and Pietrzak [48] that can capture most of the key leakage attacks. However, they constructed the leakage-resilient encryption scheme based on "only computation leaks information" which can not capture the cold-boot attack. To resist the cold-boot attack, the bounded-leakage model [49] was proposed by Akavia et al. What is more, the relative-leakage model [50] was proposed by Naor et al. Leakage resilience (anonymous) IBE schemes have been discussed previously. A leakage-resilient IBE scheme was proposed and showed being secure in the standard model by Alwen et al. [51]. Chow et al. [52] proposed three new leakage-resilient IBE schemes under the respective static assumptions of the original systems. Li et al. [53] proposed a new leakage-resilient public key encryption and showed that it was secure under Decisional Diffie-Hellman (DDH) assumption. Liu et al. [54] showed that the techniques of dual system technique lead to leakage resilience and proposed an anonymous leakage-resilient identity-based encryption scheme. Li et al. [55] proposed a new leakage-resilient IBE scheme in the bounded-leakage model and showed being semantically secure against adaptive chosen ciphertext attack in the standard model.

1.3. Organization. Some preliminaries are reviewed in Section 2. In Section 3, we define the usage scenario for smart healthcare system and present the system model and leakage-resilient security model. The secure s-health system based on leakage-resilient anonymous HIBE is described in Section 4. In Section 5, our security analysis and leakage resilience analysis are described. Finally, we draw our conclusions in Section 6.

2. Preliminaries

2.1. Notations. For ease of reference, important notations are summarized in Table 1.

2.2. Random Extractor. We define the statistical distance between two random variables X and Y over a finite domain Ω to be

$$SD(X,Y) = \frac{1}{2}\sum_{w\in\Omega} |Pr[X = w] - Pr[Y = w]|. \quad (1)$$

Table 1: Notations used throughout the paper.

Notation	Description	Notation	Description	Notation	Description
SD	statistical distance	X, Y	random variable	Ω	finite domain
Pr	probability	$H_\infty(X)$	min-entropy	$\widetilde{H}_\infty(X \mid Y)$	average min-entropy
e	bilinear map	Ext	extractor	G, G_T	cyclic group
g	generator	$\mathcal{A}, \mathcal{B}$	algorithm	ε	advantage
PK	public key	MSK	master secret key	ID	identities
d_{ID}	private key	$L_{d_{ID}}$	number of leaked bits	D_{ID}	rerandomize private keys
$\ell(\lambda)$	leakage parameter	f	leakage function	$\perp$	reject symbol
M	message	CT	ciphertext	p	large prime number
v_{ij}	vector of Identity	h_{ij}	auxiliary parameters	s	random seed
l	maximum depth of HIBE	V	view without leakage	ρ	leakage ratio

The min-entropy of a random variable X is defined as

$$H_\infty(X) = -\log\left(\max_x Pr[X = x]\right). \tag{2}$$

The average min-entropy of a random variable X conditioned on another random variable Y is defined as follows:

$$\widetilde{H}_\infty(X \mid Y) = -\log\left(E_{y \leftarrow Y}\left[2^{-H_\infty(X|Y=y)}\right]\right). \tag{3}$$

Definition 1. If, for all pairs of random variables (X, Y) such that $X \in (0,1)^u$ and $\widetilde{H}_\infty(X \mid Y) \geq k$, it holds that

$$SD\left((Ext(X,S), S, Y), (U_m, S, Y)\right) \leq \varepsilon, \tag{4}$$

where S is uniform over $(0,1)^t$, we call polynomial-time function $Ext : (0,1)^u \times (0,1)^t \rightarrow (0,1)^m$ an average-case (k, ε)-strong extractor.

Lemma 2. *If* Y *has* 2^r *possible values and Z is random variable, we have*

$$\widetilde{H}_\infty(X \mid (Y, Z)) \geq \widetilde{H}_\infty(X \mid Z) - r. \tag{5}$$

2.3. Bilinear Groups. Let G and G_T be two cyclic groups of prime order p and $e : G \times G \rightarrow G_T$ be a bilinear map. We call G a bilinear group if it has the following properties:

(i) Bilinearity: $\forall u, v \in G$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.

(ii) Nondegeneracy: for the generator $g \in G$, we have $e(g, g) \neq 1$.

(iii) Computability: $\forall u, v \in G$, the bilinear map $e(u, v)$ can be efficiently computed.

2.4. Computational Assumptions

2.4.1. Bilinear Diffie-Hellman Exponent (BDHE) Assumption. Let g be a generator of group G and $\alpha, c \in Z_p^*$. We define the computational $(n+1)$−BDHE problem [56] to be

$$\begin{aligned} &input : \left(g, y_0, y_1, \ldots, y_n, y_{n+2}, \ldots, y_{2n+2}\right) \in G, \\ &output : e\left(g, y_0\right)^{\alpha^{n+1}} \left(= e\left(g, g\right)^{\alpha^{n+1} c}\right), \end{aligned} \tag{6}$$

where $y_0 = g^c$, $y_i = g^{\alpha^i}$. Algorithm $\mathcal{A}$ has advantage ε in solving the computational $(n+1)$−BDHE problem if

$$\begin{aligned} &Pr\left(\mathcal{A}\left(g, y_0, y_1, \ldots, y_n, y_{n+2}, \ldots, y_{2n+2}\right)\right. \\ &\left. = e\left(g, y_0\right)^{\alpha^{n+1}}\right) \geq \varepsilon. \end{aligned} \tag{7}$$

The decisional version of the $(n+1)$−BDHE problem is defined in the usual manner.

$$\begin{aligned} &input : \left(g, y_0, y_1, \ldots, y_n, y_{n+2}, \ldots, y_{2n+2}, K\right), \\ &ouput : b \in \{0, 1\}. \end{aligned} \tag{8}$$

Let $K' = (g, y_0, y_1, \ldots, y_n, y_{n+2}, \ldots, y_{2n+2})$; Algorithm $\mathcal{B}$ has advantage ε in solving the decisional $(n+1)$−BDHE problem if

$$\begin{aligned} &Pr\left(\mathcal{B}\left(K', e\left(g, y_0\right)^{\alpha^{n+1}}\right) = 0\right) - Pr\left(\mathcal{B}\left(K', K\right) = 0\right) \\ &\geq \varepsilon. \end{aligned} \tag{9}$$

2.4.2. Decisional Linear Assumption. The decisional linear problem [57] is defined as

$$\begin{aligned} &input : \left(g, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, K\right), \\ &ouput : b \in \{0, 1\}. \end{aligned} \tag{10}$$

Algorithm $\mathcal{A}'s$ goal is to output 1 when $K = g^{z_3 + z_4}$ or 0 otherwise. We give three weak versions of the decisional linear problem [35] as follows:

(i) Version (1):

$$\begin{aligned} &input : \\ &\left(g, g^{z_1}, g^{z_2}, g^{z_2^2}, \ldots, g^{z_2^{n+1}}, g^{z_2^n / z_1}, g^{z_2^{n+1} / z_3}, g^{z_4}, K\right), \\ &ouput : b \in \{0, 1\}. \end{aligned} \tag{11}$$

Algorithm $\mathcal{A}'s$ goal is to output 1 when $K = g^{z_1(z_3 + z_4)}$ or 0 otherwise.

(ii) Version (2):

$$input : \left(g, g^{z_1}, g^{z_2}, g^{z_2^2}, \ldots, g^{z_2^n}, g^{z_2^{n+2}}, \ldots, g^{z_2^{2n}}, g^{z_3}, g^{z_4}, g^{z_4 z_2}, \ldots, g^{z_2^n z_4}, K\right), \quad (12)$$

$$ouput : b \in \{0, 1\}.$$

Algorithm $\mathscr{A}'s$ goal is to output 1 when $K = g^{z_1(z_3+z_4)}$ or 0 otherwise.

(iii) Version (3):

$$input : \left(g, g^{z_1}, g^{z_2}, g^{z_2^2}, \ldots, g^{z_2^n}, g^{z_2^{n+2}}, g^{z_2^{2n}}, \ldots, g^{z_3}, g^{z_4}, g^{z_4 z_2}, \ldots, g^{z_2^n z_4}, g^{z_1 z_2}, \ldots, g^{z_2^n z_1}, K\right), \quad (13)$$

$$ouput : b \in \{0, 1\}.$$

Algorithm $\mathscr{A}'s$ goal is to output 1 when $K = g^{z_1(z_3+z_4)}$ or 0 otherwise.

3. System Model

3.1. Usage Scenario. The hospital uses the system software developed by our proposal, and each member of the hospital is registered in the system at a certain level. The system allocates private keys to them through the key generation algorithm; however, the private key generated may be leaked partly by malicious attackers through various forms of leakage attacks.

A patient, named Alice, visits a doctor in this hospital. According to condition, a nurse assigns Alice to the doctor named Bob. Through diagnosis, Bob thinks that Alice needs the doctor named Carol to treat the illness together. And Bob uploads Alice's EHRs to the cloud server with public key of Carol through the system. Carol uses his private key to download and decrypt Alice's EHRs. They complete the diagnosis and treatment of Alice.

During the entire process, Bob sends Alice's EHRs to Carol through the cloud, but Carol's private key may have leaked partly. If the general system is used, Alice's EHRs may be leaked, but our program can ensure that Alice's EHRs will not be leaked. Thus, the patient's EHRs have been protected safely.

3.2. System Model. We divide the system model into two parts, and the first part is shown in Figure 1, which is to produce the private keys to the different level of users (patients, doctors). The S-Health Authority (SHA) is an entity that produces the public key parameters and the master secret key. In our system, the level is divided into l levels. The private key of users in the first level is defined as the root private key. The private key of the k-th level users is related to the private key of the $(k-1)$-th level users, where $k \leq l$.

The second part is shown in Figure 2, which is to share medical information among all users. It is described in the picture where doctor A shares medical information to patient B. A encrypts medical information with B's public key

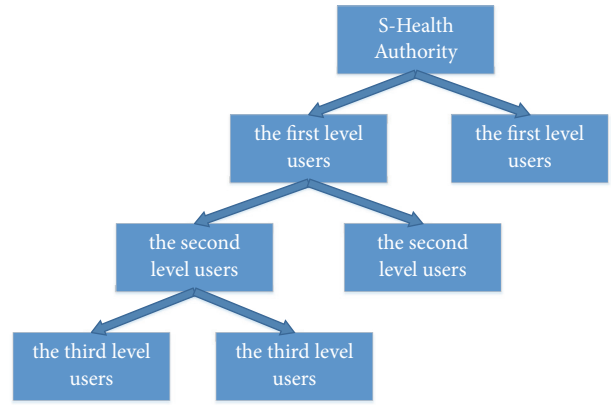


Figure 1: The generation of the private keys.

(identity) and then uploads it to the s-health cloud (SHC). Then B decrypts medical information with its own private key. The adversary $\mathscr{A}$ can know part of the private keys information of B through the leak attack.

We define an description of the proposed secure s-health system:

(i) Initialization: SHA produces the public key parameters PK and the master secret key MSK. All users can obtain PK.

(ii) User Registration: A user (a patient, a doctor) can join the s-health system by confirming its level to SHA.

(iii) Information Upload: A user encrypts medical information based on a leakage-resilient anonymous HIBE scheme and uploads the final ciphertext to SHC.

(iv) Information Access: A user downloads a ciphertext from SHC. The ciphertext can be decrypted if and only if the private keys correspond to the public key used for encryption.

3.3. Leakage-Resilient Security Model for Anonymous HIBE. The security of leakage-resilient anonymous HIBE is defined by the following game ($\boldsymbol{Game}_{\text{real}}$) between an adversary $\mathscr{A}$ and a challenger $\mathscr{C}$.

(i) Init: The adversary $\mathscr{A}$ gives the challenge identity ID^* to the challenger $\mathscr{C}$.

(ii) Setup: $\mathscr{C}$ computes $(PK, MSK) \leftarrow Setup(\lambda)$. $\mathscr{C}$ gives PK to $\mathscr{A}$ and keeps MSK to itself. $\mathscr{C}$ will initialize a set $S = \emptyset$, which will be the set of tuples of identities; private keys have been created and the number of leaked bits corresponds to the private key d_{ID} $(ID, d_{ID}, L_{d_{ID}})$. Let $\ell(\lambda)$ be a leakage parameter.

(iii) Phase 1: $\mathscr{A}$ adaptively issues the following two kinds of queries:

(a) Private Key Queries: $\mathscr{A}$ adaptively queries $\mathscr{C}$ with ID where $ID \neq ID^*$ and ID is not a prefix of ID^*; $\mathscr{C}$ responds with the private key d_{ID} corresponding to the identity ID.

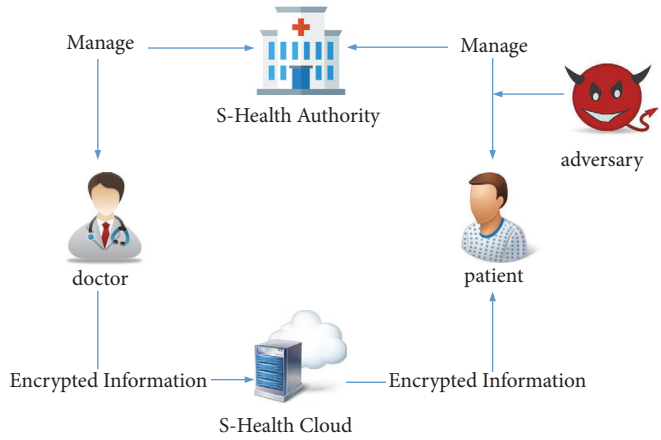


Figure 2: The sharing of medical information.

(b) Leak Queries: $\mathscr{A}$ gives a polynomial-time leakage function $f : \{0,1\} \to \{0,1\}$ and adaptively queries $\mathscr{C}$ with ID. $\mathscr{C}$ finds the tuple $(ID, d_{ID}, L_{d_{ID}})$ and replies with $f(d_{ID})$ when $L_{d_{ID}} + |f(d_{ID})| \leq \ell(\lambda)$ or a reject symbol $\perp$.

(iv) Challenge: $\mathscr{A}$ selects two messages M_0, M_1 on which it wishes to be challenged. $\mathscr{C}$ chooses a random bit $b \leftarrow \{0,1\}$ and gives $CT = Encrypt(params, ID^*, M_b)$ to $\mathscr{A}$.

(v) Phase 2: $\mathscr{C}$ answers the queries in the same way as phase 1 with the added restriction that $\mathscr{A}$ can not execute leakage queries.

(vi) Guess: $\mathscr{A}$ outputs a bit $b' \in \{0,1\}$ and wins if $b' = b$.

Definition 3. We say that an anonymous HIBE scheme is leakage-resilient and selectively secure against chosen plaintext attacks (ANO-IND-sID-CPA) if all polynomial-time adversaries $\mathscr{A}$'s advantage is negligible in the above game. We define $\mathscr{A}$'s advantage to be

$$\boldsymbol{Adv}_{HIBE,\mathscr{A}}^{ANO-IND-sID-CPA} = \left| Pr\left[b = b'\right] - \frac{1}{2} \right|. \quad (14)$$

4. Secure s-Health System

4.1. A Leakage-Resilient Anonymous HIBE Scheme. For an HIBE of maximum depth l and an identity $ID_j = (v_1, \cdots, v_j)$ where $1 \leq j \leq l$, a leakage-resilient anonymous HIBE scheme is defined as follows:

(i) Setup$(\lambda) \to (PK, MSK)$: The Setup algorithm takes a security parameter λ and produces the public key parameters PK and the master secret key MSK. All users can obtain PK.

(ii) KenGen$(PK, ID_j, d_{ID|j-1}) \to d_{ID|j}$: The KenGen algorithm takes as input the public key PK, an identity $ID_j = (v_1, \ldots, v_j)$, and the private key $d_{ID|j-1}$ corresponding to the identity $(v_1, \ldots, v_{j-1})$. It outputs the private key $d_{ID|j}$.

(iii) Encrypt$(PK, ID_j, M) \to CT$: The Encrypt algorithm takes as input the public key PK, an identity $ID_j = (v_1, \ldots, v_j)$, and a message M. It outputs a ciphertext CT.

(iv) Decrypt$(PK, d_{ID|j}, CT) \to M$: The Decrypt algorithm takes as input the public key PK, a ciphertext CT, and the private key $d_{ID|j}$ corresponding to the identity $ID_j = (v_1, \ldots, v_j)$. It outputs the message M or a reject symbol $\perp$ if the ciphertext is invalid.

4.2. Description of Secure s-Health System. Let p be a large prime number and G be a group of order p. Let l be the HIBE of maximum depth and represent identity information as bit strings of length n [58].

(i) Initialization: SHA randomly chooses $\alpha, t_1, t_2, t_3 \in Z_p$. Set $v = g^{t_1}$, $g_1 = v^{\alpha}$, $u = g^{t_2}$, $x = g^{t_3}$, and then pick g_2, w in group G at random. The public key parameters are

$$PK = \{g, g_1, g_2, v, u, x, w\}. \quad (15)$$

The master secret key is $MSK = g_2^{\alpha}$.

(ii) User Registration: The user joins the s-health system and gets its private keys. The user initiates the following key generation protocol.

KeyGen: The KeyGen algorithm is defined as follows.

(a) Root private keys: For the first level user $ID_1 = (v_1)$ where $v_1 = (v_{11}, \ldots, v_{1n})$ and $v_{1i} \in \{0,1\}$, root PKG randomly chooses $\{\alpha_{11}, \ldots, \alpha_{1n}, \beta_{11}, \ldots, \beta_{1n}\} \in Z_p$ and produces the auxiliary parameters

$$h_{1i} = \left(h_{1(i-1)}\right)^{\alpha_{1i}^{v_{1i}} \beta_{1i}^{1-v_{1i}}}, \quad (16)$$

where $h_{10} = g$, $1 \leq i \leq n$, and h_{1n} is outputted as the public key. Root PKG outputs the private key for ID_1 as

$$d_{ID_1} = (d_{10}, d_{11}, d_{12}, d_{13}, d_{14}, d_{15}) = \left(g_2^{\alpha} h_{1n}^{r_1}, g^{r_1}, v^{r_1}, w^{r_1}, u^{r_1 t_1}, x^{r_1 t_1}\right) \quad (17)$$

and

$$D_{ID_1} = (D_{10}, D_{11}, D_{12}, D_{13}, D_{14}, D_{15}) = \left(h_{1n}^{r_2}, g^{r_2}, v^{r_2}, w^{r_2}, u^{r_2 t_1}, x^{r_2 t_1}\right), \quad (18)$$

where $r_1, r_2 \in Z_p$ and D_{ID_1} are used to rerandomize the private keys.

(b) Delegate: For the k-th level user $ID_k = (v_1, \ldots, v_k)$ where $k \le l$, $v_i = (v_{i1}, \ldots, v_{in})$ and $v_{ij} \in \{0, 1\}$, by using the private key $d_{ID|k-1}$ corresponding to the $(k-1)$-th level user $ID_{k-1} = (v_1, \ldots, v_{k-1})$:

$$d_{ID_{k-1}} = (d_{(k-1)0}, d_{(k-1)1}, d_{(k-1)2}, d_{(k-1)3}, d_{(k-1)4}, d_{(k-1)5}) = \left(g_2^{\alpha} \left(\prod_{i=1}^{k-1} h_{in}\right)^{r_1'}, g^{r_1'}, v^{r_1'}, w^{r_1'}, u^{r_1' t_1}, x^{r_1' t_1}\right) \quad (19)$$

and

$$D_{ID_{k-1}} = (D_{(k-1)0}, D_{(k-1)1}, D_{(k-1)2}, D_{(k-1)3}, D_{(k-1)4}, D_{(k-1)5}) = \left(\left(\prod_{i=1}^{k-1} h_{in}\right)^{r_2'}, g^{r_2'}, v^{r_2'}, w^{r_2'}, u^{r_2' t_1}, x^{r_2' t_1}\right). \quad (20)$$

PKG_k randomly chooses

$$\{\alpha_{k1}, \ldots, \alpha_{kn}, \beta_{k1}, \ldots, \beta_{kn}\} \in Z_p, \quad (21)$$

and it produces the auxiliary parameters

$$h'_{kj} = \left(h'_{k(j-1)}\right)^{\alpha_{kj}^{v_{kj}} \beta_{kj}^{1-v_{kj}}} = \left(g^{r_1'}\right)^{\prod_{j=1}^{n} \alpha_{kj}^{v_{kj}} \beta_{kj}^{1-v_{kj}}},$$
$$h''_{kj} = \left(h''_{k(j-1)}\right)^{\alpha_{kj}^{v_{kj}} \beta_{kj}^{1-v_{kj}}} = \left(g^{r_2'}\right)^{\prod_{j=1}^{n} \alpha_{kj}^{v_{kj}} \beta_{kj}^{1-v_{kj}}}, \quad (22)$$

where $h'_{k0} = d_{(k-1)1}$, $h''_{k0} = D_{(k-1)1}$ and $1 \le j \le n$.

Let $h_{kn} = g^{\prod_{j=1}^{n} \alpha_{kj}^{v_{kj}} \beta_{kj}^{1-v_{kj}}}$; then one can obtain $h'_{kn} = (h_{kn})^{r_1'}$ and $h''_{kn} = (h_{kn})^{r_2'}$. PKG_k outputs the private key for ID_k as

$$d_{ID_k} = (d_{k0}, d_{k1}, d_{k2}, d_{k3}, d_{k4}, d_{k5}) = \left(d_{(k-1)0} \left(h'_{kn}\right)^{r'} \cdot \left(D_{(k-1)0} h''_{kn}\right)^{r'}, d_{(k-1)1} D_{(k-1)1}^{r'}, d_{(k-1)2} D_{(k-1)2}^{r'}, d_{(k-1)3} D_{(k-1)3}^{r'}, d_{(k-1)4} D_{(k-1)4}^{r'}, d_{(k-1)5} D_{(k-1)5}^{r'}\right) = \left(g_2^{\alpha} \left(\prod_{i=1}^{k} h_{in}\right)^{r_1}, g^{r_1}, v^{r_1}, w^{r_1}, u^{r_1 t_1}, x^{r_1 t_1}\right) \quad (23)$$

and

$$D_{ID_k} = (D_{k0}, D_{k1}, D_{k2}, D_{k3}, D_{k4}, D_{k5}) = \left(\left(D_{(k-1)0} h''_{kn}\right)^{r'}, D_{(k-1)1}^{r'}, D_{(k-1)2}^{r'}, D_{(k-1)3}^{r'}, D_{(k-1)4}^{r'}, D_{(k-1)5}^{r'}\right) = \left(\left(\prod_{i=1}^{k} h_{in}\right)^{r_2}, g^{r_2}, v^{r_2}, w^{r_2}, u^{r_2 t_1}, x^{r_2 t_1}\right), \quad (24)$$

where $r_1 = r_1' + r' r_2', r_2 = r' r_2', r' \in Z_p^*$.

(iii) Information Upload: A user encrypts medical information based on a leakage-resilient anonymous HIBE scheme as follows.

Encrypt: The encryptor randomly chooses $s_1, s_2 \in Z_p$ and a random seed $s \in \{0, 1\}^t$. The encryptor creates the ciphertext as follows:

$$CT = (C_0, C_1, C_2, C_3, C_4, C_5) = \left(M \oplus Ext\left(e(g_1, g_2)^{s_1}, s\right), \left(\prod_{i=1}^{k} h_{in}\right)^{s_1} \cdot u^{s_2}, w^{s_1} x^{s_2}, g^{s_2}, v^{s_1}, s\right). \quad (25)$$

(iv) Information Access: A user downloads and decrypts a ciphertext from SHC as follows.

Decrypt: The k-th level user decrypts a ciphertext CT using the private key $d_{ID|k}$ and computes

$$M = Ext\left(\frac{e(d_{k0}, C_4)\, e(d_{k4} d_{k5}, C_3)\, e(d_{k3}, C_4)}{e(C_1, d_{k2})\, e(C_2, d_{k2})}, C_5\right) \oplus C_0. \quad (26)$$

4.3. Correctness. The correctness can be checked:

$$\frac{e(d_{k0}, C_4)\, e(d_{k4} d_{k5}, C_3)\, e(d_{k3}, C_4)}{e(C_1, d_{k2})\, e(C_2, d_{k2})} = \frac{e\left(g_2^{\alpha} \left(\prod_{i=1}^{k} h_{in}\right)^{r_1}, v^{s_1}\right) e\left((ux)^{r_1 t_1}, g^{s_2}\right) e\left(w^{r_1}, v^{s_1}\right)}{e\left(\left(\prod_{i=1}^{k} h_{in}\right)^{s_1} u^{s_2}, v^{r_1}\right) e\left(w^{s_1} x^{s_2}, v^{r_1}\right)} = e(g_1, g_2)^{s_1}. \quad (27)$$

5. Analysis

5.1. Security Analysis. Following [59], the security proof can be completed by a series of hybrid games. Let $(C_0, C_1, C_2, C_3, C_4, C_5)$ denote the challenge ciphertext given to the adversary during a $\boldsymbol{Game}_{\text{real}}$. We define the hybrid games to be

(1) ***Game 1***: $C = (C_0, C_1, C_2, C_3, C_4, C_5)$;

(2) ***Game 2***: $C = (R_0, C_1, C_2, C_3, C_4, C_5)$;

(3) ***Game 3***: $C = (R_0, R_1, C_2, C_3, C_4, C_5)$;

(4) ***Game 4***: $C = (R_0, R_1, R_2, C_3, C_4, C_5)$,

where $R_0 \in G_T$ and $R_1, R_2 \in G$.

We will show these games are indistinguishable in the following lemmas.

Lemma 4. *Suppose the decision $(n+1)$-BDHE assumption holds, there is no polynomial-time adversary that can distinguish **Game 1** and **Game 2**.*

Proof. The proof follows from the security of the Boneh-Boyen selective-ID scheme [60] and Abdalla's security analysis [58]. Suppose there is adversary $\mathcal{A}$ that can distinguish between ***Game 1*** and ***Game 2*** with advantage ε. Then challenge $\mathcal{C}$ will be made to solve the decision $(n+1)$-BDHE assumption.

$\mathcal{C}$ receives a challenge tuple $(g, y_0, y_1, \cdots, y_n, y_{n+2}, \cdots, y_{2n+2}, K)$ where $y_0 = g^c$, $y_i = g^{\alpha^i}$, and K is either $e(g,g)^{\alpha^{n+1}c}$ or a random element of G_T. $\mathcal{C}$ interacts with $\mathcal{A}$ as follows:

(i) Init: The adversary $\mathcal{A}$ gives the challenge identity $ID^* = (v_1^*, \ldots, v_k^*)$ to the challenger $\mathcal{C}$ where $k \le l$.

(ii) Setup: $\mathcal{C}$ randomly chooses $t, t_1, t_2, t_3, \gamma, \alpha_{ij}, \beta_{ij} \in Z_p^*$, where $1 \le i \le l$, $1 \le j \le n$. It sets

$$\begin{aligned} v &= g^{t_1}, \\ g_1 &= y_1^{t_1}, \\ u &= g^{t_2}, \\ x &= g^{t_3}, \\ w &= g^t, \\ g_2 &= y_n g^\gamma. \end{aligned} \tag{28}$$

The public key parameters are $PK = \{g, g_1, g_2, v, u, x, w\}$. The master secret key is $MSK = g_2^\alpha = y_1^\gamma y_{n+1}$, which is unknown to $\mathcal{C}$.

(iii) Phase 1: $\mathcal{A}$ adaptively queries $\mathcal{C}$ with $ID = (v_1, \ldots, v_k)$ where $ID \ne ID^*$ and ID is not a prefix of ID^*. This condition ensures that there is $j \in \{1, \ldots, k\}$ such that $v_j \ne v_j^*$. $\mathcal{C}$ produces the private key corresponding to the identity $ID_j = (v_1, \ldots, v_j)$, where j denotes the first element such that $v_j \ne v_j^*$. Let τ be the number of sites such that $v_{ji} = v_{ji}^*$ in v_j. To respond to the query, $\mathcal{C}$ produces the auxiliary parameters as follows. For $1 \le i \le j$, compute

$$\begin{aligned} h_{i1} &= g^{\alpha_{i1}^{v_{i1}} \beta_{i1}^{1-v_{i1}}} \quad if\ v_{i1} \ne v_{i1}^*, \\ h_{i1} &= y_1^{\alpha_{i1}^{v_{i1}} \beta_{i1}^{1-v_{i1}}} \quad if\ v_{i1} = v_{i1}^*. \\ h_{i2} &= h_{i1}^{\alpha_{i2}^{v_{i2}} \beta_{i2}^{1-v_{i2}}} \quad if\ v_{i2} \ne v_{i2}^*, \\ h_{i2} &= y_1^{\alpha_{i2}^{v_{i2}} \beta_{i2}^{1-v_{i2}}} \quad if\ v_{i2} = v_{i2}^* \wedge v_{i1} \ne v_{i1}^*, \\ h_{i2} &= y_2^{\alpha_{i2}^{v_{i2}} \beta_{i2}^{1-v_{i2}}} \quad if\ v_{i2} = v_{i2}^* \wedge v_{i1} = v_{i1}^*. \end{aligned} \tag{29}$$

Finally, the auxiliary parameters can be computed as follows.

$$\begin{aligned} h_{1n} &= y_n^{T(v_1)}, \\ &\cdots, \\ h_{(j-1)n} &= y_n^{T(v_{j-1})}, \\ h_{jn} &= y_\tau^{T(v_j)}, \end{aligned} \tag{30}$$

where $T(v_i) = \prod_{j=1}^n \alpha_{ij}^{v_{ij}} \beta_{ij}^{1-v_{ij}}$, for $1 \le i \le j, \tau \le n$. $\mathcal{C}$ randomly chooses $r_1' \in Z_p$ and sets $r_1 = r_1' - \alpha^{n-\tau+1}/T(v_j)$. The private keys corresponding to the identity ID_j are simulated as follows.

$$\begin{aligned} d_{j0} &= y_1^\gamma \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \left(y_\tau^{T(v_j)}\right)^{r_1'}, \\ d_{j1} &= g^{r_1'} y_{n-\tau+1}^{-1/T(v_j)}, \\ d_{j2} &= g^{r_1' t_1} y_{n-\tau+1}^{-t_1/T(v_j)}, \\ d_{j3} &= g^{r_1' t} y_{n-\tau+1}^{-t/T(v_j)}, \\ d_{j4} &= g^{r_1' t_1 t_2} y_{n-\tau+1}^{-t_1 t_2/T(v_j)}, \\ d_{j5} &= g^{r_1' t_1 t_3} y_{n-\tau+1}^{-t_1 t_3/T(v_j)}. \end{aligned} \tag{31}$$

In fact,

$$\begin{aligned} d_{j0} &= y_1^\gamma \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \left(y_\tau^{T(v_j)}\right)^{r_1'} \\ &= y_{n+1} y_1^\gamma \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \\ &\quad \cdot \left(y_\tau^{T(v_j)}\right)^{r_1'} y_{n+1}^{-1} \\ &= g_2^\alpha \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \\ &\quad \cdot \left(y_\tau^{T(v_j)}\right)^{r_1'} y_{n+1}^{-1} \\ &= g_2^\alpha \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \\ &\quad \cdot \left(y_\tau^{T(v_j)}\right)^{r_1'} \left(y_\tau^{T(v_j)}\right)^{-\alpha^{n-\tau+1}/T(v_j)} \end{aligned}$$

$$= g_2^{\alpha}\left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'}\left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right)$$
$$\cdot\left(y_\tau^{T(v_j)}\right)^{(r_1'-\alpha^{n-\tau+1}/T(v_j))}$$
$$= g_2^{\alpha}\left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{(r_1'-\alpha^{n-\tau+1}/T(v_j))}$$
$$\times\left(y_\tau^{T(v_j)}\right)^{(r_1'-\alpha^{n-\tau+1}/T(v_j))}$$
$$= g_2^{\alpha}\left(\prod_{i=1}^{j} h_{in}\right)^{(r_1'-\alpha^{n-\tau+1}/T(v_j))} = g_2^{\alpha}\left(\prod_{i=1}^{j} h_{in}\right)^{r_1}. \tag{32}$$

In addition,

$$\begin{aligned} d_{j1} &= g^{r_1'} y_{n-\tau+1}^{-1/T(v_j)} = g^{r_1}, \\ d_{j2} &= g^{r_1' t_1} y_{n-\tau+1}^{-t_1/T(v_j)} = v^{r_1}, \\ d_{j3} &= g^{r_1' t} y_{n-\tau+1}^{-t/T(v_j)} = v^{r_1} = w^{r_1}, \\ d_{j4} &= g^{r_1' t_1 t_2} y_{n-\tau+1}^{-t_1 t_2/T(v_j)} = u^{r_1 t_1}, \\ d_{j5} &= g^{r_1' t_1 t_3} y_{n-\tau+1}^{-t_1 t_3/T(v_j)} = x^{r_1 t_1}. \end{aligned} \tag{33}$$

Then we can obtain

$$\begin{aligned} d_{ID_j} &= \left(d_{j0}, d_{j1}, d_{j2}, d_{j3}, d_{j4}, d_{j5}\right) \\ &= \left(g_2^{\alpha}\left(\prod_{i=1}^{j} h_{in}\right)^{r_1}, g^{r_1}, v^{r_1}, w^{r_1}, u^{r_1 t_1}, x^{r_1 t_1}\right). \end{aligned} \tag{34}$$

$\mathscr{C}$ also produces D_{ID_j}. $\mathscr{C}$ uses d_{ID_j} to derive a private key for the descendant identity ID_k and gives $\mathscr{A}$ the result.

Then, $\mathscr{A}$ submits the leak queries to $\mathscr{C}$.

(iv) Challenge: $\mathscr{A}$ selects two messages M_0, M_1 on which it wishes to be challenged. $\mathscr{C}$ first produces the auxiliary parameters as $h_{in} = g^{T_i^*}$ for challenge identity ID^*, where $T_i^* = T(v_i^*)$. $\mathscr{C}$ then randomly chooses $s_1, s_2 \in Z_p$ and a random seed $s \in \{0,1\}^t$ and responds to the ciphertexts as

$$\begin{aligned} CT^* &= \left(C_0^*, C_1^*, C_2^*, C_3^*, C_4^*, C_5^*\right) = \Big(M_b \\ &\oplus Ext\left(K^{t_1} e\left(y_1^{\gamma}, y_0\right)^{t_1}, s\right), y_0^{\sum_{i=1}^{k} T_i^*} g^{s_2 t_2}, y_0^t g^{s_2 t_3}, g^{s_2}, \\ &y_0^{t_1}, s\Big), \end{aligned} \tag{35}$$

where $b \in \{0,1\}$.

(v) Phase 2: $\mathscr{C}$ answers the queries in the same way as phase 1 with the added restriction that the $\mathscr{A}$ can not execute leakage queries.

(vi) Guess: $\mathscr{A}$ outputs a bit $b' \in \{0,1\}$ and wins if $b' = b$.

From the above game, we can see that if $T = e(g,g)^{\alpha^{n+1}c}$, $\mathscr{C}$ is playing ***Game 1***. The challenge ciphertexts are valid encryption to M_b. In fact, let $c = s_1$. Then one can obtain

$$\begin{aligned} C_0^* &= M_b \oplus Ext\left(K^{t_1} e\left(y_1^{\gamma}, y_0\right)^{t_1}, s\right) \\ &= M_b \oplus Ext\left(e\left(g,g\right)^{\alpha^{n+1} c t_1} e\left(y_1^{\gamma}, y_0\right)^{t_1}, s\right) \\ &= M_b \oplus Ext\left(e\left(g^{\alpha^n}, g^{\alpha c}\right)^{t_1} e\left(y_1^{\gamma}, y_0\right)^{t_1}, s\right) \\ &= M_b \oplus Ext\left(e\left(y_n, y_1\right)^{t_1 c} e\left(g^{\gamma}, y_1\right)^{t_1 c}, s\right) \\ &= M_b \oplus Ext\left(e\left(y_n g^{\gamma}, y_1^{t_1}\right)^{c}, s\right) \\ &= M_b \oplus Ext\left(e\left(g_1, g_2\right)^{c}, s\right) \\ &= M_b \oplus Ext\left(e\left(g_1, g_2\right)^{s_1}, s\right), \\ C_1^* &= y_0^{\sum_{i=1}^{k} T_i^*} g^{s_2 t_2} = \left(\prod_{i=1}^{k} h_{in}\right)^{c} u^{s_2} = \left(\prod_{i=1}^{k} h_{in}\right)^{s_1} u^{s_2}, \\ C_2^* &= y_0^t g^{s_2 t_3} = w^c x^{s_2} = w^{s_1} x^{s_2}, \\ C_3^* &= g^{s_2}, \\ C_4^* &= y_0^{t_1} = v^c = v^{s_1}, \\ C_5^* &= s. \end{aligned} \tag{36}$$

Otherwise, K is a random element in G_T; $\mathscr{C}$ is playing ***Game 2***. Thus, ***Game 1*** and ***Game 2*** are computationally indistinguishable. □

Lemma 5. *Suppose the decisional linear assumption holds. Then **Game 2** and **Game 3** are indistinguishable.*

Proof. Suppose there is adversary $\mathscr{A}$ that can distinguish between ***Game 2*** and ***Game 3*** with advantage ε. Then a challenge $\mathscr{C}$ will be made to solve the decision linear problem.

$\mathscr{C}$ receives a challenge tuple

$$\begin{aligned} &\left(g, g^{z_1}, g^{z_2}, g^{z_2^2}, \ldots, g^{z_2^n}, g^{z_2^{n+2}}, \ldots, g^{z_2^{2n}}, g^{z_3}, g^{z_4}, g^{z_4 z_2}, \ldots,\right. \\ &\left. g^{z_2^n z_4}, K\right), \end{aligned} \tag{37}$$

K is either $g^{z_1(z_3+z_4)}$ or a random element of G. $\mathscr{C}$ interacts with $\mathscr{A}$ as follows:

(i) Init: The adversary $\mathscr{A}$ gives the challenge identity $ID^* = (v_1^*, \ldots, v_k^*)$ to the challenger $\mathscr{C}$ where $k \le l$.

(ii) Setup: $\mathscr{C}$ randomly chooses $t, t_1, t_3, \gamma, \alpha_{ij}, \beta_{ij} \in Z_p^*$ where $1 \le i \le l$, $1 \le j \le n$. It sets

$$\begin{aligned} v &= g^{t_1}, \\ g_1 &= g^{t_1 z_2}, \\ u &= g^{z_4 \sum_k^{i=1} T_i^*}, \\ w &= g^{t}, \\ x &= g^{t_3}, \\ g_2 &= g^{z_2^n} g^{\gamma}, \end{aligned} \tag{38}$$

where $T_i^* = T(v_i^*)$. The public key parameters are $PK = \{g, g_1, g_2, v, u, w, x\}$. The master secret key is $MSK = g_2^{z_2} = y_1^{\gamma} g^{z_2^{n+1}}$ which is unknown to $\mathscr{C}$, where $y_i = g^{z_2^i}$.

(iii) Phase 1: $\mathscr{A}$ adaptively queries $\mathscr{C}$ with $ID = (v_1, \ldots, v_k)$ where $ID \neq ID^*$ and ID is not a prefix of ID^*. This condition ensures that there is $j \in \{1, \ldots, k\}$ such that $v_j \neq v_j^*$. $\mathscr{C}$ produces the private key corresponding to the identity $ID_j = (v_1, \ldots, v_j)$ where j denotes the first element such that $v_j \neq v_j^*$. Let τ be the number of sites such that $v_{ji} = v_{ji}^*$ in v_j. To respond to the query, $\mathscr{C}$ produces the auxiliary parameters as follows. For $1 \le i \le j$,

$$\begin{aligned} h_{i1} &= g^{\alpha_{i1}^{v_{i1}} \beta_{i1}^{1-v_{i1}}} \quad if\ v_{i1} \neq v_{i1}^*, \\ h_{i1} &= \left(g^{z_2}\right)^{\alpha_{i1}^{v_{i1}} \beta_{i1}^{1-v_{i1}}} \quad if\ v_{i1} = v_{i1}^*. \\ h_{i2} &= h_{i1}^{\alpha_{i2}^{v_{i2}} \beta_{i2}^{1-v_{i2}}} \quad if\ v_{i2} \neq v_{i2}^*, \\ h_{i2} &= \left(g^{z_2}\right)^{\alpha_{i2}^{v_{i2}} \beta_{i2}^{1-v_{i2}}} \quad if\ v_{i2} = v_{i2}^* \cap v_{i1} \neq v_{i1}^*, \\ h_{i2} &= \left(g^{z_2^2}\right)^{\alpha_{i2}^{v_{i2}} \beta_{i2}^{1-v_{i2}}} \quad if\ v_{i2} = v_{i2}^* \cap v_{i1} = v_{i1}^*. \end{aligned} \tag{39}$$

Finally, the auxiliary information parameters can be computed as follows.

$$\begin{aligned} h_{1n} &= y_n^{T(v_1)}, \\ &\ldots, \\ h_{(j-1)n} &= y_n^{T(v_{j-1})}, \\ h_{jn} &= y_\tau^{T(v_j)}, \end{aligned} \tag{40}$$

where

$$\begin{aligned} T(v_i) &= \prod_{j=1}^{n} \alpha_{ij}^{v_{ij}} \beta_{ij}^{1-v_{ij}}, \\ y_i &= g^{z_2^i}, \\ &1 \le i \le j,\ \tau \le n. \end{aligned} \tag{41}$$

$\mathscr{C}$ randomly chooses $r_1' \in Z_p$ and sets $r_1 = r_1' - z_2^{n-\tau+1}/T(v_j)$. The private keys corresponding to the identity ID_j are simulated as follows.

$$\begin{aligned} d_{j0} &= y_1^{\gamma} \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \left(y_\tau^{T(v_j)}\right)^{r_1'}, \\ d_{j1} &= g^{r_1'} y_{n-\tau+1}^{-1/T(v_j)}, \\ d_{j2} &= g^{r_1' t_1} y_{n-\tau+1}^{-t_1/T(v_j)}, \\ d_{j3} &= g^{r_1' t} y_{n-\tau+1}^{-t/T(v_j)}, \\ d_{j4} &= u^{r_1' t_1} y_{n-\tau+1}^{-t_1 z_4 \sum_{i=1}^{k} T_i^*/T(v_j)}, \\ d_{j5} &= x^{r_1' t_1} y_{n-\tau+1}^{-t_1 t_3/T(v_j)}. \end{aligned} \tag{42}$$

In fact,

$$\begin{aligned} d_{j0} &= y_1^{\gamma} \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \left(y_\tau^{T(v_j)}\right)^{r_1'} \\ &= y_{n+1} y_1^{\gamma} \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \\ &\quad \cdot \left(y_\tau^{T(v_j)}\right)^{r_1'} y_{n+1}^{-1} \\ &= g_2^{z_2} \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \\ &\quad \cdot \left(y_\tau^{T(v_j)}\right)^{r_1'} y_{n+1}^{-1} \\ &= g_2^{\alpha} \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \\ &\quad \cdot \left(y_\tau^{T(v_j)}\right)^{r_1'} \left(y_\tau^{T(v_j)}\right)^{-z_2^{n-\tau+1}/T(v_j)} \\ &= g_2^{\alpha} \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{r_1'} \left(\prod_{i=1}^{j-1} y_{2n-\tau+1}^{-T(v_i)/T(v_j)}\right) \\ &\quad \cdot \left(y_\tau^{T(v_j)}\right)^{(r_1' - z_2^{n-\tau+1}/T(v_j))} \\ &= g_2^{\alpha} \left(\prod_{i=1}^{j-1} y_n^{T(v_i)}\right)^{(r_1' - z_2^{n-\tau+1}/T(v_j))} \\ &\quad \times \left(y_\tau^{T(v_j)}\right)^{(r_1' - z_2^{n-\tau+1}/T(v_j))} \\ &= g_2^{\alpha} \left(\prod_{i=1}^{j} h_{in}\right)^{(r_1' - z_2^{n-\tau+1}/T(v_j))} = g_2^{\alpha} \left(\prod_{i=1}^{j} h_{in}\right)^{r_1}. \end{aligned} \tag{43}$$

In addition,

$$\begin{aligned} d_{j1} &= g^{r_1'} y_{n-\tau+1}^{-1/T(v_j)} = g^{r_1}, \\ d_{j2} &= g^{r_1' t_1} y_{n-\tau+1}^{-t_1/T(v_j)} = v^{r_1}, \\ d_{j3} &= g^{r_1' t} y_{n-\tau+1}^{-t/T(v_j)} = v^{r_1} = w^{r_1}, \\ d_{j4} &= u^{r_1' t_1} y_{n-\tau+1}^{-t_1 z_4 \sum_{i=1}^{k} T_i^*/T(v_j)} = u^{r_1 t_1}, \\ d_{j5} &= x^{r_1' t_1} y_{n-\tau+1}^{-t_1 t_3/T(v_j)} = x^{r_1 t_1}. \end{aligned} \tag{44}$$

Then we can obtain

$$\begin{aligned} d_{ID_j} &= \left(d_{j0}, d_{j1}, d_{j2}, d_{j3}, d_{j4}, d_{j5}\right) \\ &= \left(g_2^{\alpha} \left(\prod_{i=1}^{j} h_{in}\right)^{r_1}, g^{r_1}, v^{r_1}, w^{r_1}, u^{r_1 t_1}, x^{r_1 t_1}\right). \end{aligned} \tag{45}$$

$\mathscr{C}$ also produces D_{ID_j}. $\mathscr{C}$ uses d_{ID_j} to derive a private key for the descendant identity ID_k and gives $\mathscr{A}$ the result.

Then, $\mathscr{A}$ submits the leak queries to $\mathscr{C}$.

(iv) Challenge: $\mathscr{A}$ selects two messages M_0, M_1 on which it wishes to be challenged. $\mathscr{C}$ first produces the auxiliary parameters as $h_{in} = g^{z_1 T_i^*}$ for challenge identity ID^*, where $T_i^* = T(v_i^*)$. $\mathscr{C}$ then randomly chooses $R_0 \in G_T$ and a random seed $s \in \{0,1\}^t$ and responds to the ciphertexts as

$$\begin{aligned} CT^* &= \left(C_0^*, C_1^*, C_2^*, C_3^*, C_4^*, C_5^*\right) \\ &= \left(R_0, K^{\sum_{i=1}^{n} T_i^*}, g^{z_3 t} g^{z_1 z_3}, g^{z_1}, g^{t_1 z_3}, s\right). \end{aligned} \tag{46}$$

(v) Phase 2: $\mathscr{C}$ answers the queries in the same way as phase 1 with the added restriction that $\mathscr{A}$ can not execute leakage queries.

(vi) Guess: $\mathscr{A}$ outputs a bit $b' \in \{0,1\}$ and wins if $b' = b$.

From the above game, we can see that if $K = g^{z_1(z_3+z_4)}$, $\mathscr{C}$ is playing ***Game 2***. In fact, let $z_3 = s_1, z_1 = s_2$. Then one can obtain

$$\begin{aligned} C_1^* &= K^{\sum_{i=1}^{n} T_i^*} = \left(g^{z_1(z_3+z_4)}\right)^{\sum_{i=1}^{n} T_i^*} \\ &= \left(\left(g^{z_1}\right)^{\sum_{i=1}^{n} T_i^*}\right)^{z_3} \left(\left(g^{z_4}\right)^{\sum_{i=1}^{k} T_i^*}\right)^{z_1} \\ &= \left(\prod_{i=1}^{k} h_{in}\right)^{s_1} u^{s_2}, \\ C_2^* &= g^{z_3 t} g^{z_1 z_3} = w^{s_1} x^{s_2}, \\ C_3^* &= g^{z_1} = g^{s_2}, \\ C_4^* &= g^{t_1 z_3} = v^{s_1}, \\ C_5^* &= s. \end{aligned} \tag{47}$$

Otherwise, K is a random element in G, and $\mathscr{C}$ is playing ***Game 3***. Thus, ***Game 2*** and ***Game 3*** are computationally indistinguishable. □

Lemma 6. *Suppose the decisional linear assumption holds. Then **Game 3** and **Game 4** are indistinguishable.*

Proof. This proof is similar to Lemma 5. $\mathscr{C}$ receives a challenge tuple

$$\begin{aligned} &\Big(g, g^{z_1}, g^{z_2}, g^{z_2^2}, \ldots, g^{z_2^n}, g^{z_2^{n+2}}, \ldots, g^{z_2^{2n}}, g^{z_3}, g^{z_4}, g^{z_4 z_2}, \ldots, \\ &\quad g^{z_2^n z_4}, g^{z_1 z_2}, \ldots, g^{z_2^n z_1}, K\Big), \end{aligned} \tag{48}$$

K is either $g^{z_1(z_3+z_4)}$ or a random element of G. $\mathscr{C}$ interacts with $\mathscr{A}$ as follows:

(i) Init: adversary $\mathscr{A}$ gives the challenge identity $ID^* = (v_1^*, \ldots, v_k^*)$ to the challenger $\mathscr{C}$ where $k \le l$.

(ii) Setup: $\mathscr{C}$ randomly chooses $t, t_1, t_2, \gamma, \alpha_{ij}, \beta_{ij} \in Z_p^*$ where $1 \le i \le l$, $1 \le j \le n$. It sets

$$\begin{aligned} v &= g^{t_1}, \\ g_1 &= g^{t_1 z_2}, \\ u &= g^{t_2}, \\ w &= g^{t z_1}, \\ x &= g^{t z_4}, \\ g_2 &= g^{z_2^n} g^{\gamma}. \end{aligned} \tag{49}$$

The public key parameters are $PK = \{g, g_1, g_2, v, u, w, x\}$. The master secret key is $MSK = g_2^{z_2} = y_1^{\gamma} g^{z_2^{n+1}}$ which is unknown to $\mathscr{C}$, where $y_i = g^{z_2^i}$.

(iii) Phase 1: this section is similar to Lemma 5.

(iv) Challenge: $\mathscr{A}$ selects two messages M_0, M_1 on which it wishes to be challenged. $\mathscr{C}$ randomly chooses $R_0 \in G_T$, $R_1 \in G$, and a random seed $s \in \{0,1\}^t$ and responds to the ciphertexts as

$$\begin{aligned} CT^* &= \left(C_0^*, C_1^*, C_2^*, C_3^*, C_4^*, C_5^*\right) \\ &= \left(R_0, R_1, K^t, g^{z_1}, g^{t_1 z_3}, s\right). \end{aligned} \tag{50}$$

(v) Phase 2: $\mathscr{C}$ answers the queries in the same way as phase 1 with the added restriction that $\mathscr{A}$ can not execute leakage queries.

(vi) Guess: $\mathscr{A}$ outputs a bit $b' \in \{0,1\}$ and wins if $b' = b$.

From the above game, we can see that if $K = g^{z_1(z_3+z_4)}$, $\mathscr{C}$ is playing ***Game 3***. In fact, let $z_3 = s_1, z_1 = s_2$. Then one can obtain

$$\begin{aligned} C_2^* &= K^t = \left(g^{z_1(z_3+z_4)}\right)^t = g^{tz_1z_3}g^{tz_1z_4} = w^{s_1}x^{s_2}, \\ C_3^* &= g^{z_1} = g^{s_2}, \\ C_4^* &= g^{t_1z_3} = v^{s_1}, \\ C_5^* &= s. \end{aligned} \tag{51}$$

Otherwise, K is a random element in G, and $\mathscr{C}$ is playing ***Game 4***. Thus, ***Game 3*** and ***Game 4*** are computationally indistinguishable. □

Theorem 7. *If the decision $(n+1)$-BDHE assumption and the decisional linear assumption hold, then our anonymous HIBE scheme is $\ell(\lambda)$-leakage resilience secure.*

5.2. Leakage Resilience Analysis. Let V be used to represent a view that adversary $\mathscr{A}$ sees without leakage, we have $\widetilde{H}_\infty(\mathscr{A} \mid V) = \log p$. From the above leakage-resilient security model, one can know that, for the challenge identity ID^*, the adversary can get the most $\ell(\lambda)$ leak information when doing leak queries. We set $\ell(\lambda)$ as ξ bit; in other words, $\ell(\lambda)$ has 2^ξ values. From Lemma 2, one can obtain

$$\widetilde{H}_\infty(\mathscr{A} \mid (\ell(\lambda), V)) \geq \widetilde{H}_\infty(\mathscr{A} \mid V) - \xi = \log p - \xi. \tag{52}$$

Therefore, as long as the extractor's strength is $(\log p - \xi, \varepsilon)$, one can obtain

$$\begin{aligned} &SD\left(\left(Ext\left(e\left(g_1, g_2\right)^{s_1}, s\right), s, \ell(\lambda), V\right),\right. \\ &\left.(U, s, \ell(\lambda), V)\right) \leq \varepsilon, \end{aligned} \tag{53}$$

where U is uniform distribution and $s \in \{0,1\}^t$. The premise is that extractor performance is good enough; one can obtain $\xi \approx \log p$. Hence the distance between $C_0 = M_b \oplus Ext(e(g_1, g_2)^{s_1}, s)$ and uniform distribution is ε and the statistical distance between two ciphertexts is at most 2ε. Therefore, no PPT adversary can distinguish the two challenge ciphertexts with the advantage of more than 2ε. The leakage ratio of our scheme is

$$\rho = \frac{\ell(\lambda)}{(6\log p)} = \frac{\xi}{(6\log p)} \approx \frac{\log p}{(6\log p)} = \frac{1}{6}. \tag{54}$$

We compare our work with schemes [24, 35, 54] in Table 2, where *pk.size*, *sk.size*, *c.size*, and *LR* mean the public key size, the secret key size, the ciphertext size, and leakage-resilience, respectively. We define l as the maximum depth of HIBE, k represents the user identity depth, and the symbol "-" represents the fact that the corresponding scheme does not have this feature. It is noted that our scheme supports anonymity and leakage-resilience where both private keys and ciphertext have a constant size.

Our scheme has no obvious advantage in computational efficiency, but the public key length and private key length of our HIBE scheme are both in $O(1)$ time. However, we mainly solve the problem of key leakage in the acceptable range between computational efficiency and security balance.

Table 2: Comparisons of different schemes.

Scheme	*pk.size*	*sk.size*	*c.size*	Anonymity	LR
[35]	$O(l)$	$O(l)$	$O(1)$	√	-
[24]	$O(1)$	$O(1)$	$O(1)$	√	-
[55]	$O(1)$	$O(1)$	$O(1)$	-	√
Ours	$O(1)$	$O(1)$	$O(1)$	√	√

6. Conclusion and Future Work

In this paper, we present a secure s-health system based on a leakage-resilient anonymous HIBE scheme in the bounded-leakage model. Our scheme can protect the patient's privacy well, even when the private key is partially leaked. Our system also achieves the safe transmission of the patient's EHRs in the case of leakage attacks. And we provide an example to show the systems feasibility. The proposed scheme was proved to be secure against chosen plaintext attacks in the standard model under the Diffie-Hellman exponent assumption and decisional linear assumption. However, the doctors may reveal the privacy of patients in a malicious way; thus, in the future work, we can increase tracking technology to limit doctors' malicious information disclosure. In the future work, we can also study how to construct a secure s-health system which allows the master-key leakage.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by National Key R&D Program of China (no. 2017YFB0802000), National Natural Science Foundation of China (nos. 61772418, 61472472, and 61402366), and Natural Science Basic Research Plan in Shaanxi Province of China (nos. 2018JZ6001, 2015JQ6236, 2016JM6033, and 2015JQ6262). Yinghui Zhang is supported by New Star Team of Xi'an University of Posts and Telecommunications.

References

[1] J. Li, L. Wang, and Z. Huang, "Verifiable Chebyshev Maps-Based Chaotic Encryption Schemes with Outsourcing Computations in the Cloud/Fog Scenarios," *Concurrency and Computation: Practice and Experience*, 2018.

[2] L. Sun, Z. Li, Q. Yan, W. Srisa-An, and Y. Pan, "Sigpid: significant permission identification for android malware detection," in *Proceedings of the International Conference on Malicious and Unwanted Software*, pp. 1–8, 2017.

[3] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An ID-based linearly homomorphic signature scheme and its

application in blockchain," *IEEE Access*, vol. PP, no. 99, pp. 1-1, 2018.

[4] T. Li, J. Li, Z. Liu, P. Li, and C. Jia, "Differentially private Naive Bayes learning over multiple data sources," *Information Sciences*, vol. 444, pp. 89–104, 2018.

[5] J. Li, Z. Liu, X. Chen, F. Xhafa, X. Tan, and D. S. Wong, "L-EncDB: A lightweight framework for privacy-preserving data queries in cloud computing," *Knowledge-Based Systems*, vol. 79, pp. 18–26, 2015.

[6] Q. Lin, J. Li, Z. Huang, W. Chen, and J. Shen, "A short linearly homomorphic proxy signature scheme," *IEEE Access*, vol. 6, pp. 12966–12972, 2018.

[7] R. Xie, C. He, D. Xie, C. Gao, and X. Zhang, "A Secure Ciphertext Retrieval Scheme against Insider KGAs for Mobile Devices in Cloud," *Security and Communication Networks*, vol. 2018.

[8] Y. Zhang, J. Li, X. Chen, and H. Li, "Anonymous attribute-based proxy re-encryption for access control in cloud computing," *Security and Communication Networks*, vol. 9, no. 14, pp. 2397–2411, 2016.

[9] C. Yuan, X. Li, Q. J. Wu, J. Li, and X. Sun, "Fingerprint liveness detection from different fingerprint materials using convolutional neural network and principal component analysis," *Computers Materials & Continua*, vol. 53, no. 4, pp. 357–371, 2015.

[10] J. Li, X. Chen, S. S. M. Chow, Q. Huang, D. S. Wong, and Z. Liu, "Multi-authority fine-grained access control with accountability and its application in cloud," *Journal of Network Computer Applications*, 2018.

[11] Y. Zhang, D. Zheng, and R. H. Deng, "Security and Privacy in Smart Health: Efficient Policy-Hiding Attribute-Based Access Control," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2130–2145, 2018.

[12] P. Li, J. Li, Z. Huang et al., "Multi-key privacy-preserving deep learning in cloud computing," *Future Generation Computer Systems*, vol. 74, pp. 76–85, 2017.

[13] Y. Zhang, X. Chen, J. Li, D. S. Wong, and H. Li, "Anonymous attribute-based encryption supporting efficient decryption test," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, pp. 511–516, ACM, 2013.

[14] J. Li, X. Chen, M. Li, J. Li, P. P. C. Lee, and W. Lou, "Secure deduplication with efficient and reliable convergent key management," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 6, pp. 1615–1625, 2014.

[15] P. Li, J. Li, Z. Huang, C.-Z. Gao, W.-B. Chen, and K. Chen, "Privacy-preserving outsourced classification in cloud computing," *Cluster Computing*, no. 1, pp. 1–10, 2017.

[16] Y. Dodis and K. Pietrzak, *Leakage-Resilient Pseudorandom Functions and Side-Channel Attacks on Feistel Networks*, Springer, Berlin, Heidelberg, Germany, 2010.

[17] J. A. Halderman, S. D. Schoen, N. Heninger et al., "Lest we remember: cold-boot attacks on encryption keys," *Communications of the ACM*, vol. 52, no. 5, pp. 91–98, 2008.

[18] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems," *International Cryptology Conference on Advances in Cryptology*, pp. 104–113, 1996.

[19] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic analysis: Concrete results," *Ches May*, vol. 2162, pp. 251–261, 2001.

[20] D. Boneh, R. A. DeMillo, and R. J. Lipton, *On the Importance of Checking Cryptographic Protocols for Faults*, Springer, Berlin, Heidelberg, Germany, 1997.

[21] Y. H. Zhang, X. F. Chen, H. Li, and J. Cao, "Identity-based construction for secure and efficient handoff authentication schemes in wireless networks," *Security and Communication Networks*, vol. 5, no. 10, pp. 1121–1130, 2012.

[22] J. Li, X. Chen, X. Huang et al., "Secure distributed deduplication systems with improved reliability," *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 3569–3579, 2015.

[23] C. Gao, Q. Cheng, P. He, W. Susilo, and J. Li, "Privacy-preserving Naive Bayes classifiers secure against the substitution-then-comparison attack," *Information Sciences*, 2018.

[24] C.-I. Fan, L.-Y. Huang, and P.-H. Ho, "Compact Anonymous Hierarchical Identity-Based Encryption with Constant Size Private Keys," *Computer Journal*, vol. 59, no. 4, pp. 452–461, 2018.

[25] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure attribute-based data sharing for resource-limited users in cloud computing," *Computers & Security*, vol. 72, pp. 1–12, 2018.

[26] Y. Zhang, D. Zheng, X. Chen, J. Li, and H. Li, "Efficient attribute-based data sharing in mobile clouds," *Pervasive and Mobile Computing*, vol. 28, pp. 135–149, 2016.

[27] Y. Zhang, X. Chen, J. Li, D. S. Wong, H. Li, and I. You, "Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing," *Information Sciences*, vol. 379, pp. 42–61, 2017.

[28] A. Shamir, *Identity-based cryptosystems and signature schemes*, vol. 21 of *Lecture Notes in Computer Science*, 2 edition, 1984.

[29] J. Li, J. Li, X. Chen, C. Jia, and W. Lou, "Identity-based encryption with outsourced revocation in cloud computing," *IEEE Transactions on computers*, vol. 64, no. 2, pp. 425–437, 2015.

[30] C. Gentry and A. Silverberg, "Hierarchical id-based cryptography," in *Proceedings of the Advances in Cryptology - ASIACRYPT 2002, International Conference on the Theory and Application of Cryptology and Information Security*, pp. 548–566, Queenstown, New Zealand, December, 2002.

[31] D. Boneh, X. Boyen, and E.-J. Goh, "Hierarchical identity based encryption with constant size ciphertext," in *Proceedings of the International Conference on Theory and Applications of Cryptographic Techniques*, pp. 440–456, 2005.

[32] B. Waters, "Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions," *International Cryptology Conference on Advances in Cryptology*, vol. 5677, pp. 619–636, 2009.

[33] J. H. Seo, T. Kobayashi, M. Ohkubo, and K. Suzuki, "Anonymous hierarchical identity-based encryption with constant size ciphertexts," in *Proceedings of the International Conference on Practice and Theory in Public Key Cryptography*, vol. 5443, pp. 215–234, PKC, 2009.

[34] J. H. Seo and J. H. Cheon, "Fully secure anonymous hierarchical identity-based encryption with constant size ciphertexts," *Iacr Cryptology Eprint Archive*, vol. 2011, no. 2011, pp. 215–234, 2011.

[35] J. H. Park and D. H. Lee, "Anonymous hibe: Compact construction over prime-order groups," *IEEE Transactions on Information Theory*, vol. 59, no. 4, pp. 2531–2541, 2013.

[36] S. C. Ramanna and P. Sarkar, *Anonymous Constant-Size Ciphertext HIBE from Asymmetric Pairings*, Springer, Berlin, Heidelberg, Germany, 2013.

[37] F. W. Dillema and S. Lupetti, "Rendezvous-based access control for medical records in the pre-hospital environment," in *Proceedings of the 1st ACM SIGMOBILE international workshop on Systems and networking support for healthcare and assisted living environments*, pp. 1–6, ACM, 2007.

[38] R. Zhang and L. Liu, "Security models and requirements for healthcare application clouds," in *Proceedings of the Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pp. 268–275, IEEE, 2010.

[39] J. Sun, X. Zhu, C. Zhang, and Y. Fang, "Hcpp: Cryptography based secure ehr system for patient privacy and emergency healthcare," in *Proceedings of the Distributed Computing Systems (ICDCS), 2011 31st International Conference on*, pp. 373–382, 2011.

[40] L. Guo, C. Zhang, J. Sun, and Y. Fang, "PAAS: A Privacy-Preserving Attribute-Based Authentication System for eHealth Networks," in *Proceedings of the IEEE International Conference on Distributed Computing Systems*, pp. 224–233, 2012.

[41] L. Guo, C. Zhang, J. Sun, and Y. Fang, "A Privacy-Preserving Attribute-Based Authentication System for Mobile Health Networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 9, pp. 1927–1941, 2014.

[42] T. Kumar, A. Braeken, M. Liyanage, and M. Ylianttila, "Identity privacy preserving biometric based authentication scheme for Naked healthcare environment," *IEEE International Conference on Communications*, 2017.

[43] A. Sudarsono, M. Yuliana, and H. A. Darwito, "A secure data sharing using identity-based encryption scheme for e-healthcare system," in *Proceedings of the International Conference on Science in Information Technology*, pp. 429–434, 2017.

[44] Y. Zhang, J. Li, D. Zheng, X. Chen, and H. Li, "Towards privacy protection and malicious behavior traceability in smart health," *Personal & Ubiquitous Computing*, vol. 21, no. 8, pp. 1–16, 2017.

[45] M. Dawoud and D. T. Altilar, "Cloud-based E-health systems: Security and privacy challenges and solutions," in *Proceedings of the International Conference on Computer Science and Engineering*, pp. 861–865, 2017.

[46] L. Zhang, Y. Zhang, S. Tang, and H. Luo, "Privacy protection for e-health systems by means of dynamic authentication and three-factor key agreement," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 3, pp. 2795–2805, 2017.

[47] M. A. Sahi, H. Abbas, K. Saleem et al., "Privacy preservation in e-healthcare environments: State of the art and future directions," *IEEE Access*, vol. 6, pp. 464–478, 2018.

[48] S. Dziembowski and K. Pietrzak, "Leakage-resilient cryptography," *IEEE Symposium on Foundations of Computer Science*, pp. 293–302, 2008.

[49] A. Akavia, S. Goldwasser, and V. Vaikuntanathan, *Simultaneous hardcore bits and cryptography against memory attacks*, vol. 5444, Springer, Berlin, Heidelberg, Germany, 2009.

[50] M. Naor and G. Segev, "Public-key cryptosystems resilient to key leakage," *International Cryptology Conference on Advances in Cryptology*, vol. 5677, pp. 18–35, 2009.

[51] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Walfish, and D. Wichs, "Public-key encryption in the bounded-retrieval model," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 113–134, 2010.

[52] S. S. M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, "Practical leakage-resilient identity-based encryption from simple assumptions," in *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 152–161, 2010.

[53] S. Li, F. Zhang, Y. Sun, and L. Shen, "Efficient leakage-resilient public key encryption from ddh assumption," *Cluster Computing*, vol. 16, no. 4, pp. 797–806, 2013.

[54] P. Liu, C. Hu, S. Guo, and Y. Wang, "Anonymous identity-based encryption with bounded leakage resilience," in *Proceedings of the IEEE International Conference on Advanced Information NETWORKING and Applications Workshops*, pp. 287–292, 2015.

[55] J. Li, M. Teng, Y. Zhang, and Q. Yu, "A leakage-resilient cca-secure identity-based encryption scheme," *Computer Journal*, vol. 59, no. 7, pp. 1066–1075, 2016.

[56] J. Quisquater and D. Samyde, *ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards*, Springer, Berlin, Heidelberg, Germany, 2001.

[57] D. M. Freeman, *Converting pairing-based cryptosystems from composite-order groups to prime-order groups*, vol. 2009 of *Lecture Notes in Computer Science*, Springer, Berlin, 2010.

[58] M. Abdalla, D. Catalano, and D. Fiore, "Verifiable random functions from identity-based key encapsulation," *EUROCRYPT*, pp. 554–571, 2009.

[59] X. Boyen and B. Waters, *Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles)*, Springer, Berlin, Heidelberg, Germany, 2006.

[60] D. Boneh and X. Boyen, "Efficient selective-id secure identity-based encryption without random oracles," *Journal of Cryptology*, no. 4, p. 172, 2004.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 6280768, 12 pages
https://doi.org/10.1155/2018/6280768

Research Article

Static and Dynamic Analysis of Android Malware and Goodware Written with Unity Framework

Jaewoo Shim,[1] **Kyeonghwan Lim**,[1] **Seong-je Cho**,[1] **Sangchul Han**,[2] **and Minkyu Park**[2]

[1]*Department of Computer Science and Engineering, Dankook University, Yongin 16890, Republic of Korea*
[2]*Department of Computer Engineering, Konkuk University, Chungju 27478, Republic of Korea*

Correspondence should be addressed to Minkyu Park; minkyup@kku.ac.kr

Received 23 February 2018; Accepted 15 May 2018; Published 20 June 2018

Academic Editor: Karl Andersson

Unity is the most popular cross-platform development framework to develop games for multiple platforms such as Android, iOS, and Windows Mobile. While Unity developers can easily develop mobile apps for multiple platforms, adversaries can also easily build malicious apps based on the "*write once, run anywhere*" (WORA) feature. Even though malicious apps were discovered among *Android apps written with Unity framework (Unity apps)*, little research has been done on analysing the malicious apps. We propose static and dynamic reverse engineering techniques for malicious Unity apps. We first inspect the executable file format of a Unity app and present an effective static analysis technique of the Unity app. Then, we also propose a systematic technique to analyse dynamically the Unity app. Using the proposed techniques, the malware analyst can statically and dynamically analyse Java code, native code in C or C ++, and the Mono runtime layer where the C# code is running.

1. Introduction

The smartphone market is currently dominated by various mobile platforms such as Android, iOS, Windows Phone, and BlackBerry OS. Because these mobile platforms use different development frameworks, app developers who want to cover many users need to develop their native apps for each platform. A *native app* is an application built to run only on a certain mobile platform. Developing an app for each platform has disadvantages. Developers need to know in-depth knowledge of each platform, dedicated software development kits (SDKs), and different programming language. Developers cannot reuse the source codes of one platform for another platform and must write codes multiple times. The development and maintenance cost increases.

Several *cross-platform development frameworks* [1–4] have emerged to tackle these challenges. These development frameworks employ the "*write code once, run on multiple platforms*" feature: the frameworks translate codes for one platform to another platform and provide rum-time library supporting app execution. The frameworks reduce development and maintenance costs. Such frameworks include *Unity*, *Xamarin*, *Appcelerator Titanium*, *PhoneGap*, *Sencha*, and *Cocos2d*.

Among the frameworks, Unity (commonly known as Unity3D) is a very popular cross-platform framework for mobile app development [5–8]. According to the report from Unity Technologies [9], Unity currently supports 28 different platforms and touches 770 million gamers all over the world through games made using the engine. In Q1 2016, 34% of top 1000 free mobile games are developed and around 2.4 billion unique mobile devices have been running Unity-made games.

Adversaries also utilise the frameworks to make their malware spreadable easily while minimizing malware development cost and time. Lee et al. of ShophosLabs [10] reported that more and more malicious apps are being developed with the frameworks. They collected Android malware statistics for a couple of months in 2015 and observed that the number of malware and potentially unwanted app (PUA) samples written with Unity was 32 and 1351, respectively. PUA may pose high risk or have unwanted impact on user security and privacy. Also, according to the study performed by Kaspersky Lab, malware writers are eager to use Microsoft's .NET and

high-level programming languages such as C# to create and modify their malware more rapidly [11]. Adopting such convenient frameworks or tools, malware writers can gain the additional benefit of leveraging already available source code from the underground scene.

Each framework uses its own executable format and loads many libraries necessary for app execution. These are utilised to hide malicious payloads in the package. The malware developers can hide malicious codes in components (e.g., libraries) loaded by the framework as well as the platform's native codes (e.g., Java or Swift). Therefore, we need to analyse framework's components as well as native codes to detect malware [10].

We propose an effective technique to statically and dynamically analyse suspicious Android apps written in C# on Unity framework, which allows for (1) detection of instances of known malware and (2) early detection of known buggy or vulnerable code. We define an Android app written in C# on Unity framework as a Unity app or a Unity APK. We focus on analysing Unity apps which do not have debugging symbol and cannot be easily debugged. We first generate debugging symbols for the target app; repackage the app with the generated symbols; and run and debug it by attaching SDB (a client for the Mono soft debugger).

The rest of this paper is organised as follows. Section 2 explains background and related research work. Section 3 proposes an effective technique for statically analysing Unity apps and presents some results of the static analysis of a Unity app focusing on its initialisation to show the structure and interaction between object codes. Section 4 describes a dynamic analysis technique of a Unity app built in release mode which does not originally have debugging symbols. Section 5 discusses the proposed technique including the differences between native apps and Unity apps in terms of programming languages, library, and debugging tools and protocol. Conclusions are presented in Section 6.

2. Background and Related Work

2.1. Background. The Unity framework is built on Mono runtime which enables developers to use C# or UnityScript [7, 10–12]. The Mono project is an open source implementation of Microsoft's .NET Framework. The .NET Framework consists of three components: a set of supported programming languages, a base class library which implements all basic operations involved in software development, and the *Common Language Runtime* (CLR), which is the core of the framework [10].

CLR is also known as the Mono runtime and we use the terms CLR and Mono runtime interchangeably. CLR is designed to comply with a *common language infrastructure* (CLI), which is a specification for the format of executable code and the runtime environment that can execute that code. CLI is a platform-independent development framework that enables programs written in different languages to run on different types of hardware. No matter which language they are written in, CLI applications are compiled into an intermediate language (IL), which is further compiled into the target machine language by the CLR.

Unity apps are primarily developed using C# and Unity JavaScript (UnityScript). C# source codes are converted to a *Common Intermediate Language* (CIL) which is formerly known as *Microsoft Intermediate Language* (MSIL). CIL code is stored in the form of a DLL file. CIL code is assembled into CLI assembly code and this code will be translated to machine code at runtime using just-in-time (JIT) compilation by the CLR. Unity apps execution can be summarised as follows: (1) source code is converted to CIL, (2) CIL is assembled into a CLI assembly code, similar to Java's bytecode, (3) the CLI assembly code is compiled to machine code for a specific platform, and (4) the processor executes the machine code.

The Mono runtime provides various services such as code execution, garbage collection, code generation using JIT compilation and backend engine, operating system interface, program isolation using Application Domains (AppDomain), thread management, console access, and security system [12]. The Mono runtime also interacts with the existing Android execution environment, DVM (Dalvik Virtual Machine) or ART (Android Runtime).

2.2. Related Work. For program understanding or malware detection, many studies have been conducted on static or dynamic analysis techniques for various types of executable files (DEX, ELF, PE, etc.) on several processors and operating systems (Android, Linux, Microsoft Windows OSes, etc.). Static analysis examines the code of executables to determine control or data flows and certain code patterns of these executables without running them [13–18]. Static analysis can cover the complete program code because it is not bound to a specific execution of an executable and can give guarantees that apply to all execution of the executable. However, static analysis has limitations that it cannot effectively deal with deliberately crafted code with obfuscation and encryption schemes [19–21].

In contrast to static techniques, dynamic analysis monitors the execution of an executable by inspecting runtime behaviours that correspond to selected test cases [21–26]. Dynamic analysis can examine the behaviour of a suspicious program by executing the program in a restricted environment and observing its actions. This dynamic technique can handle packed executables and self-modifying programs and is even immune to code obfuscation attempts to obstruct analysis.

Some researchers have tried to detect malicious Android native apps which use evasion techniques such as various code obfuscations to avoid detection by anti-virus software [27–29]. As a result, several interesting studies [13–29] have been made of static and dynamic techniques for analysing native applications (PE, ELF, JavaScript, etc.) for Microsoft Windows and Linux, or Android malware (DEX), not Unity apps. However, there have been few studies on systematic analysis on Android apps written in C# with Unity.

As the cross-platform development frameworks become more and more popular [30–32], researchers are studying which frameworks and tools are efficient for developing

their mobile apps. Majchrzak et al. [31] performed a comprehensive analysis of the three approaches, React Native, the Ionic Framework, and Fuse using a real-world use case. The analysed results showed that there was no clear winner. Latif et al. [32] conducted an exhaustive survey of cross-platform approaches and tools, provided an overview of recent tools and platforms for each approach, and discussed the advantages and disadvantages of each one.

Cross-platform mobile app development frameworks allow app developers to specify the app's logic once using the programming language and APIs of a home platform and automatically generate versions of the app for other target platforms. It is very hard to develop such frameworks because they must correctly translate the home platform API to the target platform API while preserving the same behaviour and functionality. Inconsistencies and bugs may arise during the translation. To solve the problems, we need a kind of translation testing tool. For example, X-Checker [2] was designed and applied to test fidelity of Xamarin, a popular framework that enables Windows Phone apps to be cross-compiled into native Android apps. X-Checker generates randomized test cases for both platforms and detects inconsistencies by comparing both execution results. X-Checker had found 47 bugs, and 12 of them was fixed after reporting. The X-Checker's approach can be used to dynamically analyse Android apps.

Lee et al. [10] had observed an increase in the number of malware developed with cross-platform frameworks. They reported PhoneGap malware codes and found out that the pieces of malware conceal the malicious code in HTML files or containers loaded by cross-platform frameworks instead of the platform's native codes. They explained the app package structure for native Android and iOS platforms and then emphasized each cross-platform framework's characteristics including Unity application files. Their study analysed a *proof of concept* (POC) app that used cross-platform features to embed malicious code. The POC app was designed to include a malicious plug-in employed to conceal intended bad activities. By creating a Unity POC app, they showed that DLL files in the Unity app, both Android and iOS, shared the same compiled C# code in their package, and the C# code from the `Assembly-CSharp.dll` file invoked the plug-in APIs. Finally, they present a solution to identify an app's framework type and to write a detection signature for malware based on those frameworks.

Malware writers are eager to use the convenient tools such as Microsoft's .NET, high-level programming languages, and PowerSell frameworks to write fresh pieces of malware [11]. In addition, they have considered many measures to slow down malware analysis. For example, PowerSploit framework integrates a collection of PowerShell scripts and modules to be used in the post-exploitation stages of an attack. PowerSploit can execute the scripts to conduct low-level and administrative tasks without the need to drop malicious executables, aiming to avoid timely anti-virus detection. As another example, the Mono-developed projects can be ported from one platform to another using MoMA (Mono Migration Analyser) which gives malware writers a productivity boost in their malware lifecycle efforts. As mobile cross-platform frameworks are getting popular, app portability has also increased the number of pieces of malware that run on multiple platforms. To address these problems, Pontiroli et al. of Kaspersky Lab [11] have briefly reviewed how the .NET framework works and how a .NET PE is built to understand the differences in the analysis of .NET assemblies. For malware analysts, a set of tools including ILSpy decompiler is required to understand .NET malware, and a standardized process that fits their needs.

Dalmasso et al. [33] carried out a survey of "Write Once Run Anywhere" (WORA) tools along with a classification and comparison among the tools. They examined the performance of several cross-platform frameworks in terms of CPU, memory usage, and power consumption by developing Android test apps. They also compared a cross-platform app development approach with a native app development approach by using the nine decision criteria including "security of app". According to their research results, apps developed with cross-platform frameworks are not highly secure while security of the apps developed in a native approach is excellent. They point out that proper research needs to be carried out to secure the tools and apps.

Mylonas et al. [34, 35] evaluated comparatively the security level of popular smartphone platforms, considering their protection against simple malicious apps. They also examined the feasibility and easiness of writing malware by average programmers who could access the official tools and libraries supplied by smartphone platforms. According to their case study findings, (1) the Android malware was developed by the B.S. student in one day using the official development toolkit (SDK), Java, and the documentation of its API, (2) the BlackBerry malware was developed by the B.S. student in one day using RIM's SDK, Java, and the API documentation, (3) the iOS malware was developed by the M.S. student in seven days using Apple's SDK, Objective C, and the API documentation (in iOS case study, the malware developer had no prior experience with Objective C), and (4) on a Windows Mobile 6 and a Windows Phone 7, the malicious app was developed by the B.S. student in 2 days and in one day, respectively, using Microsoft SDK, C#, and the API documentation. Based on this proof of concept study, they had proven that, under circumstances, all examined platforms could be used by average programmers as privacy attack vectors, harvesting data from the smartphone without the users knowledge and consent.

Chen et al. [36] conducted a systematic study on potentially-harmful libraries (PhaLibs) across Android and iOS by observing that many iOS libraries had Android counterparts which could be used to understand their behaviours and the relations between the libraries on both sides. They first clustered similar packages from lots of popular Android apps to identify libraries and analysed the libraries using anti-virus (AV) systems to detect PhaLibs. Then, these libraries were mapped to the code components within iOS apps based on the invariant features shared across two platforms. They examined that 36 of top 38 iOS libraries have Android versions. The top 38 libraries include Unity, Cordova, PhoneGap, and Appcelerator Titanium.

Table 1: Common library files of Unity apps.

Filename	Description
`Assembly-CSharp.dll`	CIL codes generated by compiling developer's C# codes
`Assembly-CSharp-firstpass.dll`	
`System.*.dll`	File I/O support
`UnityEngine.dll`	Unity API support
`libmain.so`	Load and run `libmono.so` and `libunity.so` (redefine load/unload method)
`libmono.so`	Load classes in `Assembly-CSharp.dll` onto Mono runtime
`libunity.so`	Create Unity engine and Mono runtime

HTML5-based mobile apps and JavaScript apps are widely used because they are much easier to be ported across different mobile platforms. This flexibility of the JavaScript makes such apps prone to code injection attacks [37, 38]. Jin et al. [37] showed how HTML5-based apps could be vulnerable, how adversaries could exploit the vulnerabilities, and which damage could be incurred by the attacks. They also demonstrated the attacks using example apps and found that 11 PhoneGap plugins were vulnerable. Mao et al. [38] recently presented a technique to identify malicious behaviours in JavaScript apps and detect injection attacks to the apps and evaluated the effectiveness of their technique.

3. Static Analysis of Unity Apps

Unity is based on Mono project which is a cross-platform, open source .NET framework. It includes C# Compiler, Mono Runtime, .NET Framework Class Library, Mono Class Library, and APIs for each mobile environment. Unity users can develop mobile applications by writing programs in C# language and building them for any target device environment.

To support multiple platforms effectively, Unity apps deploy various types of object codes. For example, C# based Android apps developed with Unity contain Dalvik bytecode in `*.dex` files, .NET framework CIL code in DLL files and machine code in `*.so` files. In this section, we present a technique for static analysis of Unity apps, which includes disassembling or decompilation of these object codes. And we also present some results of the static analysis of an app we downloaded from the market to show the structure and interaction between object codes of Unity apps.

3.1. Structure of Unity Apps. We download a Unity app (Arrow.io) from the market and inspect its APK file. Figure 1 illustrates the structure of the APK file. The structure of Unity app is similar to native Android apps except that additional files and directories are included. In directory `/asset/bin/Data/Managed`, there are PE-format DLL files that contain CIL codes in their text sections. Developer's C# codes are compiled into `Assembly-CSharp*.dll` files, and these files are loaded and executed by Mono runtime. Other DLL files are added to support APIs such as Unity API and Mono embedded API. Directory `lib` contains native libraries for Mono runtime and JIT compilation. Table 1 lists several libraries common to Unity apps.

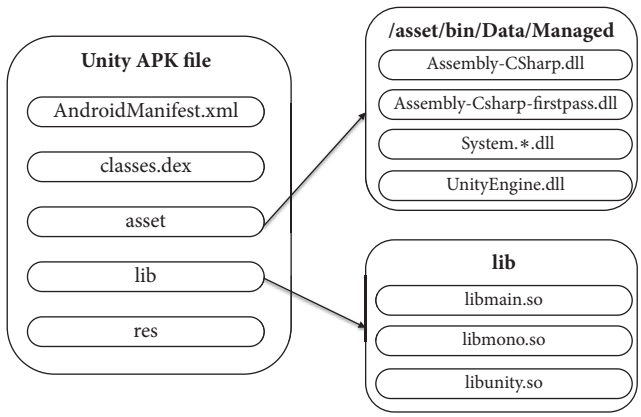


Figure 1: Structure of Unity APK File.

3.2. A Framework for Static Analysis of Unity Apps. As mentioned above, there are three types of execution codes in C# based Android apps built with Unity. They are Dalvik bytecode, .NET CIL code, and native code. Dalvik bytecode can be decompiled using Dex to Java decompilers such as jadx or JEB. CIL code can be decompiled into .NET C# code using .NET decompiler such as dotpeek, ILSpy, or .NET Reflector. Native code can be disassembled using disassembler such as IDA. Figure 2 shows our framework for static analysis.

3.3. Initialisation Process of Unity Apps. This section presents some results of the analysis focusing on the initialisation of Unity apps in order to show the effectiveness of the proposed framework for static analysis. Figure 3 illustrates the process of initialisation of a Unity app, Arrow.io, from the start of the app to loading of native libraries. When the app is launched, `onCreate()` method of `MainActivity` class (the class of activity component specified in `AndroidManifest.xml`) is invoked. `MainActivity`'s `onCreate()` calls its superclass method `UnityPlayerActivity`'s `onCreate()` first. Note that the activity component of Unity apps always inherits `UnityPlayerActivity`. `UnityPlayerActivity`'s `onCreate()` instantiates `UnityPlayer`.

Table 2: Comparison of Android native apps and Unity apps.

	Android native apps	Unity apps
Language	Java, C/C++	Java, C#, IL
APK directories	Lib, res, META-INF	Lib, res, META-INF, assets/bin/Data/Managed
Runtime Environment	Android Runtime	Mono Runtime
Symbol Recovery	Unnecessary	Necessary
Debugger	JDB, GDB	JDB, GDB, SDB
Debugger protocol	Java Debugger Wire Protocol	Soft Debugger Wire Format

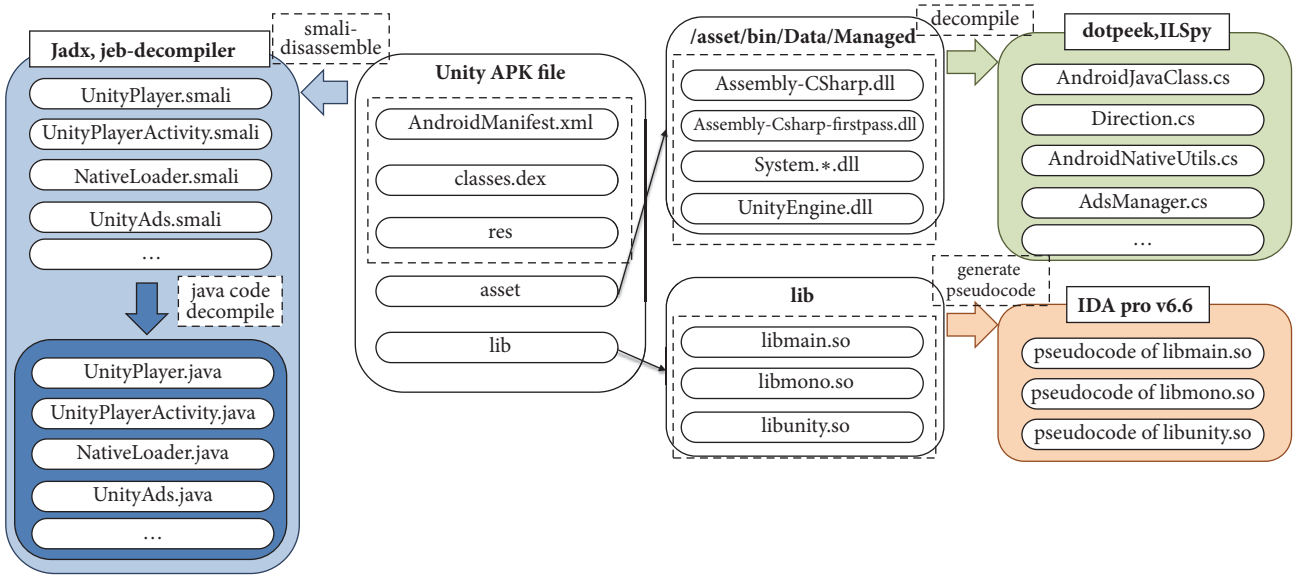


Figure 2: A framework for static analysis of Unity apps.

`UnityPlayer` loads `libmain.so` by calling `System.loadLibraryStatic()` method. When `libmain.so` is loaded, `JNI_OnLoad()` function of `libmain.so` is called and it binds its native function to `load()` method of `NativeLoader` class. `NativeLoader`'s `load()` loads `libunity.so` (a library for the core Unity game engine) and `libmono.so` (a library for Mono runtime). `JNI_OnLoad()` function of `libunity.so` binds its `nativeXXXX()` functions to native methods of `UnityPlayer`. It registers 17 native functions using `RegisterNatives()` function.

One of the registered native functions is `nativeRender()`. Called by `UnityPlayer`'s constructor, (1) it launches Mono runtime by invoking `mono_jit_init_version()` in `libmono.so`. Then (2) it creates an instance of `MonoManager` which is a C++ class declared in `libunity.so`. Finally (3) it invokes `MonoManager::LoadAssemblies()` which loads all DLL files (containing CIL codes) in `/assets/bin/Data/Managed` (see Figure 4).

Many mobile games support authentication using social login such as Facebook login or Google sign-in and/or display advertisement pages to earn money. Such facilities are typically incorporated in apps as a form of library. Since most of them are written in Java, the main game (written in C#) needs to call Java methods.

Unity apps utilise internal calls to find and invoke Java methods. C/C++ code can invoke Java methods using JNI functions such as `FindClass()`, `GetStaticFieldID()`, `GetMethodID()`, and `CallObjectMethod()`. `UnityEngine.dll` binds these functions to C# functions `INTERNAL_CALL_FindClass()`, `INTERNAL_CALL_GetStaticFieldID()`, `INTERNAL_CALL_GetMethodID()`, and `INTERNAL_CALL_CallObjectMethod()`, respectively. These functions are called internal calls and can be invoked by C# codes. They can be registered by calling `mono_add_internal_call()` in `libmono.so`.

The process of invoking Java methods from C# codes is as follows.

(1) Generate the values of options for Java virtual machine (`libunity.so`)

(2) Create Java virtual machine (`libunity.so`)

(3) Search and load Java class (`UnityEngine.dll`)

(4) Get the ID of Java method to invoke (`UnityEngine.dll`)

(5) Generate argument information for Java method (`UnityEngine.dll`)

(6) Invoke method (`UnityEngine.dll`)

(7) Destroy Java virtual machine (`libunity.so`)

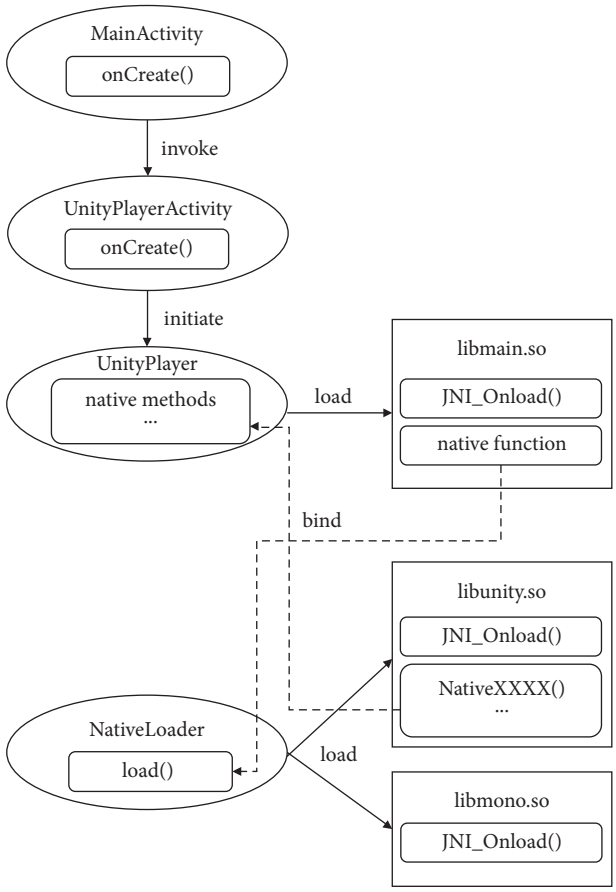


Figure 3: Loading native libraries.

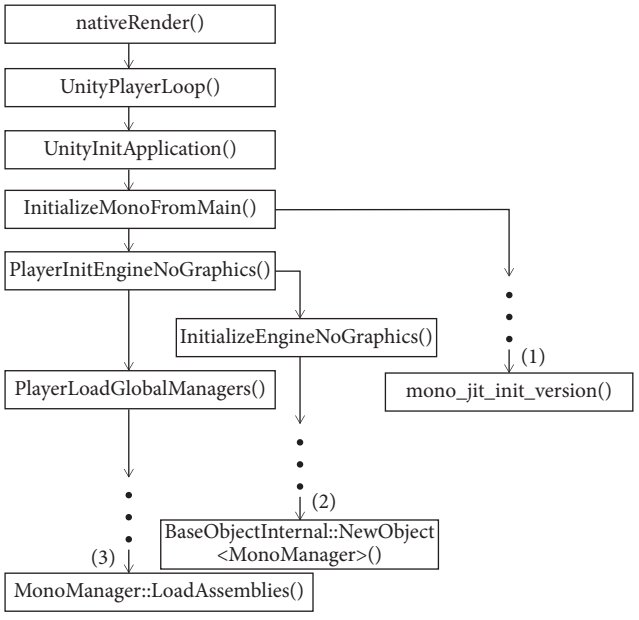


Figure 4: Loading DLL files.

Since it is impossible to create or destroy a Java virtual machine in the domain of C#, the operations related to creation/destroy of Java virtual machine are implemented in the native domain (`libunity.so`). The remaining jobs are implemented as C# functions (in `UnityEngine.dll`) for internal calls such as finding and loading Java classes, obtaining method ID of Java methods to invoke, generating argument information to pass to Java methods, and invoking Java methods.

Now the Unity engine and the Mono runtime are ready to execute developer's codes. `UnityPlayer`, `libunity.so`, and `libmono.so` collaboratively proceed with the main game. `libunity.so` invokes `libmono.so`'s functions to JIT-compile and execute developer's CIL codes.

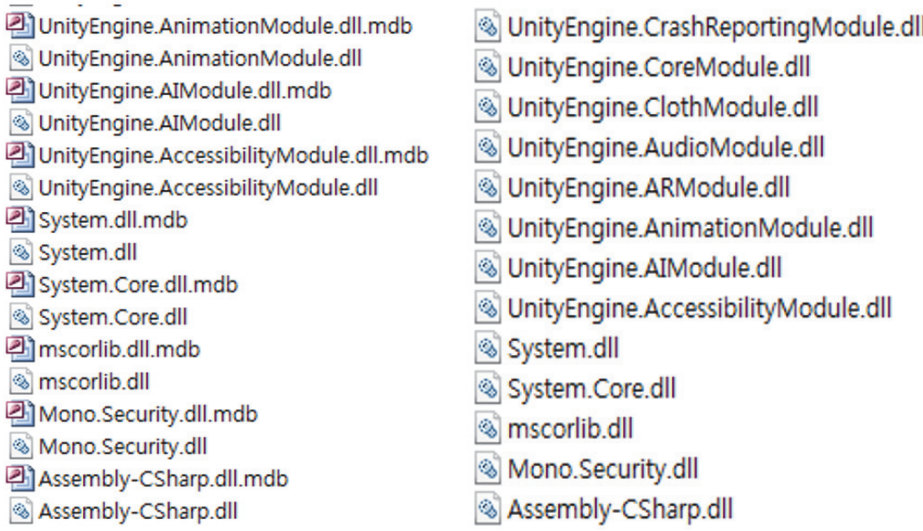
4. Dynamic Analysis of Unity Apps

Developers can build their apps in either a debugging or release mode. The former is called debug build and the latter is called release build. In debug build, apps include both debug symbol files and some libraries that can perform profiling functions. In release build, however, apps do not include debug symbol files or profiling libraries because they do not influence the execution of apps and the size of APK files can be reduced. Hence, the way how to perform dynamic analysis of an app depends on which mode the app is built in. Note that the term development build is used instead of the term debug build in Unity.

4.1. Development Build versus Release Build. This section explains the difference between development build and release build of Unity apps. We conduct both development build and release build on the same source codes of a Unity app and obtain two APK files. Then we analyse each of them within the framework presented in Section 3.

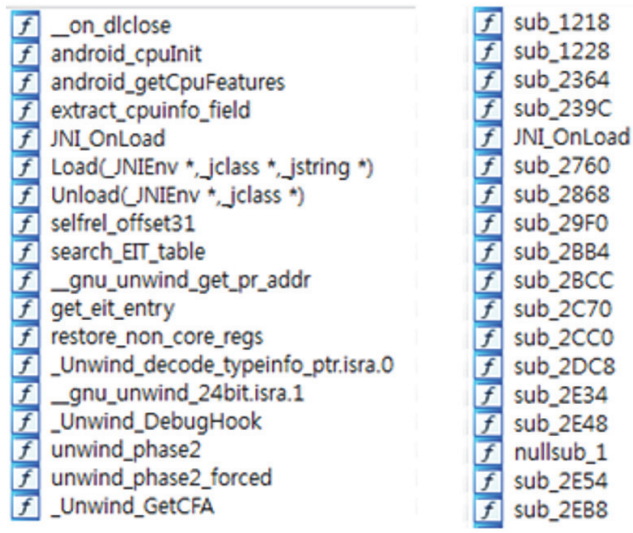
Figure 5(a) shows files in `/asset/bin/Data/Managed` of development-built APK, where there are DLL files and mdb files (files with extension .mdb). DLL files contain CIL codes and each mdb file contains the symbol information for its corresponding DLL file. In MS Windows environment, the symbol file for compiled C# code is pdb file (files with extension .pdb). In Unity apps, the symbol file is mdb file whose format is different from pdb file and is read and understood by Mono soft debugger which is a debugger built in Mono runtime. mdb files contain the information for mapping line numbers and methods in source codes. If Mono soft debugger cannot find mdb file, it cannot perform debugging operation such as setting breakpoints on the codes in the corresponding DLL file. Figure 5(b) shows the files in `/asset/bin/Data/Managed` of release-built APK. There is no mdb file because Unity does not generate mdb files in release build.

Development build and release build also differ in shared libraries. We inspect `libmain.so` of the two APK files using IDA pro. As shown in Figure 6, the symbol names remain in development build but are removed from release build. This is the same for other libraries common to Unity apps such as `libmain.so`, `libunity.so`, and `libmono.so`.



(a) Development build (b) Release build

Figure 5: Files in `/asset/bin/Data/Managed`.



(a) Development build (b) Release build

Figure 6: Symbols in `libmain.so`.

Another difference is the part that creates Mono runtime which is located in `libunity.so`. Our static analysis on `libunity.so` identified the part that initialises Mono runtime. Figure 7 shows the codes that create Mono runtime in development build. Mono runtime is created by invoking `mono_jit_init_version()`. Before invoking this function, `mono_debug_init()` is invoked to prepare Mono soft debugger inside Mono runtime. Then debuggers outside Mono runtime can communicate with Mono soft debugger over a socket connection. In release build, `mono_debug_init()` is not invoked as shown in Figure 8. This implies that we should put some codes invoking `mono_debug_init()` into `libunity.so` in order to connect to Mono soft debugger and dynamically analyse release-built apps.

The abovementioned difference between development build and release build makes it complicated to analyse release-built app dynamically. Since most Unity apps in the market are built in release mode, they neither contain debug symbol files nor initialise Mono soft debugger. The rest of this section presents a technique that can implement a debugging environment for dynamic analysis of release-built apps by generating debug symbol files and executing codes that initialise Mono soft debugger.

4.2. Debugging Release-Built Apps. The proposed dynamic analysis technique proceeds in the following order (as shown in Figure 9). To use Mono soft debugger, we generate debug symbol files (mdb files), modify `AndroidManifest.xml` file,

```
mono_debug_init(1);
v67 = v91 == 0;
if ( v91 )
  v67 = v92 == 0;
if ( !v67 )
  free_alloc_internal(v91, &v94);
v68 = LODWORD(v96) == 0;
if ( LODWORD(v96) )
  v68 = HIDWORD(v96) == 0;
if ( !v68 )
  free_alloc_internal(LODWORD(v96), &v99);
v69 = 0;
mono_set_commandline_arguments(v76);
v70 = mono_jit_init_version("Unity Root Domain", "v2.0.50727");
```

Figure 7: Mono runtime initialisation in development-built APK.

```
mono_set_assemblies_path(v38, v50);
v39 = v50 == 0;
if ( v50 )
  v39 = v51 == 0;
if ( !v39 )
  sub_102614(v50, v53);
sub_2DEA8C(unk_12D2060, v3);
v40 = v54 == 0;
if ( v54 )
  v40 = v55 == 0;
if ( !v40 )
  sub_102614(v54, v57);
v41 = v58 == 0;
if ( v58 )
  v41 = v59 == 0;
if ( !v41 )
  sub_102614(v58, v61);
v42 = 0;
mono_config_parse(0);
mono_set_signal_chaining(1);
v43 = mono_parse_default_optimizations(0);
mono_set_defaults(0, v43);
mono_set_commandline_arguments(v49);
v44 = mono_jit_init_version("Unity Root Domain", "v2.0.50727");
```

Figure 8: Mono runtime initialisation in release-built APK.

and repackage the app. After installing this repackaged app on an Android device, we launch the app and stop it before Mono runtime is created. We may modify `classes.dex` to stop the app or use the Debugger-Wait function with `am` command. When the app is stopped, we hook a native function to execute the code that invokes `mono_debug_init()` and resume the app. Once this code is executed, we can connect to Mono soft debugger for debugging by attaching SDB, a client for Mono soft debugger. The following sections explain this procedure step by step.

4.2.1. Generating Debug Symbol Files. To debug a release-built app, we must first create mdb files. However, as mentioned in Section 4.1, the release build does not create mdb files. We need to create mdb files manually and repackage the app. Figure 10 illustrates this process.

We use the `ildasm` and the `ilasm` commands, which are installed with Visual Studio. To create an mdb file, we need to create a pdb file first. The `ildasm` command disassembles a DLL file and creates an IL (intermediate language) text file with extension .il. The `ilasm` command reassembles an IL text file. Given proper arguments, it creates a new DLL file and a pdb file. We, then, use the `pdb2mdb` command to convert the pdb file to an mdb file.

We modify the `AndroidManifest.xml` file as follows. We set `debuggable` attribute of `application` element to `true` and allow `INTERNET` permission to perform remote debugging. We repackage the app after signing back and then install the app on the device. Now preparing the app for dynamic analysis completes.

4.2.2. Attaching a Debugger for Dynamic Analysis. We describe how to connect SDB to repackaged apps. As mentioned in Section 4.1, a release-built app does not have code to call the `mono_debug_init()` function. To connect SDB to Mono soft debugger, we must call the `mono_debug_init()` function before the Mono runtime instance is created. To do this, we must be able to do two things:

(1) To stop execution of an app before Mono runtime instance is created

(2) To call `mono_debug_init()` function directly

On Android, we can use the am command to run an app in the Debugger-Wait state. If we do so, we can see that the process is suspended waiting for a debugger to attach. When we attach JDB, the process is resumed. The example command line is as follows.

```
# adb shell am set-debug-app -w
<application> --persistent
```

We hook a native function `mono_set_default()` in `libunity.so` using an instrumentation tool such as Frida. We ascertained by static analysis that this function is always called prior to `mono_jit_init_version()`. We inject the following code into `mono_set_default()` to initialise Mono soft debugger by invoking `mono_debug_init()`.

```
char * dbgoption={"--debugger-
agent=transport=dt_socket,embedding=1,
defer=y,address=0.0.0.0:9999",NULL};
mono_jit_parse_options(1,dbgoption);
mono_debug_init(1);
```

To attach SDB to an app, we first connect JDB to the app, execute the above Mono soft debugger initialisation code, and run SDB to connect to Mono soft debugger. We can then debug a Unity app as usual. Figure 11 is a screen shot of SDB. We set breakpoint at the `Test.Update()` method using bp command (fourth line) and connect to Mono soft debugger using `connect` command with the IP address of the target Android device (sixth line). Once the connection is established, Mono runtime process and C# threads are started (8th–15th lines). Then the app stops at the `Test.Update()` method and prompts for debugger command (16th–18th lines). The app stops at line 69 in the IL text file (`Assembly-CSharp.il`) and displays the next code to execute, `ldstr "Test Text"`, which is the first code of `Test.Update()`.

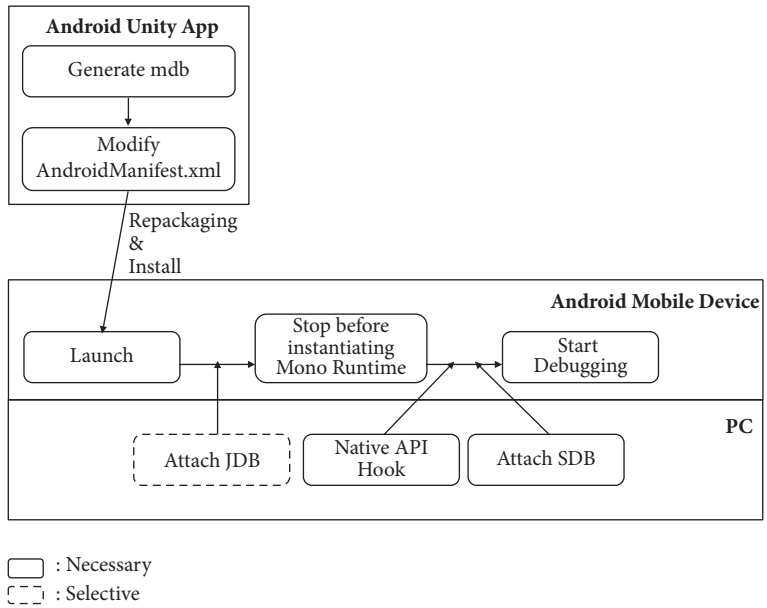


Figure 9: Procedure of dynamic analysis.

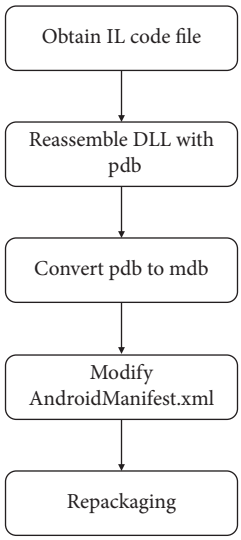


Figure 10: Process of creating mdb files and repackaging apps.

Figure 11: A screen shot of SDB.

5. Discussion

Typical Android apps and Unity apps differ in terms of their app structure and execution environment (Table 2). First, Unity apps have `/assets/bin/Data/Managed` folder and contain DLL files for execution. Second, they have different libraries in their package. Android apps include native libraries only when developers explicitly specify, while Unity apps always include three libraries: `libmono.so`, `libmain.so`, and `libunity.so`. Of course, developers can specify libraries as needed. Third, the code runs in different runtime environments. Android apps run on the Android Runtime, while Unity apps run on the Mono runtime, which executes the IL codes. Fourth, Android apps can be analysed dynamically if we merely modify `AndroidManifest.xml`, but Unity apps can be analysed only after restoring symbol information of DLL files. Finally, they use different debugger protocols due to different runtime environments. JDB uses Java Debug Wire Protocol (JDWP), which performs debugging on Java code. SDB, however, uses Soft Debugger Wire Format protocol (SDB protocol), which performs debugging on IL codes. Two protocols are similar because SDB protocol is made by referring to JDWP.

We propose a method for dynamically analysing Unity apps on the Java layer, the native layer, and the Mono runtime layer where C # code is executed. JDB is connected to the Java layer, GDB to the native layer, and SDB debugger to the Mono runtime layer to perform operations such as setting breakpoints or querying parameters.

Malware analysis techniques can be classified into two categories: static and dynamic. Static analysis, mostly used by anti-virus program, can analyse target's executable by looking at suspicious patterns or by disassembling and further decompiling it into high-level language like C# or Java. Static analysis can give a complete result of the executable by covering different execution paths including static control flow and data flow. However, static analysis may have limitations

Table 3: Static analysis tools for Unity apps.

Tool	Description
jadx	Command line and GUI tools for producing Java source code from Android DEX file
jd-gui	GUI tool for displaying Java source codes of ".class" files
ILspy	Open-source decompiler and static analyser for software created with .NET Framework
dotpeek	Decompiler and static analyser for software created with .NET Framework
.NET Reflector	Commercial decompiler and static analyser for software created with .NET Framework
IDA pro	Disassembler for computer software which generates assembly language source code from machine-executable code
ildasm	Tool for generating Common Intermediate Language (CIL) code from portable executable (PE) file
ilasm	Tool for generating a portable executable (PE) file from Common Intermediate Language (CIL) code
pdb2mdb	Command line tool for convert .NET debug symbol files (namely, pdb files) to debug symbols files (namely, mdb files) that can be understood by Unity's scripting engine

Table 4: Dynamic analysis tools for Unity apps.

Tool	Description
JDB	Command line debugger for Java classes
GDB	Portable debugger for tracing and altering the execution of computer programs
SDB	Command line client for Mono's soft debugger

due to the complexity of pointer aliasing, the prevalence of indirect jumps, and the lack of types in executables. It is also known that static analysis is vulnerable to code packing and obfuscation methods, which are commonly applied to malware to avoid anti-virus detection. Some malicious apps are distributed even in a form of components, which are decrypted at runtime.

In case malware is obfuscated or packed to hinder static analysis, dynamic analysis or behaviour-based analysis can be effective. Dynamic analysis can observe execution traces of the target app behaviour and analyse its properties by executing the malware in an isolated and controlled environment. It can also monitor actions of the target app during execution with the help of a specialized tool such as a debugger. Therefore, dynamic analysis is resilient to code obfuscation and packing and is able to monitor malicious behaviour on an actual execution path. The disadvantage is the lack of code coverage as it explores a single execution path at a time.

We summarise useful tools for analysing Unity apps statically and dynamically in Tables 3 and 4, respectively. Static and dynamic analysis both have their own merits and demerits. A few researches have tried to combine static and dynamic analysis whenever possible to have the benefits of both. In this paper, we present an effective static and dynamic analysis technique for detecting Android malware written with Unity framework.

6. Conclusion

Cross-platform mobile app development tools allow developers to write only one mobile app that is then deployed to all the supported target platforms, to reduce the cost and time for developing mobile apps, and to reach out to maximum users across several platforms. While app authors use cross-platform mobile app development tools for the "*Write once, Run everywhere*" feature, malware authors use them for the "*Write once, Infect everywhere*" feature. Unity is one of the most popular cross-platform mobile app development tools. A recent study conducted by SophosLabs reported that the number of malware and *potentially unwanted apps* (PUAs) written with Unity had been increasing in proportion to the number of mobile apps developed with Unity. Moreover, the pieces of malware conceal its malicious code in specific containers (e.g., DLL files) loaded by Unity instead of the target platform's native codes. Therefore, security researchers are confronted with great challenges to detect these pieces of mobile malware.

To address the challenges, in this paper, we first present a systematic technique that statically and dynamically analyse *Android apps developed with Unity framework* (Unity apps). Our static analysis focuses on the initialisation of target apps to examine the structure and interaction between object codes of the apps. The static analysis consists of the following behaviours: understanding executable file format, decompiling DEX, Microsoft .NET DLLs, and native codes, and inspecting static control flow of the decompiled codes. Next, we propose an effective dynamic analysis technique for Unity apps that are built in release mode. Such apps cannot be debugged as they are, since they do not contain any debugging symbol. To debug a release version of Unity app, we generate the debugging symbols from the release version, repackage the release version with the generated symbols, and then run the repackaged app. We can stop the execution of the Unity app before the Mono runtime instance is created and call debugging preparation function directly. After attaching the debugger, we can set breakpoints and inspect the parameters. The proposed techniques are effective and efficient for determining if a suspicious app written with Unity is malicious or benign and can be used to discover bugs or vulnerabilities in the suspicious app.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning [no. 2015R1A2A1A15053738]. This research was also supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program [IITP-2018-2015-0-00363] supervised by the IITP (Institute for Information & Communications Technology Promotion).

References

[1] M. Willocx, J. Vossaert, and V. Naessens, "A Quantitative Assessment of Performance in Mobile App Development Tools," in *Proceedings of the 3rd IEEE International Conference on Mobile Services, MS 2015*, pp. 454–461, usa, July 2015.

[2] N. Boushehrinejadmoradi, V. Ganapathy, S. Nagarakatte, and L. Iftode, "Testing cross-platform mobile app development frameworks," in *Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering, ASE 2015*, pp. 441–451, usa, November 2015.

[3] M. Martinez and S. Lecomte, "Towards the Quality Improvement of Cross-Platform Mobile Applications," in *Proceedings of the 4th IEEE/ACM International Conference on Mobile Software Engineering and Systems, MOBILESoft 2017*, pp. 184–188, arg, May 2017.

[4] W. S. El-Kassas, B. A. Abdullah, A. H. Yousef, and A. M. Wahba, "Taxonomy of Cross-Platform Mobile Applications Development Approaches," *Ain Shams Engineering Journal*, vol. 8, no. 2, pp. 163–190, 2017.

[5] H. Rocky, *Designing Platform Independent Mobile Apps and Services*, John Wiley & Sons, 2016.

[6] S. Blackman, *Beginning 3D Game Development with Unity 4: All-in-one, multi-platform game development*, Apress, Berkeley, CA, USA, 2013.

[7] J. Haas, *A History of the Unity Game Engine*, Worcester Polytechnic Institute, Worcester, UK, 2014.

[8] P. P. Patil and R. Alvares, "Cross-platform Application Development using Unity Game Engine," *International Journal of Advance Research in Computer Science and Management Studies*, vol. 3, no. 4, pp. 19–27, 2015.

[9] Unity Engine - Official Site, "Unity," http://unity3d.com.

[10] W. Lee and X. Wu, "Cross-platform mobile malware: write once, run everywhere," in *Proceedings of the Virus Bulletin Conference*, 2015.

[11] S. M. Pontiroli and F. R. Martinez, "he Tao of .NET and PowerShell Malware Analysis," in *Proceedings of the Virus Bulletin Conference*, 2015.

[12] "The Mono Runtime - Official Site," http://www.mono-project.com/docs/advanced/runtime/.

[13] L. Batyuk, M. Herpich, S. A. Camtepe, K. Raddatz, A.-D. Schmidt, and S. Albayrak, "Using static analysis for automatic assessment and mitigation of unwanted and malicious activities within Android applications," in *Proceedings of the 6th International Conference on Malicious and Unwanted Software, Malware 2011*, pp. 66–72, pri, October 2011.

[14] S. Y. Yerima, S. Sezer, G. McWilliams, and I. Muttik, "New android malware detection approach using Bayesian classification," in *Proceedings of the 27th IEEE International Conference on Advanced Information Networking and Applications, AINA 2013*, pp. 121–128, Spain, March 2013.

[15] B. Jean et al., "Static detection of malicious code in executable programs," *International Journal of Advanced Structural Engineering*, vol. 79, pp. 184–189, 2001.

[16] T. Ball and S. K. Rajamani, "The SLAM project: debugging system software via static analysis," *ACM SIGPLAN Notices*, vol. 37, no. 1, pp. 1–3, 2002.

[17] D. Evans and D. Larochelle, "Improving security using extensible lightweight static analysis," *IEEE Software*, vol. 19, no. 1, pp. 42–51, 2002.

[18] S. Arzt, S. Rasthofer, C. Fritz et al., "FlowDroid: precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," in *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '14)*, pp. 259–269, ACM, June 2014.

[19] S. Schrittwieser and S. Katzenbeisser, "Code Obfuscation against Static and Dynamic Reverse Engineering," in *Information Hiding*, vol. 6958 of *Lecture Notes in Computer Science*, pp. 270–284, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[20] A. Moser, C. Kruegel, and E. Kirda, "Limits of static analysis for malware detection," in *Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC '07)*, pp. 421–430, December 2007.

[21] U. Bayer, A. Moser, C. Kruegel, and E. Kirda, "Dynamic analysis of malicious code," *Journal of Computer Virology and Hacking Techniques*, vol. 2, no. 1, pp. 67–77, 2006.

[22] G. Willems, T. Holz, and F. Freiling, "Toward automated dynamic malware analysis using CWSandbox," *IEEE Security & Privacy*, vol. 5, no. 2, pp. 32–39, 2007.

[23] A. Lanzi, M. I. Sharif, and Lee. Wenke, "K-Tracer: A System for Extracting Kernel Malware Behavior," *NDSS*, 2009.

[24] G. Richards, S. Lebresne, B. Burg, and J. Vitek, "An analysis of the dynamic behavior of JavaScript programs," *ACM SIGPLAN Notices*, vol. 45, no. 6, p. 1, 2010.

[25] K. Rieck, P. Trinius, C. Willems, and T. Holz, "Automatic analysis of malware behavior using machine learning," *Journal of Computer Security*, vol. 19, no. 4, pp. 639–668, 2011.

[26] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A survey on automated dynamic malware-analysis techniques and tools," *ACM Computing Surveys*, vol. 44, no. 2, article 6, 2012.

[27] P. Faruki, V. Ganmoor, V. Laxmi, M. S. Gaur, and A. Bharmal, "AndroSimilar: Robust statistical feature signature for android malware detection," in *Proceedings of the 6th International Conference on Security of Information and Networks, SIN 2013*, pp. 152–159, tur, November 2013.

[28] J. Lim and J. H. Yi, "Structural analysis of packing schemes for extracting hidden codes in mobile malware," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, 2016.

[29] T. Cho, H. Kim, and J. H. Yi, "Security Assessment of Code Obfuscation Based on Dynamic Monitoring in Android Things," *IEEE Access*, vol. 5, pp. 6361–6371, 2017.

[30] D. Andrade, R. M. Paulo et al., *Cross platform app: a comparative study*, 2015, arXiv:1503.03511.

[31] T. Majchrzak and T. Grønli, "Comprehensive Analysis of Innovative Cross-Platform App Development Frameworks," in *Proceedings of the Hawaii International Conference on System Sciences*, 2017.

[32] M. Latif, Y. Lakhrissi, E. H. Nfaoui, and N. Es-Sbai, "Review of mobile cross platform and research orientations," in *Proceedings of the 2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems, WITS 2017*, mar, April 2017.

[33] I. Dalmasso, S. K. Datta, C. Bonnet, and N. Nikaein, "Survey, comparison and evaluation of cross platform mobile application development tools," in *Proceedings of the 9th International Wireless Communications and Mobile Computing Conference (IWCMC '13)*, pp. 323–328, July 2013.

[34] A. Mylonas, S. Dritsas, B. Tsoumas, and D. Gritzalis, "On the Feasibility of Malware Attacks in Smartphone Platforms," in *E-Business and Telecommunications*, vol. 314 of *Communications in Computer and Information Science*, pp. 217–232, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[35] A. Mylonas, S. Dritsas, B. Tsoumas, and D. Gritzalis, "Smartphone security evaluation—the malware attack case," in *Proceedings of the 8th International Conference on Security and Cryptography (SECRYPT '11)*, pp. 25–36, Seville, Spain, July 2011.

[36] K. Chen, X. Wang, Y. Chen et al., "Following Devil's Footprints: Cross-Platform Analysis of Potentially Harmful Libraries on Android and iOS," in *Proceedings of the 2016 IEEE Symposium on Security and Privacy (SP)*, pp. 357–376, San Jose, CA, May 2016.

[37] J. Xing et al., *Code injection attacks on HTML5-based mobile apps*, 2014.

[38] J. Mao, J. Bian, G. Bai et al., "Detecting Malicious Behaviors in JavaScript Applications," *IEEE Access*, vol. 6, pp. 12284–12294, 2018.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 9641273, 6 pages
https://doi.org/10.1155/2018/9641273

Research Article

A Homomorphic Network Coding Signature Scheme for Multiple Sources and its Application in IoT

Tong Li,[1] **Wenbin Chen**,[1] **Yi Tang**,[2] **and Hongyang Yan**[3]

[1]*School of Computer Science, Guangzhou University, Guangzhou, China*
[2]*School of Mathematics and Information Science, Guangzhou University, Guangzhou, China*
[3]*College of Computer and Control Engineering, Nankai University, Tianjin, China*

Correspondence should be addressed to Yi Tang; ytang.bjs@139.com

Received 13 January 2018; Revised 21 March 2018; Accepted 15 April 2018; Published 14 June 2018

Academic Editor: Ilsun You

As a method for increasing throughput and improving reliability of routing, network coding has been widely used in decentralized IoT systems. When files are shared in the system, network coding signature techniques can help authenticate whether a modified packet in files is injected or not. However, in an IoT system, there are often multiple source devices each of which has its own authentication key, where existing single-source network coding signature schemes cannot work. In this paper, we study the problem of designing secure network coding signatures in the network with multiple sources and propose the multisource homomorphic network coding signature. We also give construction and prove its security.

1. Introduction

With the rapid development of Internet and communication technologies, the Internet of Things (IoT) has emerged as a leading technology that brings convenience to our daily lives. More and more smart terminals are connected on the Internet, and files, logs, and other real-time contents are shared among these terminals all the times. According to a report of International Data Corporation (IDC), there will be nearly 28 billion installed IoT devices by 2020.

Considering the scale of IoT's expansion, it is very essential to increase its throughput in such a huge decentralized network. When a source device transmits a file to a set of target receivers, an effective way is to split the file into t data packets and send them to its neighbouring nodes by using the network coding technique. In the network coding, each intermediate node linearly combines packets rather than simply storing and forwarding the incoming packets. In other words, an intermediate node that receives a set of packets from its incoming links can modify them and send the modifications to other nodes through its outgoing edges. In some applications, either an ad hoc node or an intermediate device can play a role of an intermediate node. This linear network coding allows receivers to recover the original information with high probability if they collect sufficiently many correct packets. Thus, the throughput for sharing real-time contents in IoT is increased.

However, security is one of the most important requirement of IoT systems, and IoT devices often interact with third-party applications. Without authenticate mechanisms, the inherent flaw of linear network coding would be disturbed by invalid packets injected by third-party applications. Intermediate nodes can later use the invalid incoming vectors in its output, which means that the errors are propagated subsequently and data receivers will not obtain the original information. As a result, adversaries could easily initiate a Denial of Service (DoS) attack to prevent the original file from being recovered. The main idea to mitigate attacks is to provide a way to authenticate valid packets, and Catalano et al. [1] proposed an efficient network coding signature scheme as a solution of the authentication problem. By verifying a modified signature of the corresponding modified packet, any device can easily know whether this packet is valid.

Unfortunately, in an IoT system, origin data are usually collected from various sources (e.g., sensors) each of which could have its own signature for authentication. It is required

that any (intermediate) receiver can perform the combination of incoming packets which are signed by different keys. As a drawback, trivially adopting the existing network coding has to generate signatures are linear in the number t of the sources, and thus the signatures cannot be directly combined when packets are modified. Motivated by this problem, in this paper, we propose a multisource linearly homomorphic network coding signature scheme. The proposed scheme is extended from our previous work [2] and enables a multilayers routing network rather than a 3-layer one, which can be used to implement authentication for transmitting files in the IoT system.

The rest of this paper is organized as follows. Section 2 presents some related works. Section 3 overviews some definitions. In Section 4, we describe our multisource linearly homomorphic network coding signature scheme. Section 5 analyzes the correctness and security of the proposed scheme. In Section 6 we summarize the paper.

2. Related Works

In the traditional network routing, every node simply receives packets and forwards them to neighbour nodes. A routing method called network coding [3, 4] is proposed and developed for increasing throughput in the network. In the network coding, intermediate nodes combine received data packets and transit them and the data receiver still obtains the original data. This technique can be used in IoT applications and cloud systems for broadcast and transmission [5–19].

In the single-source scenario, some schemes were proposed to make sure that there is always a recipient bound to the corresponding for authentication. M. Krohn et al. introduced the homomorphic hash function [20, 21] and extended it to network coding. The linearly homomorphic signature is a more effective authentication for the network coding. Reference [22] proposed the first linearly homomorphic signature scheme. Reference [23–25] found some security flaw and errors, and Yu et al. [26] gave a construction by combining the RSA-based signature with the homomorphic hash function. Reference [27, 28] designed signature schemes for peer-to-peer networks and distributed contents respectively. Reference [29, 30] proposed homomorphic network coding signature schemes based on the bilinear mapping and RSA assumption respectively. In [31], Boneh et al. designed a signature scheme with the property of signing unlimited number of messages. Based on the complexity of lattice problems, [32] introduced the k-SIS problem and constructed a signature scheme over binary fields. For a fine-grain access control, [33, 34] proposed schemes based on the identity-based signature. The schemes above are proven secure in the random oracle model. In the standard model, some homomorphic network coding signature schemes were proposed [1, 34–36]. The security of the scheme in [35] is based on the discrete logarithm assumption. Independent of these works, [37] proposed a method that transforms standard signature schemes to linearly homomorphic signatures in the standard model.

However, in a multisource case which is the common scenario in the IoT system, there is still no linearly homomorphic network coding signature scheme. Our goal is to design a multisource linearly homomorphic network coding signature scheme.

3. Preliminaries

Then, we show some definitions of the linearly homomorphic network coding signature as follows.

Definition 1 (linearly homomorphic network coding signatures adapted from [1]). A linearly homomorphic network coding signature scheme $\mathcal{LS}$ consists of a tuple of probabilistic, polynomial-time algorithms ($NetKG$, $NetSign$, $NetVer$, $NetEval$) with the following functionality.

NetKG$(1^\lambda, m, n) \to (pk, sk)$. Given the security parameter λ and m, n, this algorithm outputs a key pair (sk, pk), where sk is the secret key and pk is the public verification key. Note that m is the dimension of the vector spaces and $n + m$ is an upper bound to the size of the signed vectors.

NetSign$(sk, Id, w) \to \sigma$. The signing algorithm takes a secret key sk, a file identifier $Id \in \mathbb{F}_p$ and a vector $w \in \mathbb{F}_p^{n+m}$ as input and then outputs a signature σ.

NetVer$(pk, Id, w, \sigma) \to accept$. Given the public key pk, a file identifier Id, a vector $w \in \mathbb{F}_p^{n+m}$, and a signature σ, the algorithm outputs a bit *accept* represents accept or reject.

NetEval$(pk, Id, \{(w_i, \alpha_i, \sigma_i)\}_{i=1}^{\mu}) \to \sigma$. Given a public key pk, a file identifier Id, and a set of tuples $(w_i, \alpha_i, \sigma_i)$, this algorithm outputs a new signature σ such that if each σ_i is a valid signature on vector w_i, then σ is a valid signature for w obtained from the linear combination $\sum_{i=1}^{\mu} \alpha_i w_i$.

For *correctness*, it is required that if the key pair (sk, pk) is output by **NetKG**$(1^\lambda, m, n)$, then

(i) let $Id \in \mathbb{F}_p$ and $w \in \mathbb{F}_p^{n+m}$; if $\sigma \leftarrow$ **NetSign**(sk, Id, w), then **NetVer**$(pk, Id, w, \sigma) = 1$;

(ii) for all Id, any $t > 0$ and all sets of triples $\{(w_i, \alpha_i, \sigma_i)\}_{i=1}^{\mu}$; if **NetVer**$(pk, Id, w_i, \sigma_i) = 1$ for all i, then **NetVer**$(pk, Id, \sum_{i=1}^{\mu} \alpha_i w_i$ and **NetEval**$(pk, Id, \{(w_i, \alpha_i, \sigma_i)\}_{i=1}^{\mu})) = 1$.

The definition of unforgeability of linearly homomorphic signature is presented as follows.

Definition 2 (unforgeability adapted from [32]). For a linearly homomorphic network coding signature scheme $\mathcal{LS} = (NetKG, NetSign, NetVer, NetEval)$, the following game is considered.

Setup: The challenger runs $NetKG(1^\lambda, m, n)$ to obtain (sk, pk) and gives pk to $\mathcal{A}$.

Queries: Proceeding adaptively, $\mathcal{A}$ specifies a sequence of data sets $\mathbf{w}_i$. For each i, the challenger chooses Id_i uniformly

from $\mathbb{F}_p$ and gives to $\mathcal{A}$ the tag Id_i and the signatures $\sigma_{ij} \leftarrow NetSign(sk, Id_i, w_{ij})$ for $j = 1, \ldots, k$.

Output: $\mathcal{A}$ outputs a file identifier Id^*, a message m^*, and a signature σ^*. The adversary wins if $NetVer(pk, Id^*, w^*, \sigma^*) = 1$, and either (1) $Id^* \neq Id_i$ for all i or (2) $Id^* = Id_i$ for some i but $w^* \notin \text{span}(\mathbf{w}_i)$, where $\text{span}(\mathbf{w}_i)$ is the subspace generated by all $\mathbf{w}_i$.

The advantage of $\mathcal{A}$ is defined to be the probability that $\mathcal{A}$ wins the security game. $\mathcal{LS}$ is called unforgeable if for any PPT adversary $\mathcal{A}$, the advantage in the game is negligible in λ.

Let $e : \mathbb{G} \times \mathbb{G}' \rightarrow \mathbb{G}_{\mathbb{T}}$ be a bilinear map, where $\mathbb{G}, \mathbb{G}'$ and $\mathbb{G}_T$ are bilinear groups of prime order p. In [38], Boneh and Boyen introduced the definition of the q-Strong Diffie-Hellman Assumption (q-SDH for short).

Definition 3 (q-SDH assumption [38]). Let $k \in \mathbb{N}$ be the security parameter, $p > 2^\lambda$ be a prime, and $\mathbb{G}, \mathbb{G}', \mathbb{G}_{\mathbb{T}}$ be bilinear groups of prime order p. Let g be a generator of $\mathbb{G}$ and g' be a generator of $\mathbb{G}'$, respectively. Then we say that the q-SDH Assumption holds in $\mathbb{G}, \mathbb{G}', \mathbb{G}_{\mathbb{T}}$ if for any PPT algorithm $\mathcal{A}$ and any $q = \text{poly}(k)$, the following probability is negligible in λ:

$$\Pr\left[\mathcal{A}\left(g, g^x, g^{x^2}, \ldots, g^{x^q}, g', \left(g'\right)^x\right) = \left(c, g^{1/(x+c)}\right)\right] \leq \text{negl}(\lambda) \quad (1)$$

4. The Proposed Scheme

4.1. Architecture. Consider an application in practical. A log report of some intelligent terminals is supposed to be jointly published via the linear network coding. To prevent the injection of invalid data packets and make the transmission reliable, each data packet should be suffixed with a valid recipient before forwarding. A network coding signature scheme can help to meet this requirement when all terminal devices have the same key used for signing packets. However, if each device has its own key, a group of signatures cannot be directly combined for the corresponding packets. As a solution for the verification problem, we present this homomorphic network coding signature scheme for multiple sources.

An architecture is shown in Figure 1. A terminal device can be seen as a source, while the receiver wants to get the log report with a correct recipient. Each entity in the scheme is described as follows:

(i) **Source nodes**. After some parameters are generated as public information, the ith source node generates its own key pair (pk_i, sk_i) for signing and verifying. Each node has a part of the original file, and the part can be seen as a data packet. To obtain a signature σ, the ith node signs its packet that belongs to the file with an identifier Id using sk_i. Then, it sends the signed tuple $(Id, \mathbf{w}, \sigma)$ on its outgoing edges.

(ii) **Intermediate nodes**. When an intermediate node receives some packets with the corresponding signatures, it checks whether any one is not valid. Then, it selects μ coefficients for the rest valid packets $\mathbf{w}_1, \ldots, \mathbf{w}_\mu$, and combines the packets and their signatures, respectively. Finally, the combined tuple is forwarded on the outgoing edges.

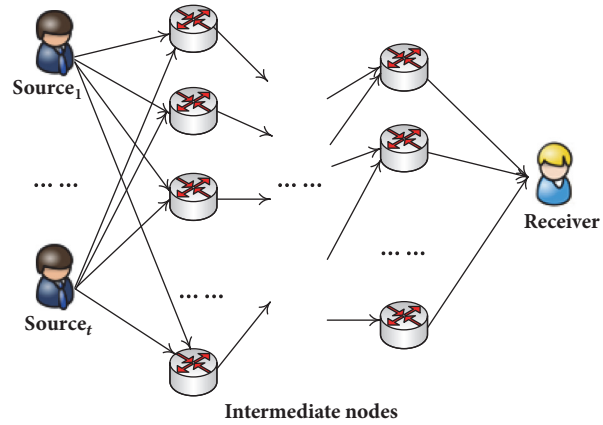


Figure 1: Architecture of the proposed scheme.

(iii) **Receiver**. Once the receiver has collected Id file's packets signed by using all t secret keys, it checks the validity and recovers the original file if the check is passed.

4.2. Scheme Description. In this section, we present our construction of the multisource homomorphic network coding signature scheme. There are t devices as source nodes, any number of intermediate nodes, and several receivers. For simplicity, we assume that each source node holds and forwards a packet of a file. The whole file can be represented as a augmented vector set $\mathbf{w} = \{\mathbf{w}_1, \ldots, \mathbf{w}_t\}$, of which the ith packet can be represented as $\mathbf{w}^{(i)} = (u_1^{(i)}, \ldots, u_m^{(i)}, v_1^{(i)}, \ldots, v_n^{(i)})$. Each packet belongs to a file with the ID Id and some of the packets are encoded together with the same Id.

As described above, each source node has its own key pair in the system. For the ith source, its private key is used to sign $\mathbf{w}^{(i)}$ so that the signed packets can be verified by intermediate nodes which receive the signatures. After receiving several input packets, an intermediate node firstly checks each packet and discards all the packets that cannot pass the check. With a signature σ, the corresponding packet can be verified even though it is linear combinations of vectors originated from different sources. Then, using random or established coefficients, the node makes linear combinations of the remaining data packets and produces a signature for the encoded packet based on the received signatures without accessing the private keys. Briefly, an adversary's attack is successful if it can generate a forged data packet $(Id^*, \mathbf{w}^*, \sigma^*)$ that makes verification algorithm output 1 using public keys, while either Id^* is an invalid file ID or Id^* is valid but $\mathbf{w}^*$ is not in the domain of files.

Then, we describe the algorithms of a multisource homomorphic network coding signature scheme based on the signature scheme CFW in [1] as follows:

Public System Parameters$(\lambda) \to param$. Choose a bilinear map e: $\mathbb{G} \times \mathbb{G}' \to \mathbb{G}_{\mathbb{T}}$, where $\mathbb{G}, \mathbb{G}', \mathbb{G}_{\mathbb{T}}$ are bilinear groups of prime order $p > 2^{\lambda}$, while g is a generator of $\mathbb{G}$ and g' is a generator of $\mathbb{G}'$. Randomly choose elements $h, h_1, \ldots, h_n, g_1, \ldots, g_m$ from $\mathbb{G}$. The algorithm outputs system parameters $param = (p, g, g', h, h_1, \ldots, h_n, g_1, \ldots, g_m)$ as public.

NetKG$(i, m, n) \to (pk_i, sk_i)$. This algorithm is run by each ith source node for setting up its own key pair. The ith source node randomly chooses $a_i \in \mathbb{F}_p$ to set its private key as $sk_i = a_i$ and public key as $k_i = P_i = (g')^{a_i}$. The algorithm outputs the key pair (pk_i, sk_i).

NetSign$(sk_i, Id, \mathbf{w}^{(i)}) \to \sigma$. Each ith source node signs its data packet as if in the single-source signature scheme CFW, but their secret key is different. For the packet vector $\mathbf{w}^{(i)} = (u_1^{(i)}, \ldots, u_m^{(i)}, v_1^{(i)}, \ldots, v_n^{(i)}) \in \mathbb{F}_p^{m+n}$, the ith source computes its signature as follows:

(i) Let $Id \in \mathbb{F}_p^*$ and $s_i \in \mathbb{F}_p$;

(ii) Compute $X_i = (h^{s_i} \prod_{j=1}^{m} h_j^{u_j} \prod_{j=1}^{m} g_j^{v_j})^{1/(a_i+Id)}$ and output its signature $\sigma = (X^{(1)}, \ldots, X^{(i)}, \ldots, X^{(t)}, s)$, where $X^{(i')} = 1$ for each $i' \neq i$.

Note that, for the ith source node, the rest packets in $\mathbf{w} = \{\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(t)}\}$ other than $\mathbf{w}^{(i)}$ are all set as zero vectors before forwarding.

NetEval$(Id, \{\mathbf{w}_j, \alpha_j, \sigma_j\}_{j=1}^{\mu}) \to \sigma$. If an intermediate node has received μ valid packet sets $\mathbf{w}_j = (u_{j,1}^{(i)}, \ldots, u_{j,m}^{(i)}, v_{j,1}^{(i)}, \ldots, v_{j,n}^{(i)})$ from the same Id, of which signature is $\sigma_j = (X_j^{(1)}, \ldots, X_j^{(t)}, s_j)$. Each coefficient $\alpha_j \in \mathbb{F}_p$ is determined by the intermediate node. The combined signature of this node is also a $t+1$-dimension vector $\sigma = (X^{(1)}, \ldots, X^{(t)}, s)$, where $s = \sum_{i=1}^{t} \sum_{j=1}^{\mu} \alpha_j s_j \bmod p$ and each $X^{(i)} = \prod_{j=1}^{\mu} (X_j^{(i)})^{\alpha_j}$. This algorithm outputs the combined signature σ. Then, the modified packet $\mathbf{w}$ is forwarded to other nodes along with its signature σ.

NetVer$(\{pk_i\}_{i=1}^{t}, Id, \mathbf{w}, \sigma) \to accept$. Taking public keys $\{pk_i\}_{i=1}^{t}$, file ID Id, data packet $\mathbf{w}$, and the corresponding signature σ as input, the verification algorithm outputs

1, if $\prod_{i=1}^{t} e(X^{(i)}, P_i(g')^{Id}) = e(h^s \prod_{j=1}^{m} h_j^{u_j} \prod_{j=1}^{n} g_j^{v_j}, g')$ and 0, otherwise.

5. Analysis

5.1. Correctness. According to the definition in Section 3, we analyze the correctness of the proposed scheme.

Theorem 4. *The proposed multisource homomorphic network coding signature scheme is correct.*

Proof. For each $i \in \{1, \ldots, t\}$, the ith original vector is denoted as $\mathbf{w}^{(i)} = (u_1^{(i)}, \ldots, u_m^{(i)}, v_1^{(i)}, \ldots, v_n^{(i)}) \in \mathbb{F}_p^{m+n}$. There is $\prod_{i=1}^{t} e(X^{(i)}, P_i(g')^{Id}) = e(X^{(i)}, P_i(g')^{Id}) = e((h^{s_i} \prod_{j=1}^{m} h_j^{u_j} \prod_{j=1}^{m} g_j^{v_j})^{1/(a_i+Id)}, (g')^{a_i+Id}) = e(h^s \prod_{j=1}^{m} h_j^{u_j} \prod_{j=1}^{n} g_j^{v_j}, g')$. Thus, the verification result on a valid original signature σ is 1.

On the other hand, in a modified packet, $\mathbf{w} = \sum_{j=1}^{\mu} \alpha_j \mathbf{w}_j$ and $s = \sum_{i=1}^{t} \sum_{j=1}^{\mu} \alpha_j s_j \bmod p$. There is $\prod_{i=1}^{t} e(X^{(i)}, P_i(g')^{Id}) = \prod_{i=1}^{t} e((h^{s_i} \prod_{j=1}^{m} h_j^{u_j} \prod_{j=1}^{m} g_j^{v_j})^{1/(a_i+Id)}, (g')^{a_i+Id}) = e(h^s \prod_{j=1}^{m} h_j^{u_j} \prod_{j=1}^{n} g_j^{v_j}, g')$. Thus, The verification result on a combined signature σ is 1.

Therefore, algorithms in the proposed is correct. □

5.2. Security. Then, we give the proof that the signatures in the scheme is unforgeable according to the definition in Section 3.

Theorem 5. *The proposed multisource homomorphic network coding signature scheme is secure under the q-SDH assumption.*

Proof. If an adversary has a PPT algorithm $\mathscr{B}^*$ which can forge the valid signature for a data packet, $\mathscr{B}^*$ can be used to construct an efficient algorithm $\mathscr{B}$ to forge valid signatures the CFW signature scheme.

The public system parameters are chosen as follows: $\mathbb{G}, \mathbb{G}', \mathbb{G}_{\mathbb{T}}$ are bilinear groups of prime order $p > 2^{\lambda}$ and e is a bilinear map: $\mathbb{G} \times \mathbb{G}' \to \mathbb{G}_{\mathbb{T}}$; g is a generator of $\mathbb{G}$ and g' is a generator of $\mathbb{G}'$; $h, h_1, \ldots, h_n, g_1, \ldots, g_m$ are random factors.

The algorithm $\mathscr{B}^*$ takes the public system parameters $param$, the public keys $\{pk_i\}$ a valid identifier Id, and each packet $\mathbf{w}$ with its signature as input. According to the received data signed packets $(Id, \{\mathbf{w}_j, \alpha_j, \sigma_j\}_{j=1}^{\mu})$, algorithm $\mathscr{B}^*$ tries to output a forged signed packet $(Id^*, \{\alpha_j^*\}, \mathbf{w}^*, \sigma^*)$ which makes verification algorithm output 1. That is, either $Id^* \neq Id$ or $Id^* = Id$ and $\mathbf{w}^* \neq \sum \alpha_j^* \mathbf{w}_j$.

Then, based on $\mathscr{B}^*$ algorithm, we construct another PPT algorithm $\mathscr{B}$ which can produce a forged signature for packets in the CFW signature scheme.

We assume that the first-source node uses (a, P) as its private and public keys, i.e., $sk_1 = a$ and $pk_1 = P$. Other $t-1$ source nodes whose private and public keys are generated as follows:

(i) Randomly select $t-1$ numbers from $\mathbb{F}_p$: $x_1, \ldots, x_{t-1}$;

(ii) Set the private key of the ith source $a_i = x_{i-1}(a + Id) - Id$ and its public key $P_i = (g')^{a_i}$;

Using the **NetSign** algorithm, each source node outputs its signature.

A forged signature packet $(Id^*, \{\alpha_j^*\}, \mathbf{w}^*, \sigma^*)$ for CFW is output, where $\sigma^* = (X^{(1)*}, \ldots, X^{(t)*}, s^*)$ and $s^* = \prod_{j=1}^{\mu} \alpha_j^* s_i^* \bmod p$.

It is easy to get

$$
\begin{aligned}
& e\left(\prod_{i=1}^{t} X^{i*}, P\cdot\left(g'\right)^{Id}\right) \\
&= e\left(\left(X^{(1)*}\right)\prod_{i=2}^{t}\left(X^{(i)*}\right)^{\alpha_i^* x_{i-1}}, P\cdot\left(g'\right)^{Id}\right) \\
&= e\left(\left(X^{(1)*}\right)^{\alpha_1^*}, P\cdot\left(g'\right)^{Id}\right) \\
&\quad\cdot e\left(\prod_{i=2}^{t}\left(X^{(i)*}\right)^{\alpha_i^* x_{i-1}}, P\cdot\left(g'\right)^{Id}\right) \\
&= e\left(\left(X^{(1)*}\right)^{\alpha_1^*}, P\cdot\left(g'\right)^{Id}\right) \\
&\quad\cdot e\left(\prod_{i=2}^{t}\left(X^{(i)*}\right)^{\alpha_i^*}, P\cdot\left(g'\right)^{Id}\right)^{x_{i-1}} \\
&= e\left(\left(X^{(1)*}\right)^{\alpha_1^*}, P\cdot\left(g'\right)^{Id}\right) \\
&\quad\cdot e\left(\prod_{i=2}^{t}\left(X^{(i)*}\right)^{\alpha_i^*}, \left(P\cdot\left(g'\right)^{Id}\right)^{x_{i-1}}\right) \\
&= e\left(\left(X^{(1)*}\right)^{\alpha_1^*}, P\cdot\left(g'\right)^{Id}\right) \\
&\quad\cdot e\left(\prod_{i=2}^{t}\left(X^{(i)*}\right)^{\alpha_i^*}, \left(g'\right)^{a_i+Id}\right) \\
&= e\left(\left(X^{(1)*}\right)^{\alpha_1^*}, P\cdot\left(g'\right)^{Id}\right) \\
&\quad\cdot e\left(\prod_{i=2}^{t}\left(X^{(i)*}\right)^{\alpha_i^*}, P_i\cdot\left(g'\right)^{Id}\right) \\
&= e\left(\prod_{i=1}^{t}\left(X^{(i)*}\right)^{\alpha_i^*}, P_i\cdot\left(g'\right)^{Id}\right) \\
&= e\left(h^{s^*}\prod_{j=1}^{n} h_j^{u_j}\prod_{j=1}^{m} g_j^{v_j}, g'\right).
\end{aligned}
\tag{2}
$$

Since either $Id^* \neq Id$, or $\mathbf{w}^* \neq \sum \alpha_j^* \mathbf{w}_j$, the forged signature is valid.

However, the CFW signature scheme is secure under q-SDH assumption. Therefore, the proposed multisource homomorphic network coding signature scheme is secure under the q-SDH assumption. □

6. Conclusion

In this paper, to give a solution for authentication of network coding, we propose the multisource homomorphic network coding signature in the standard model and show that the signature scheme is security under q-SDH assumption holds. The proposed scheme can effectively guarantee the availability in a multisource IoT system.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Natural Science Foundation of Guangdong Province for Distinguished Young Scholars (2014A030306020), Guangzhou Scholars Project for Universities of Guangzhou (no. 1201561613), Science and Technology Planning Project of Guangdong Province, China (2015B010129015), the National Natural Science Foundation of China (no. 61472091), and the National Natural Science Foundation for Outstanding Youth Foundation (no. 61722203).

References

[1] D. Catalano, D. Fiore, and B. Warinschi, "Efficient network coding signatures in the standard model," in *Public key cryptography*, vol. 7293, pp. 680–696, Springer, 2012.

[2] W. Chen, H. Lei, J. Li, C. Gao, F. Li, and K. Qi, "A multisource homomorphic network coding signature in the standard model," in *Proceedings of the International Conference on Green, Pervasive and Cloud Computing*, pp. 66–74, 2017.

[3] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network information flow," *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[4] S. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[5] J. Zhou, M. Tang, Y. Tian et al., "Social network and tag sources based augmenting collaborative recommender system," *IEICE Transaction on Information and Systems*, vol. 98, no. 4, pp. 902–910, 2015.

[6] S. Xie and Y. Wang, "Construction of tree network with limited delivery latency in homogeneous wireless sensor networks," *Wireless Personal Communications*, vol. 78, no. 1, pp. 231–246, 2014.

[7] Z. Huang, S. Liu, X. Mao, K. Chen, and J. Li, "Insight of the protection for data security under selective opening attacks," *Information Sciences*, vol. 412-413, pp. 223–241, 2017.

[8] J. Li, Y. K. Li, X. Chen, P. P. C. Lee, and W. Lou, "A Hybrid Cloud Approach for Secure Authorized Deduplication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1206–1216, 2015.

[9] X. Zhang, Y. Tan, C. Liang, Y. Li, and J. Li, "A Covert Channel Over VoLTE via Adjusting Silence Periods," *IEEE Access*, vol. 6, pp. 9292–9302, 2018.

[10] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An ID-based linearly homomorphic signature scheme and its application in blockchain," *IEEE Access*, vol. 6, pp. 20632–20640, 2018.

[11] Q. Lin, J. Li, Z. Huang, W. Chen, and J. Shen, "A short linearly homomorphic proxy signature scheme," *IEEE Access*, vol. 6, pp. 12966–12972, 2018.

[12] D. Xie, X. Lai, X. Lei, and L. Fan, "Cognitive Multiuser Energy Harvesting Decode-and-Forward Relaying System with Direct Links," *IEEE Access*, vol. 6, pp. 5596–5606, 2018.

[13] L. Fan, X. Lei, N. Yang, T. Q. Duong, and G. K. Karagiannidis, "Secure Multiple Amplify-and-Forward Relaying with Cochannel Interference," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 8, pp. 1494–1505, 2016.

[14] L. Fan, X. Lei, N. Yang, T. Q. Duong, and G. K. Karagiannidis, "Secrecy Cooperative Networks with Outdated Relay Selection over Correlated Fading Channels," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7599–7603, 2017.

[15] W. Meng, E. W. Tischhauser, Q. Wang, Y. Wang, and J. Han, "When Intrusion Detection Meets Blockchain Technology: A Review," *IEEE Access*, vol. 6, pp. 10179–10188, 2018.

[16] J. Xu, L. Wei, Y. Zhang, A. Wang, F. Zhou, and C. Gao, "Dynamic Fully Homomorphic encryption-based Merkle Tree for lightweight streaming authenticated data structures," *Journal of Network and Computer Applications*, vol. 107, pp. 113–124, 2018.

[17] H. Tian, Z. Chen, C. Chang et al., "Public audit for operation behavior logs with error locating in cloud storage," *Soft Computing*, pp. 1–14, 2018.

[18] J. Shen, Z. Gui, S. Ji, J. Shen, H. Tan, and Y. Tang, "Cloud-aided lightweight certificateless authentication protocol with anonymity for wireless body area networks," *Journal of Network and Computer Applications*, vol. 106, pp. 117–123, 2018.

[19] J. Shen, T. Zhou, X. Chen, J. Li, and W. Susilo, "Anonymous and Traceable Group Data Sharing in Cloud Computing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 4, pp. 912–925, 2018.

[20] M. N. Krohn, M. J. Freedman, and D. Mazières, "On-the-fly verification of rateless erasure codes for efficient content distribution," in *Proceedings of the 2004 IEEE Symposium on Security and Privacy*, pp. 226–239, May 2004.

[21] C. Gkantsidis and P. Rodriguez, "Cooperative security for network coding file distribution," in *Proceedings of the INFOCOM*, vol. 3, 2006.

[22] R. Johnson, D. Molnar, D. Song, and D. Wagner, "Homomorphic signature schemes," in *Topics in cryptology-CT-RSA*, vol. 2271, pp. 244–262, Springer, Berlin, 2002.

[23] S. Agrawal, D. Boneh, X. Boyen, and D. M. Freeman, "Preventing pollution attacks in multi-source network coding," in *Public Key Cryptography-PKC*, vol. 6056, pp. 161–176, Springer, Berlin, Germany, 2010.

[24] A. Yun, J. H. Cheon, and Y. Kim, "On homomorphic signatures for network coding," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 59, no. 9, pp. 1295-1296, 2010.

[25] Y. Wang, "Insecure provably secure network coding and homomorphic authentication schemes for network coding," *IACR Cryptology ePrint Archive*, vol. 2010, p. 60, 2010.

[26] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan, "An efficient signature-based scheme for securing network coding against pollution attacks," in *Proceedings of the 27th IEEE Communications Society Conference on Computer Communications (INFOCOM '08)*, pp. 2083–2091, April 2008.

[27] H. J. Kang, A. Yun, E. Y. Vasserman, H. T. Lee, J. H. Cheon, and Y. Kim, "Secure network coding for a p2p system," in *Proceedings of the ACM Conference on Computer and Communications Security*, Poster, 2009.

[28] F. Zhao, T. Kalker, M. Médard, and K. J. Han, "Signatures for content distribution with network coding," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT '07)*, pp. 556–560, Nice, France, June 2007.

[29] D. Charles, K. Jain, and K. Lauter, "Signatures for network coding," in *Proceedings of the 2006 40th Annual Conference on Information Sciences and Systems, CISS 2006*, pp. 857–863, usa, March 2006.

[30] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin, "Secure network coding over the integers," in *Public Key Cryptography-PKC 2010*, vol. 6056 of *Lecture Notes in Comput. Sci.*, pp. 142–160, Springer, Berlin, Germany, 2010.

[31] D. Boneh, D. Freeman, J. Katz, and B. Waters, "Signing a linear subspace: signature schemes for network coding," in *Public Key Cryptography-PKC 2009*, vol. 5443 of *Lecture Notes in Comput. Sci.*, pp. 68–87, Springer, Berlin, Germany, 2009.

[32] D. Boneh and D. M. Freeman, "Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures," in *Public Key Cryptography-PKC 2011*, vol. 6571 of *Lecture Notes in Comput. Sci.*, pp. 1–16, Springer, 2011.

[33] A. Lewko and B. Waters, "New techniques for dual system encryption and fully secure HIBE with short ciphertexts," in *Theory of Cryptography*, vol. 5978 of *Lecture Notes in Comput. Sci.*, pp. 455–479, Springer, 2010.

[34] N. Attrapadung and B. t. Libert, "Homomorphic network coding signatures in the standard model," in *Public Key Cryptography-PKC 2011*, vol. 6571, pp. 17–34, Springer, 2011.

[35] D. Catalano, D. Fiore, and B. Warinschi, "Adaptive pseudo-free groups and applications," in *Advances in Cryptology-EUROCRYPT*, vol. 6632 of *Lecture Notes in Comput. Sci.*, pp. 207–223, Springer, 2011.

[36] W. Chen, H. Lei, and K. Qi, "Lattice-based linearly homomorphic signatures in the standard model," *Theoretical Computer Science*, vol. 634, pp. 47–54, 2016.

[37] D. M. Freeman, "Improved Security for Linearly Homomorphic Signatures: A Generic Framework," in *Public Key Cryptography – PKC 2012*, vol. 7293 of *Lecture Notes in Computer Science*, pp. 697–714, Springer, 2012.

[38] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Advances in Cryptology-EUROCRYPT 2004*, vol. 3027, pp. 56–73, Springer, 2004.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 2595273, 13 pages
https://doi.org/10.1155/2018/2595273

Research Article

New Certificateless Aggregate Signature Scheme for Healthcare Multimedia Social Network on Cloud Environment

Libing Wu,[1,2] Zhiyan Xu,[1,3] Debiao He,[1,4] and Xianmin Wang[5]

[1]*The Computer School, Wuhan University, Wuhan, China*
[2]*The Co-Innovation Center for Information Supply & Assurance Technology, Anhui University, Hefei, China*
[3]*The College of Computer, Hubei University of Education, Wuhan, China*
[4]*The State Key Laboratory of Cryptology, Beijing, China*
[5]*The School of Computer Science and Educational Software, Guangzhou University, Guangzhou, China*

Correspondence should be addressed to Zhiyan Xu; csxzy@whu.edu.cn

Received 2 March 2018; Accepted 29 April 2018; Published 13 June 2018

Academic Editor: Ilsun You

With the application of sensor technology in the field of healthcare, online data sharing in healthcare industry attracts more and more attention since it has many advantages, such as high efficiency, low latency, breaking the geographical location, and time constraints. However, due to the direct involvement of patient health information, the privacy and integrity of medical data have become a matter of much concern to the healthcare industry. To retain data privacy and integrity, a number of digital signature schemes have been introduced in recent years. Unfortunately, most of them suffer serious security attacks and do not perform well in terms of computation overhead and communication overhead. Very recently, Pankaj Kumar *et al.* proposed a certificateless aggregate signature scheme for healthcare wireless sensor network. They claimed that their signature scheme was able to withstand a variety of attacks. However, in this paper, we find that their scheme fails to achieve its purpose since it is vulnerable to signature forgery attack and give the detailed attack process. Then, we propose a new certificateless aggregate signature scheme to fix the security flaws and formally prove that our proposed scheme is secure under the computationally hard Diffie-Hellman assumption. Security analysis and performance evaluation demonstrate that the security of our proposal is improved while reducing the computation cost. Compared with Pankaj Kumar *et al.*'s scheme, our proposed scheme is more efficient and suitable for the healthcare wireless sensor networks (HWSNs) to maintain security at various levels.

1. Introduction

Wireless sensor network (WSN) has been widely used in many fields such as retail, entertainment, medicine, tourism, industry, and emergency management [1], and it provides many new opportunities for traditional applications, of which healthcare is one of them. Researchers have invented many sensor-based miniature medical devices to replace the traditional thousands of wires connected to hospital equipment and to increase the mobility of devices. The combination of computer network technology and medical field makes the healthcare industry have more broad prospects for development [2].

The application of wireless sensor network technology is mainly divided into two categories: medical applications and nonmedical applications [3]. There are two main types of devices used in medical applications: wearable devices and implanted devices. The first category refers to medical devices that are used on or near the surface of a human body, and the human body can move with the wearable devices. The second category refers to medical devices injected in/with the human body.

As shown in Figure 1, there is a general healthcare wireless sensor network (HWSN) architecture, which consists of the following five components [4]: sensor, central control unit, patient, cloud based network, and healthcare professional. The medical sensor node implanted on the patient's body, using air as a transmission medium, can transmit patient's health data to a remote central control unit (CCU) for further processing, then the health data is sent to the healthcare

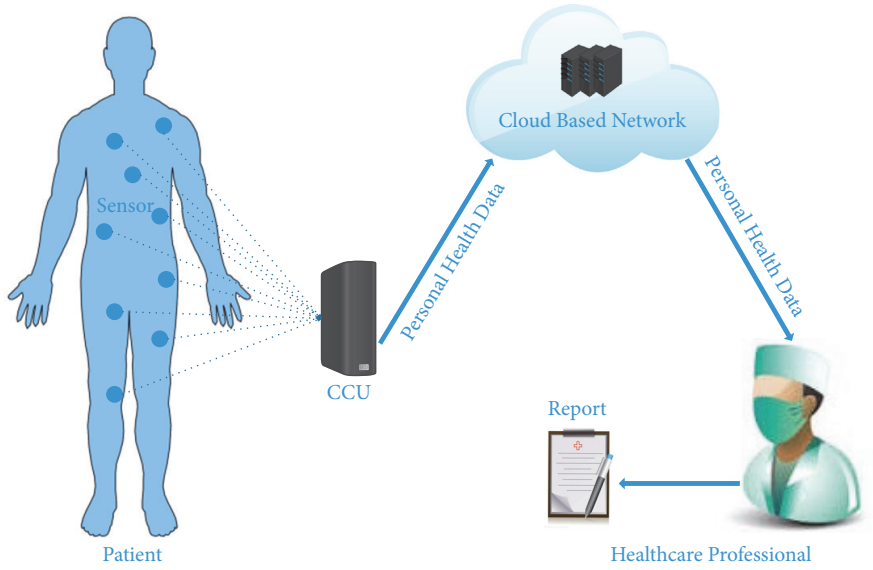


Figure 1: A general healthcare wireless sensor network architecture.

professional by CCU via Internet, and the patient's medical report is further generated.

In the HWSN, information is transmitted from medical sensor devices to the healthcare professional who analyzes the medical information and further provides a suitable solution. If the attacker modifies the medical message halfway, the healthcare professional could make a wrong diagnosis based on the modified message, which can be very dangerous to human life. Because of the direct involvement of patient health information, it is of crucial importance to address the issue of privacy and integrity of personal health data [5–7].

Motivated with the above scenario, many digital signature schemes are proposed for healthcare wireless sensor network (HWSN) to protect the privacy and integrity of patient medical information. In this paper, we first review Pankaj Kumar*et al.*'s certificateless aggregate signature (CL-AS) scheme [8] and point out a previously undiscovered security flaw in the scheme; that is, we reveal that their proposed scheme suffers the signature forgery attack. We then propose a new CL-AS scheme for the issues of security and privacy in HWSN.

1.1. Our Research Contributions. In this paper, we propose a new CL-AS scheme which could better protect the integrity and privacy of data in HMSN. The main contributions of this paper are summarized as below:

(i) *Firstly*, we identify a security weakness against Pankaj Kumar *et al.*'s CL-AS scheme for HWSN.

(ii) *Secondly*, we redefine the architecture of a HWSN, which is more close to the actual application environment.

(iii) *Thirdly*, we propose a CL-AS scheme for HWSN to mend this security flaw, and our new scheme can satisfy the security requirements.

(iv) *Finally*, we prove the security of our proposed CL-AS scheme and show that it can improve the security while reducing the computation cost compared with Pankaj Kumar *et al.*'s CL-AS scheme.

1.2. Organization of the Paper. The remainder of this paper is organized as below. Section 2 introduces the related work. Section 3 presents the problem statements associated with this paper and then briefly reviews the CL-AS scheme for HWSN in Section 4. In Section 5, we demonstrate an attack against Pankaj Kumar *et al.*'s CL-AS scheme for the HWSN. Furthermore, we present details of the proposed CL-AS scheme in Section 6. In Sections 7 and 8, the security proof and performance analysis of our scheme are described later. Finally, we give a summary of this paper in the last section.

2. Related Work

In the traditional PKI-based public key cryptography (PKC), as the number of users increases, PKC will face a variety of certificate management issues such as certificate distribution, storage, revocation, and high computational cost [11].

Although identity-based public key cryptography (IBC) [12, 13] can solve the problem of certificate management existing in PKC, it has inherent key escrow issue. This is because the user's private key is generated by the key generation center (KGC) based on the user's identity; that is, KGC can access any user's private key in IBC.

To solve the above problems, Al-Riyami *et al.* proposed certificateless public key cryptosystem (CL-PKC) [14]. Because it does not use certificates and the private key is generated both by KGC and by the user himself, it can solve certificate management issue in PKC and the key escrow issue in IBC. Since Al-Riyami *et al.* introduced the notion of CL-PKC [14], it has attracted more and more research attention, and many certificateless signature (CLS) schemes [15–21] have been introduced by researchers.

Huang *et al.* [15] proved that the CLS scheme proposed in [14] is vulnerable to the public key replacement attack and further proposed an improved certificateless signature scheme to solve this weakness. Similarly, Li *et al.* [16] also proposed a new CLS scheme to improve the security of the scheme proposed in [17], which is subject to the public key replacement attack as well. For a malicious KGC attack that exists in some certificateless signature schemes, Au *et al.* [18] proposed an enhanced security model that allows malicious KGC to generate key pairs in any way. Nevertheless, the certificateless encryption and signature schemes proposed in [19–21] have been found to be insecure against malicious KGC attack.

Boneh *et al.* proposed the concept of aggregate signature [22] in Eurocrypt 2003. The aggregator can aggregate n different signatures with respect to n messages from n users into a single short signature, which can reduce the bandwidth and computational effort of tiny devices used in HWSN. Therefore, the aggregate signature is a more suitable choice in resource-constrained HWSN.

Combining certificateless public key cryptography with aggregate signature, Gong *et al.* [9] proposed the first CL-AS scheme, but they did not give a formal security proof to the scheme. After pioneer work [9], many CL-AS schemes [10, 23–28] have been proposed for various practical applications. Zhang and Zhang [23] redefined the concept and security model for CL-AS. Furthermore, they put forward a new CL-AS scheme, but their scheme need clock synchronized while generating the aggregate signature, and, more seriously, the scheme has been proved that it cannot resist malicious KGC attack. Xiong *et al.* [24] presented a CL-AS scheme, but He *et al.* [25] showed that their scheme was forgeable and further proposed a new CL-AS scheme. The CL-AS scheme proposed in [10] has been found to be insecure against the malicious-but-passive KGC attack by the researchers in [26–28].

Recently, He and Zeadally [29] present an authentication scheme for the Ambient Assisted Living (AAL) system, which provides technical support for medical monitoring and telehealth services. He *et al.* [30] put forward an efficient certificateless public auditing scheme for cloud-assisted wireless body area networks. Very recently, Pankaj Kumar *et al.* proposed a CL-AS scheme for secure communication in HWSN [8], which is claimed to be able to achieve the message authentication and integrity audit functions while also achieving nonrepudiation and confidentiality. Unfortunately, we find that their CL-AS scheme is insecure and vulnerable to signature forgery attack from a malicious-but-passive KGC.

3. Problem Statement

Bilinear map and related hard problems are first described and then system model of our proposed CL-AS scheme is presented in this section. After that, system components of CL-AS scheme are also described.

3.1. Bilinear Map. Suppose that G_1 and G_2 are two cyclic groups with the same prime order q, where G_1 is an additive cyclic group with a generator P and G_2 is a multiplicative cyclic group. $e : G_1 \times G_1 \rightarrow G_2$ is a bilinear map. For all $P, Q, T \in G_1$, $a, b \in Z_q^*$, and e should satisfy the properties as follows:

(1) Bilinearity: $e(P, Q + T) = e(P, Q)e(P, T)$ and $e(aP, bQ) = e(abP, Q) = e(P, abQ)$.

(2) Nondegeneracy: there exists $P \in G_1$ such that $e(P, P) \neq 1$.

(3) Computability: there exists efficient algorithm to calculate $e(P, Q)$.

3.2. Complexity Assumption

(1) *Computational Diffie-Hellman (CDH) Problem:* Given a generator P of an additive cyclic group G_1 with the order q and a random instance (aP, bP), it is difficult to compute abP, where a and b are unknown.

(2) *Computational Diffie-Hellman (CDH) Assumption:* There does not exist adversary A, can solve the *CDH* problem in probabilistic polynomial time with a nonnegligible probability ϵ, where $\epsilon > 0$ is a very small number.

3.3. System Model. The architecture of our healthcare wireless sensor network is shown in Figure 2. There are five entities in the framework of a healthcare wireless sensor network: medical sensor node (MSN), medical server (MS), authorized healthcare professional (AHP), signature aggregator (SA), and aggregate signature verifier (ASV). Each entity is specifically defined as follows:

(1) *Medical sensor node.* Medical sensor node is a resource-limited medical small device on patient's body belonging to the Care-District. Let ID_i denote the identity and (sk_i, pk_i) denote the key pair of the sensor node. Each sensor node can use its private key to generate a signature for the relevant message and send the signature to the signature aggregator.

(2) *Medical server.* Medical server is a device with strong computing power and plenty of storage space, which can handle a large amount of data received from sensors. It transmits the processed patient's medical information to the AHP. In addition, it is responsible for generating system parameters *params*, its own key pair (s, MS_{pub}), and the partial private key ppk_i for each sensor node corresponding to its identity and then secretly sends ppk_i to the sensor node.

(3) *Healthcare professional.* Healthcare professional refers to an authorized medical personnel who provides patients with appropriate prescriptions by analyzing the data information sensed by the sensors.

(4) *Aggregator.* Aggregator refers to a certain computing power of device. It is responsible for collecting a single signature from Care-District and then generating an aggregate signature and sending it to the MS. Suppose that each Care-District contains one aggregator and many sensors.

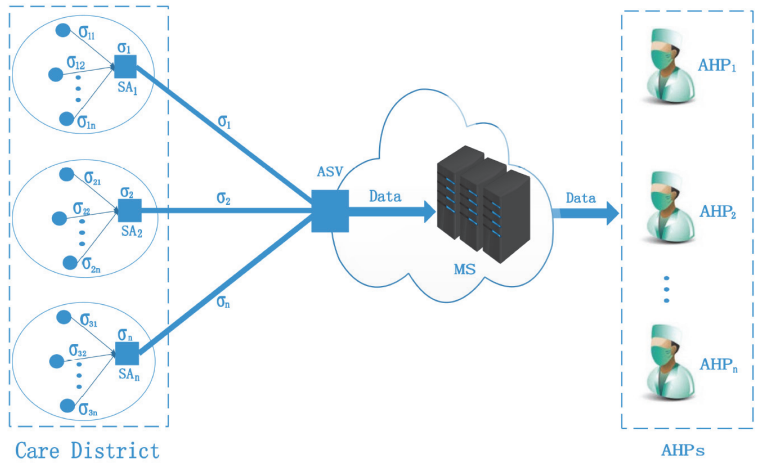


Figure 2: The architecture of our healthcare wireless sensor network.

(5) *Aggregate signature verifier.* Aggregate signature verifier refers to a certain computing power of equipment. It is responsible for verifying an aggregate signature from different Care-District and then outputting a verification result.

3.4. System Components. Our CL-AS scheme is a collection of the following seven polynomial time algorithms as below:

(1) $Setup(1^k) \rightarrow (params, s, MS_{pub})$ is a probabilistic algorithm executed by the MS, where k is a security parameter, $params$ is the system parameters, (s, MS_{pub}) is the key pair of MS, that is, s is the master secret key, and MS_{pub} is the master public key.

(2) $Partial\text{-}Private\text{-}Key\text{-}Gen\ (params, s, MS_{pub}ID_i) \rightarrow ppk_i$ is a probabilistic algorithm executed by the MS, where $params$ is the system parameters, (s, MS_{pub}) is the key pair of MS, $ID_i \in \{0,1\}^*$ is a MSN's identity, and ppk_i is the partial private key corresponding to the identity ID_i of the MSN.

(3) $User\text{-}Key\text{-}pair\text{-}Gen(params, ppk_i) \rightarrow (sk_i, pk_i)$ is a randomized algorithm executed by the MSN with identity ID_i, where $params$ is the system parameters, (s, MS_{pub}) is the key pair of MS, and (sk_i, pk_i) is the key pair of the MSN with the identity ID_i.

(4) $Sign(params, (sk_i, pk_i), \Delta, ID_i, m_i) \rightarrow \sigma_i$ is a randomized algorithm executed by the signer, where $params$ is the system parameters, (sk_i, pk_i) is the key pair of the signer, Δ is the state information, ID_i is the signer's identity, m_i is the message, and σ_i is the signature on the message m_i.

(5) $Verify(params, \Delta, ID_i, m_i, pk_i, \sigma_i) \rightarrow$ {"0", "1"} is a probabilistic algorithm executed by the verifier, where $params$ is the system parameters, ID_i is the signer's identity, pk_i is the public key of the signer, m_i is the message, and σ_i is the signature on the message m_i, 1 or 0 as outputs to indicate whether the signature σ_i is validated.

(6) $Aggregate(params, ID_i, m_i, pk_i, \sigma_i)_{1\le i\le n} \rightarrow \sigma$ is a deterministic algorithm executed by the aggregator, where $params$ is the system parameters, ID_i is the signer's identity, pk_i is the public key of the signer, m_i is the message, σ_i is the signature on the message m_i, and σ_i is the signature on the message m_i.

(7) $Aggregate\text{-}Verify\ (params, \Delta, ID_i, m_i, pk_i, \sigma)_{1\le i\le n} \rightarrow$ {"0", "1"} is a deterministic algorithm executed by the aggregate verifier, where $params$ is the system parameters and σ is the aggregate signature of the message m_i on the identity ID_i with public key pk_i. 1 or 0 act as outputs to indicate whether aggregate signature σ is validated.

3.5. Attack Model. In the attack model, we introduce an adversary $A \in \{A_1, A_2\}$ in our model. A's ultimate goal is to successfully forge the user's signature on the message. A possesses with the following capabilities:

(1) A can access any hash oracle and corresponding queries in the security model.

(2) A_1 simulates an outsider attacker, who cannot obtain the master key but can replace any user's public key with a value of his choice.

(3) A_2 simulates an honest-but-curious MS, who is an insider attacker and has no power to replace any user's public key but can access the system master key.

4. Review of Pankaj Kumar *et al.*'s Scheme

Pankaj Kumar *et al.*'s CL-AS scheme is composed of seven algorithms, i.e., $Setup$, $Partial-Private-Key-Gen$, $Private-Key-Gen$, $Sign$, $Verify$, $Aggregate$, and $Aggregate-Verify$. The scheme details are described as below.

4.1. Setup. By executing the following operations, after entering the security parameter k, the MS generates the system parameter $params$.

(1) Generates two cyclic groups G_1 and G_2 with the same order q, where q is a prime. P being a generator of G_1. $e : G_1 \times G_1 \to G_2$ being a bilinear pairing.

(2) Randomly selects a number $\alpha \in Z_q^*$, computes $MS_{pub} = \alpha P$, and sets α as the master key and MS_{pub} as the public key of MS

(3) Defines three hash functions: $H_1 : \{0,1\} \to G_1$, $H_2 : \{0,1\} \to G_1$, $H_3 : \{0,1\} \to Z_q^*$

(4) Keeps α secret and $params = (G_1, G_2, P, q, e, MS_{pub}, H_1, H_2, H_3)$ public.

4.2. Partial-Private-Key-Gen. By executing the following operations, MS generates the user's partial private key:

(1) Given ID_i as the identity of a MS, it computes $Q_{ID_i} = H(ID_i)$ and $ppk_{ID_i} = \alpha.Q_{ID_i}$ and sets ppk_{ID_i} as the user's partial private key.

(2) It secretly sends ppk_{ID_i} to the corresponding MSN.

4.3. Private-Key-Gen. By executing the following operations, a sensor with the identity ID_i generates its private key and public key:

(1) Selects a random number x_i as the secret value.

(2) Sets $\{ppk_{ID_i}, x_i\}$ as its private key.

(3) Computes $PK_{ID_i} = x_i P$ as its public key.

4.4. Sign. By executing the following operations, a signer with the identity ID_i generates a signature σ_i on the message m_i:

(1) Inputs system parameters $params$, private key ppk_{ID_i}, secret key x_i, state information Δ, and private-public key pair (x_i, PK_{ID_i})

(2) Selects $r_i \in Z_q^*$ randomly and then computes $R_i = r_i P$

(3) Computes $W = H_2(\Delta)$ and $h_i = H_3(m_i, ID_i, PK_{ID_i}, R_i)$

(4) Computes $V_i = ppk_{ID_i} + r_i W + h_i x_i MS_{pub}$

(5) Outputs (R_i, V_i) as the signature of message m_i.

4.5. Verify. By executing the following operations, the verifier verifies the signature $\sigma_i = (R_i, V_i)$ of message m_i on identity ID_i:

(1) Inputs the state information Δ

(2) Computes $Q_{ID_i} = H_1(ID_i)$, $W = H_2(\Delta)$ and $h_i = H_3(m_i, ID_i, PK_{ID_i}, R_i)$

(3) Verifies

$$e(V_i, P) = e\left(Q_{ID_i} + h_i PK_{ID_i}, MS_{pub}\right) e(R_i, W). \quad (1)$$

(4) If (1) holds, emits 1 and the verifier accepts the signature σ_i; otherwise emits 0 and rejects.

4.6. Aggregate. By executing the following operations, an aggregator generates the aggregate signature σ from user-message-public key-signature pairs $(ID_i, m_i, PK_i, \sigma_i)_{1 \le i \le n}$:

(1) Inputs n tuples $(ID_i, m_i, PK_i, \sigma_i)$, where $1 \le i \le n$

(2) Computes $V = \sum_{i=1}^{n} V_i$

(3) Outputs $\sigma = (R, V)$ as the aggregate signature, where $R = \{R_1, R_1, \ldots, R_n\}$.

4.7. Aggregate-Verify. By executing the following operations, the aggregate verifier verifies the validity of the aggregate signature $\sigma = (R, V)$:

(1) Inputs the state information Δ, the tuples $(ID_i, m_i, PK_i, \sigma_i)_{1 \le i \le n}$, and the aggregate signature $\sigma = (R, V)$

(2) For $1 \le i \le n$, computes $Q_{ID_i} = H_1(ID_i)$, $W = H_2(\Delta)$, and $h_i = H_3(m_i, ID_i, PK_{ID_i}, R_i)$

(3) Verifies

$$e(V, P) = e\left(\sum_{i=1}^{n}\left(Q_{ID_i} + h_i PK_{ID_i}\right), MS_{pub}\right) e\left(\sum_{i=1}^{n} R_i, W\right). \quad (2)$$

(4) If (2) holds, emits 1 and the verifier accepts the aggregate signature σ; otherwise emits 0 and rejects.

5. Attack on Pankaj Kumar *et al.*'s CL-AS Scheme

As we know that the signature of $\sigma_i = (R_i, V_i)$ of message m_i on identity ID_i should be unforgeable. However, a malicious MS or an outside attacker may try to forge the signature. Once the MS or the outside attacker successfully forges the signature directly or indirectly, he/she finishes the signature forgery attack.

In this section, we mainly consider the type 2 adversary A_2 and first make a security analysis for Pankaj Kumar *et al.*'s CL-AS scheme, and then we demonstrate that it is vulnerable to the signature forgery attack, the attack details are described as follows.

Setup. The challenger executes the *Setup* algorithm to generate system parameters $params$ and master key α. Then it returns $params$ and α to the adversary A_2.

Queries. The adversary A_2 could get the signature σ_j on the message m_j signed by S_i with the identity ID_i via signature queries, where

$$\sigma_j = \begin{cases} R_j = r_j P \\ V_j = ppk_{ID_i} + r_j W + h_j x_i MS_{pub} \end{cases} \quad (3)$$

Forgery. In order to forge the signature σ_k^* on m_k signed by S_i with the identity ID_i, the adversary A_2 implements its attack as follows:

(1) Lets $R_k^* = r_k^* P = R_j = r_j P$

(2) Computes $h_k^* = H_3(m_k, ID_i, PK_{ID_i}, R_k^*)$

Verify. It is easy to verify the validity of the forged signature σ_k^*. The verifier calculates $Q_{ID_i} = H_1(ID_i)$ and $W = H_2(\Delta)$. Furthermore, the verifier calculates $h_k^* = H_3(m_k, ID_i, PK_{ID_i}, R_k^*)$. Then we use the forged signature σ_k^* to verify (1) and the concrete process is as follows:

$$\begin{aligned} e(V_k^*, P) &= e\left(ppk_{ID_i} + r_j W + h_k^* \alpha PK_{ID_i}, P\right) \\ &= e\left(\alpha Q_{ID_i}, P\right) e\left(r_j W, P\right) e\left(h_k^* \alpha \mathrm{PK}_{ID_i}, P\right) \\ &= e\left(Q_{ID_i}, MS_{pub}\right) e\left(r_j P, W\right) e\left(h_k^* PK_{ID_i}, MS_{pub}\right) \\ &= e\left(Q_{ID_i} + h_k^* PK_{ID_i}, MS_{pub}\right) e\left(R_k^*, W\right) \end{aligned} \tag{4}$$

Aggregate-Verify. It is easy to verify the validity of the forged signature σ^*. For $1 \le i \le n$, the verifier calculates $Q_{ID_i} = H_1(ID_i)$ and $h_k^* = H_3(m_k, ID_i, PK_{ID_i}, R_k^*)$. Furthermore, the verifier calculates $W = H_2(\Delta)$. Then we use the forged signature to verify (2); the concrete process is as follows:

$$\begin{aligned} e(V^*, P) &= e\left(\sum_{k=1}^{n}\left(ppk_{ID_i} + r_k^* W + h_k^* \alpha PK_{ID_i}\right), P\right) \\ &= e\left(\sum_{k=1}^{n}\left(\alpha Q_{ID_i} + h_k^* \alpha PK_{ID_i}\right), P\right) e\left(\sum_{k=1}^{n} r_j W, P\right) \\ &= e\left(\sum_{k=1}^{n}(_{ID_i} + h_k^* PK_{ID_i}), MS_{pub}\right) e\left(\sum_{k=1}^{n} r_j W, P\right) \\ &= e\left(\sum_{k=1}^{n}\left(Q_{ID_i} + h_k^* PK_{ID_i}\right), MS_{pub}\right) e\left(\sum_{k=1}^{n} R_k^*, W\right) \end{aligned} \tag{5}$$

We can find that the signature verifications (1) and (2) hold. That is, the forged signature pass verification and the malicious KGC can forge the signature successfully; Pankaj Kumar *et al.*'s CL-AS scheme is insecure.

6. Our Proposed CL-AS Scheme

To overcome the security flaw of the original scheme, we propose a new CL-AS scheme. Our CL-AS scheme includes seven phases: *Setup*, *Partial−Private−Key−Gen*, *Private−Key−Gen*, *Sign*, *Verify*, *Aggregate*, and *Aggregate−Verify*. The scheme details are described as below.

6.1. Setup. By executing the following operations, MS generates the system parameters after taking a security parameter k:

(1) Generates two cyclic groups G_1 and G_2 with the same order q, where q is a prime. P being a generator of G_1. $e : G_1 \times G_1 \to G_2$ being a bilinear pairing.

(2) Randomly selects a number $s \in Z_q^*$ as the master key of MS and calculates $MS_{pub} = sP$ as the public key of MS

(3) Chooses four hash functions: $H_1 : \{0,1\} \to G_1$, $H_2 : \{0,1\} \to G_1$, $h_1 : \{0,1\} \to Z_q^*$, and $h_2 : \{0,1\} \to Z_q^*$

(4) Keeps the master key s secret and the system parameters $params = (G_1, G_2, P, e, q, MS_{pub}, H_1, H_2, h_1, h_2)$ public.

6.2. Partial-Private-Key-Gen. By executing the following operations, MS generates the MSN's partial private key:

(1) Given ID_i as a MSN's identity, MS first computes $Q_i = H_1(ID_i)$ and then computes the MSN's partial private key $ppk_i = s.Q_i$.

(2) It secretly sends ppk_i to the corresponding MSN.

6.3. Private-Key-Gen. By executing the following operations, a MSN with the identity ID_i generates its private key and public key:

(1) Selects a random number x_i as the secret value.

(2) Sets $sk_i = \{ppk_i, x_i\}$ as its private key.

(3) Computes $pk_i = x_i P$ as its public key.

6.4. Sign. By executing the following operations, a signer with the identity ID_i generates a signature σ_i on the message m_i:

(1) Inputs system parameters $params$, state information Δ, and private-public key pair (sk_i, pk_i)

(2) Selects $r_i \in Z_q^*$ randomly and then calculates $R_i = r_i P$

(3) Computes $\alpha_i = h_1(ID_i, pk_i, R_i)$, $\beta_i = h_2(m_i, ID_i, pk_i, R_i)$, and $U = H_2(\Delta)$

(4) Computes $V_i = \alpha_i ppk_i + r_i MS_{pub} + \beta_i x_i U$

(5) Outputs $\sigma_i = (R_i, V_i)$ as the signature of message m_i.

6.5. Verify. By executing the following operations, the verifier verifies the signature $\sigma_i = (R_i, V_i)$ of message m_i on identity ID_i:

(1) Inputs the state information Δ.

(2) Computes $\alpha_i = h_1(ID_i, pk_i, R_i)$, $\beta_i = h_2(m_i, ID_i, pk_i, R_i)$, $Q_i = H_1(ID_i)$, and $U = H_2(\Delta)$

(3) Verifies

$$e(V_i, P) = e\left(\alpha_i Q_i + R_i, MS_{pub}\right) e\left(\beta_i pk_i, U\right) \tag{6}$$

(4) If (6) holds, emits 1 and the verifier accepts the signature σ_i; otherwise emits 0 and rejects.

6.6. Aggregate. By executing the following operations, an aggregator generates the aggregate signature σ from user-message-public key-signature pairs $(ID_i, m_i, pk_i, \sigma_i)_{1 \le i \le n}$:

(1) Inputs n tuples $(ID_i, m_i, pk_i, \sigma_i)$, where $1 \le i \le n$

(2) Computes $V = \sum_{i=1}^{n} V_i$

(3) Outputs $\sigma = (R, V)$ as the aggregate signature, where $R = \{R_1, R_1, \ldots, R_n\}$.

6.7. Aggregate-Verify. By executing the following operations, the aggregate verifier verifies the validity of the aggregate signature $\sigma = (R, V)$:

(1) Inputs the state information Δ, the tuples $(ID_i, m_i, pk_i, \sigma_i)_{1\le i\le n}$, and the aggregate signature $\sigma = (R, V)$

(2) Computes $U = H_2(\Delta)$, furthermore, for $1 \le i \le n$, computes $Q_i = H_1(ID_i)$, $\alpha_i = h_1(ID_i, pk_i, R_i)$ and $\beta_i = h_2(m_i, ID_i, pk_i, R_i)$

(3) Verifies

$$e(V, P) = e\left(\sum_{i=1}^{n}(\alpha_i Q_i + R_i), MS_{pub}\right) e\left(\sum_{i=1}^{n}\beta_i pk_i, U\right). \quad (7)$$

(4) If (7) holds, emits 1 and the verifier accepts the aggregate signature σ; otherwise emits 0 and rejects.

7. Security Analysis

A certificateless aggregate signature scheme should satisfy the following requirements: correctness and unforgeability.

7.1. Correctness

Theorem 1. *The proposed certificateless aggregate scheme is correct, if and only if the single signature and aggregate signature generated by our scheme make (1) and (2) hold. The correctness of the protocol is elaborated as follows:*

$$\begin{aligned} e(V_i, P) &= e\left(\alpha_i ppk_i + r_i MS_{pub} + \beta_i x_i U, P\right) \\ &= e(\alpha_i ppk_i, P)\, e\left(r_i MS_{pub}, P\right) e(\beta_i x_i U, P) \\ &= e(\alpha_i Q_i, sP)\, e\left(r_i P, MS_{pub}\right) e(\beta_i x_i P, U) \\ &= e\left(\alpha_i Q_i, MS_{pub}\right) e\left(r_i P, MS_{pub}\right) e(\beta_i x_i P, U) \\ &= e\left(\alpha_i Q_i + R_i, MS_{pub}\right) e(\beta_i pk_i, U) \end{aligned} \quad (8)$$

and

$$\begin{aligned} e(V, P) &= e\left(\sum_{i=1}^{n}\alpha_i ppk_i + r_i MS_{pub} + \beta_i x_i U, P\right) \\ &= e\left(\sum_{i=1}^{n}\alpha_i ppk_i, P\right) e\left(\sum_{i=1}^{n} r_i MS_{pub}, P\right) \\ &\cdot e\left(\sum_{i=1}^{n}\beta_i x_i U, P\right) = e\left(\sum_{i=1}^{n}\alpha_i Q_i, sP\right) \\ &\cdot e\left(\sum_{i=1}^{n} r_i P, MS_{pub}\right) e\left(\sum_{i=1}^{n}\beta_i x_i P, U\right) \\ &= e\left(\sum_{i=1}^{n}\alpha_i Q_i, MS_{pub}\right) e\left(\sum_{i=1}^{n} r_i P, MS_{pub}\right) \\ &\cdot e\left(\sum_{i=1}^{n}\beta_i x_i P, U\right) = e\left(\sum_{i=1}^{n}(\alpha_i Q_i + R_i), MS_{pub}\right) \\ &\cdot e\left(\sum_{i=1}^{n}\beta_i pk_i, U\right) \end{aligned} \quad (9)$$

7.2. Unforgeability. In this subsection, we first give the security model of CL-AS scheme and then give the security proof to show that the proposal is secure under the security model.

Security Model. There are two types of adversaries in the CL-AS security model: A_1 and A_2. A_1 simulates an outsider attacker, who cannot obtain the master key but can replace any user's public key with a value of his choice, while A_2 simulates an honest-but-curious KGC, who is an insider attacker and has no power to replace any user's public key but can access the system master key.

Definition 2. The security model of a CL-AS scheme is defined by two games (denoted by **Game1** and **Game2**) played between an adversary $A \in \{A_1, A_2\}$ and a challenger C; more details are defined below.

The adversary A can access the following random oracle machines in the scheme:

Hash queries: A can access any hash oracle in the scheme, including H_1, H_2, h_1, and h_2.

Setup: C performs the *Setup* algorithm to generate the master key s and the system parameter list *params*. Then C gives the corresponding response for different types of adversary.

Reveal-Partial-private-key: While A submits a partial private key query on the identity ID_i to challenger C, it checks if there is a record that corresponds to the identity ID_i in the L^{ppk} list and, if found, sends ppk_i to A; otherwise, if $ID_i = ID_{ts}$ it aborts; otherwise, it generates the partial private key ppk_i, sends it to A, and stores it in the list L^{ppk}.

Reveal-Secret-key: While A submits a secret value query on the identity ID_i to challenger C, it checks if there is a record that corresponds to the identity ID_i in the list L^x and, if found, sends x_i to A; otherwise, if $ID_i = ID_{ts}$ it aborts; otherwise, it generates the secret value x_i and sends it to A and stores it in the list L^x.

Reveal-Public-Key: When adversary A submits a public key query on the identity ID_i to challenger C, it checks if there is a record that corresponds to the identity ID_i in the list L^{pk}, if found, sends pk_i to A; otherwise it generates the public key pk_i, sends it to A and stores it in the list L^{pk}.

Replace-Public-key: While A submits a query that replaces the public key on the identity ID_i with $A's$ choice of public key pk_i^* to challenger C, C checks if there is a record that corresponds to the identity ID_i in the list L^{pk} and, if found, then it updates the corresponding item

(ID_i, x_i, pk_i, ppk_i) to $(ID_i, x_i, pk_i^*, ppk_i)$ in the list L^{pk}; otherwise it aborts.

Sign: While A submits a signature query on the message m_i with the signer's identity ID_i to challenger C, C executes one of the following operations:

(1) If the target user ID_i has not been created, it aborts.

(2) If the target user ID_i has been created and the related user public key pk_i has not been replaced, then it returns a valid signature σ_i.

(3) If the target user ID_i has been created and the corresponding user public key pk_i has been replaced with pk_i^*, then it returns a signature σ_i^*.™

We, respectively, define two games to describe two different types of attackers in the CLS, as shown below.

Game1: The challenger C interacts with adversary A_1 as follows:

(1) Inputting k as a security parameter, C performs the *Setup* algorithm to generate the master key s and the system parameter list *params*. Then C sends *params* to A_1 and keeps s secret.

(2) A_1 is capable of accessing any hash oracle in the scheme and *Reveal* − *Partial* − *Private* − *Key*, *Reveal* − *Secret* − *Key*, *Reveal* − *Public* − *Key*, *Replace* − *Public* − *Key*, and *Sign* queries at any stage during the simulation in polynomial bound.

Forgery: A_1 outputs an aggregate signature σ^* with respect to n user-message-public key-signature pairs $(ID_i^*, m_i^*, pk_i^*, \sigma_i^*)$, where $1 \le i \le n$. We say that A_1 wins *Game*1 if and only if the following conditions are met:

(1) σ^* is a valid aggregate signature with respect to user-message-public key-signature pairs $(ID_i^*, m_i^*, pk_i^*, \sigma_i^*)$, where $1 \le i \le n$.

(2) The targeted identity ID_i^* has not been submitted during the *Reveal* − *Partial* − *Private* − *Key* query.

(3) (ID_i^*, m_i^*) has not been submitted during the *Sign* query.

Game2: The challenger C interacts with adversary A_2 as follows:

(1) Inputting k as a security parameter, C performs the *Setup* algorithm to generate the master key s and the system parameter list *params*. Then C sends *params* and s to A_2.

(2) A_2 is capable of accessing any hash oracle in the scheme and *Reveal* − *Partial* − *Private* − *Key*, *Reveal* − *Public* − *Key*, and *Sign* queries at any stage during the simulation in polynomial bound.

Forgery: A_2 outputs an aggregate signature σ^* with respect to n user-message-public key-signature pairs $(ID_i^*, m_i^*, pk_i^*, \sigma_i^*)$, where $1 \le i \le n$. We say that A_2 wins *Game*2 if and only if the following conditions are met:

(1) σ^* is a valid aggregate signature with respect to user-message-public key-signature pairs $(ID_i^*, m_i^*, pk_i^*, \sigma_i^*)$, where $1 \le i \le n$.

(2) The targeted identity ID_i^* has not been submitted during the *Reveal* − *Secret* − *Key* query.

(3) (ID_i^*, m_i^*) has not been submitted during the *Sign* query.

Provable Security. In this section, we demonstrate that the new CL-AS scheme is secure under the security model described in the previous subsection. Our security proof consists of two parts.

In this section, we prove that our proposed CL-AS scheme is secure under the security model present in the previous section, and the specific process is described in the following two parts: (1) the CL-AS is unforgeable to type 1 adversary A_1 and (2) the CL-AS scheme is unforgeable to type 2 adversary A_2.

Theorem 3. *The proposed CL-AS scheme is existentially unforgeable against type 1 adversary A_1, if the CDH problem is difficult to solve in G_1.*

Proof. We can prove the unforgeability of our CL-AS scheme against type 1 adversary with **Game1** that involves A_1 and an algorithm called simulator C. □

Given a random instance of the CDH problem $(P, Q_1 = aP, Q_2 = bP)$, where P is a generator of G_1, our ultimate goal is to find the result of abP by solving the CDH problem.

(i) *Setup:* C randomly chooses ID_{ts} as the target identity of sensor challenged, sets $MS_{pub} = Q_1 = aP$, and generates and returns system parameter $params = \{G_1, G_2, P, e, q, MS_{pub}, H_1, H_2, h_1, h_2\}$ to A_1. A_1 performs the inquiries as follows:

(ii) H_1 *query:* C maintains a list denoted L^{H_1}, and the structure of L^{H_1} is $(ID_i, \delta_i, \varepsilon_i, Q_i)$; all the elements in L^{H_1} are initialized to null. When A_1 performs the query with the identity ID_i, C checks whether a tuple $(ID_i, \delta_i, \varepsilon_i, Q_i)$ exists in L^{H_1}; if it exists, it returns Q_i to A_1; otherwise, C randomly selects $\varepsilon_i \in \{0, 1\}$ and $\delta_i \in Z_q^*$. If $\varepsilon_i = 0$, set $Q_i = \delta_i P$; otherwise, if $\varepsilon_i = 1$, set $Q_i = \delta_i Q_2 = \delta_i bP$. It returns Q_i to A_1 and stores $(ID_i, \delta_i, \varepsilon_i, Q_i)$ to L^{H_1}.

(iii) H_2 *query:* C maintains a list denoted L^{H_2}, and the structure of L^{H_2} is (MS_{pub}, ϑ, U), all the elements in L^{H_2} are initialized to null. When A_1 executes the query with MS_{pub}, C checks if a tuple (MS_{pub}, ϑ, U) exists in L^{H_2}; if it exists, it returns U to A_1; otherwise, C randomly selects $\vartheta \in Z_q^*$ and computes $U = \vartheta P$. It returns U to A_1 and stores (MS_{pub}, ϑ, U) to L^{H_2}.

(iv) h_1 *query:* C maintains a list denoted L^{h_1}, and the structure of L^{h_1} is $(ID_i, pk_i, R_i, \alpha_i)$, all the elements in L^{h_1} are initialized to null. When A_1 executes the query with the tuple (ID_i, pk_i, R_i), C check whether a tuple

$(ID_i, pk_i, R_i, \alpha_i)$ exists in L^{h_1}; if it exists, it returns α_i to A_1; otherwise, C randomly selects α_i. It returns α_i to A_1 and stores $(ID_i, pk_i, R_i, \alpha_i)$ to L^{h_1}.

(v) *h_2 query:* C maintains a list denoted L^{h_2}, and the structure of L^{h_2} is $(m_i, ID_i, pk_i, R_i, \beta_i)$, all the elements in L^{h_2} are initialized to null. When A_1 executes the query with the tuple (m_i, ID_i, pk_i, R_i), C checks if a tuple $(m_i, ID_i, pk_i, R_i, \beta_i)$ exists in L^{h_2}; if it exists, it returns β_i to A_1; otherwise, C randomly selects β_i. It returns β_i to A_1 and stores $(m_i, ID_i, pk_i, R_i, \beta_i)$ to L^{h_2}.

(vi) *Reveal-Partial-Private-Key queries:* C maintains a list denoted L^{ppk}, and the structure of L^{ppk} is (ID_i, ppk_i), all the elements in L^{ppk} are initialized to null. When A_1 executes the query with ID_i, C first checks whether $ID_i = ID_{ts}$; if it holds, output $\perp$; otherwise, C checks whether a tuple (ID_i, ppk_i) exists in L^{ppk}; if it exists, it returns ppk_i to A_1; otherwise, C recalls the corresponding tuple $(ID_i, \delta_i, \varepsilon_i, Q_i)$ from the list L^{H_1} and computes $ppk_i = \delta_i MS_{pub} = a\delta_i P$. It returns ppk_i to A_1 and stores (ID_i, ppk_i) to L^{ppk}.

(vii) *Reveal-Secret-Key-queries:* C maintains a list denoted L^x, and the structure of L^x is (ID_i, x_i), all the elements in L^x are initialized to null. When A_1 performs the query with the identity ID_i, C first checks if $ID_i = ID_{ts}$; if it holds, output $\perp$; otherwise, C checks whether a tuple exists in (ID_i, x_i); if it exists, it returns x_i to A_1; otherwise, C randomly selects x_i. It returns x_i to A_1 and stores x_i to (ID_i, x_i).

(viii) *Reveal-Public-Key queries:* C maintains a list denoted L^{pk}, and the structure of L^{pk} is (ID_i, pk_i), all the elements in L^{pk} are initialized to null. When A_1 performs the query with the identity ID_i, C checks whether a tuple (ID_i, pk_i) exists in L^{pk}; if it exists pk_i, it returns pk_i to A_1; otherwise, it accesses L^x to get x_i and computes $pk_i = x_i P$. It returns pk_i to A_1 and stores (ID_i, pk_i) to L^{pk}.

(ix) *Replace-Public-Key queries:* When A_1 executes the query with the identity (ID_i, pk_i^*), in response, C replaces the real public key pk_i of ID_i with pk_i^* chosen by A_1 in the list L^{pk}.

(x) *Sign queries:* When A_1 performs the query with the user identity ID_i and public key pk_i, message m_i, C accesses L^{H_1}, L^{H_2}, L^{h_1}, and L^{h_2} to get ε_i, Q_i, U, α_i, and β_i, respectively. Furthermore, C randomly selects r_i and computes $R_i = r_i P$; if $\varepsilon_i = 0$, C computes $V_i = \delta_i \alpha_i MS_{pub} + r_i MS_{pub} + \beta_i \vartheta pk_i$; otherwise, if $\varepsilon_i = 1$, C computes $V_i = \delta_i \alpha_i bMS_{pub} + r_i MS_{pub} + \beta_i \vartheta pk_i$. It returns $\sigma_i = (R_i, V_i)$ to A_1 as the signature of the message m_i on the user identity ID_i with the public key pk_i.

(xi) *Forgery:* Finally, A_1 outputs a forged aggregate signature $\sigma^* = (R^*, V^*)$ from message-identity-public key pairs (m_i^*, ID_i^*, pk_i^*), where $1 \le i \le n$. If all $\varepsilon_i = 0$ hold, A_1 aborts; otherwise, without loss of generality, let $ID_{ts} = ID_1$; that is, $\varepsilon_1 = 1$, $\varepsilon_i = 0$ $(2 \le i \le n)$, and then the forged signature should make the following hold:

$$e(V^*, P) = e\left(\sum_{i=1}^{n} \left(\alpha_i^* Q_i^* + R_i^*\right), MS_{pub}\right) e\left(\sum_{i=1}^{n} \beta_i^* pk_i^*, U\right) \tag{10}$$

where $Q_i^* = \delta_i^* P$ $(2 \le i \le n)$, $Q_1^* = \delta_1^* bP$, $U = \vartheta P$, $V^* = \sum_{i=1}^{n} V_i^*$, and $R^* = \{R_1^*, R_2^*, \ldots, R_n^*\}$.

Furthermore, the derivation process is shown as follows:

$$\begin{aligned}
&e(V^*, P) = e\left(\sum_{i=1}^{n} \left(\alpha_i^* Q_i^* + R_i^*\right), MS_{pub}\right) \cdot e\left(\sum_{i=1}^{n} \beta_i^* pk_i^*, U\right) \\
&\Longrightarrow e(V^*, P) = e\left(R_1^* + \sum_{i=2}^{n} \left(\alpha_i^* Q_i^* + R_i^*\right), MS_{pub}\right) \cdot e\left(\alpha_1^* Q_1^*, MS_{pub}\right) e\left(\sum_{i=1}^{n} \beta_i^* pk_i^*, \vartheta P\right) \\
&\Longrightarrow e\left(\alpha_1^* Q_1^*, MS_{pub}\right) = e(V^*, P) \cdot \left[e\left(R_1^* + \sum_{i=2}^{n} \left(\alpha_i^* Q_i^* + R_i^*\right), MS_{pub}\right) \cdot e\left(\sum_{i=1}^{n} \beta_i^* pk_i^*, \vartheta P\right)\right]^{-1} \\
&\Longrightarrow e\left(\delta_1^* \alpha_1^* abP, P\right) = e(V^*, P) \cdot \left[e\left(R_1^* + \sum_{i=2}^{n} \left(\alpha_i^* Q_i^* + R_i^*\right), MS_{pub}\right) \cdot e\left(\sum_{i=1}^{n} \beta_i^* pk_i^*, \vartheta P\right)\right]^{-1} \\
&\Longrightarrow \delta_1^* \alpha_1^* abP = V^* - \left[\left(r_1^* + \sum_{i=2}^{n} \left(\delta_i^* + r_i^*\right)\right) MS_{pub} + \sum_{i=1}^{n} \beta_i^* x_i^* \vartheta\right] \\
&\Longrightarrow abP = \left(V^* - \left(r_1^* + \sum_{i=2}^{n} \left(\delta_i^* + r_i^*\right)\right) MS_{pub} - \sum_{i=1}^{n} \beta_i^* x_i^* \vartheta\right) \left(\delta_1^* \alpha_1^*\right)^{-1}
\end{aligned} \tag{11}$$

However, this contradicts the CDH assumption; thus the single signature and aggregate signature generated by the new scheme are unforgeable.

Theorem 4. *The proposed certificateless aggregate scheme is existentially unforgeable against type 2 adversary A_2, if the CDH problem is difficult to solve in G_1.*

Proof. We can prove the unforgeability of our CL-AS scheme against type 2 adversary with **Game2** that involves A_2 and an algorithm called simulator C. □

Given a random instance of the CDH problem $(P, Q_1 = aP, Q_2 = bP)$, where P is a generator of G_1, our ultimate goal is to find the result of abP by solving the CDH problem.

(i) *Setup:* C randomly chooses ID_{ts} as the target identity of sensor challenged, sets $MS_{pub} = \lambda P$, and returns master key λ and system parameter $params = \{G_1, G_2, P, e, q, MS_{pub}, H_1, H_2, h_1, h_2\}$ to A_2. A_2 performs the inquiries as follows.

(ii) h_1, h_2, and *Reveal-Secret-Value queries* are the same as the corresponding queries in Theorem 3. Since A_2 can access the master key, there is no need to the *Reveal-Partial-Private-Key queries* and *Replace-Public-Key queries*.

(iii) H_1 *query:* C maintains a list denoted L^{H_1}, and the structure of L^{H_1} is (ID_i, δ_i, Q_i), all the elements in L^{H_1} are initialized to null. When A_2 performs the query with the identity ID_i, C checks whether a tuple L^{H_1} is (ID_i, δ_i, Q_i) exists in L^{H_1}; if it exists, it returns Q_i to A_2; otherwise, C randomly selects δ_i and computes $Q_i = \delta_i P$. It returns Q_i to A_2 and stores (ID_i, δ_i, Q_i) to L^{H_1}.

(iv) H_2 *query:* C maintains a list denoted L^{H_2}, and the structure of L^{H_2} is (MS_{pub}, ϑ, Z), all the elements in L^{H_2} are initialized to null. When A_2 executes the query with MS_{pub}, C checks whether a tuple (MS_{pub}, ϑ, Z) exists in L^{H_2}; if it exists, it returns U to A_2; otherwise, C randomly selects $\vartheta \in Z_q^*$ and computes $U = \vartheta Q_1 = \vartheta aP$. It returns U to A_2 and stores (MS_{pub}, ϑ, U) to L^{H_2}.

(v) *Reveal-Public-Key queries:* C maintains a list denoted L^{pk}, and the structure of L^{pk} is (ID_i, ω_i, pk_i), all the elements in L^{pk} are initialized to null. When A_2 performs the query with the identity ID_i, C checks whether a tuple (ID_i, ω_i, pk_i) exists in L^{pk}; if it exists pk_i, it returns pk_i to A_2; otherwise, C randomly selects $\omega_i \in \{0, 1\}$; if $\omega_i = 0$, C accesses L^x to get x_i and computes $pk_i = x_i P$; otherwise, if $\omega_i = 1$, C randomly selects $x_i \in Z_q^*$ and computes $pk_i = x_i Q_2 = x_i bP$. It returns pk_i to A_2 and stores (ID_i, ω_i, pk_i) to L^{pk}.

(vi) *Sign queries:* When A_2 performs the query with user's identity ID_i and public key pk_i, message m_i, C accesses $L^{H_1}, L^{H_2}, L^{h_1}, L^{h_2}$, and L^{pk} to get $Q_i, U, \alpha_i, \beta_i$, and ω_i, respectively. Furthermore, C randomly selects r_i and computes $R_i = r_i P$; if $\omega_i = 0$, C computes $V_i = \delta_i \alpha_i MS_{pub} + r_i MS_{pub} + \beta_i \vartheta pk_i$; otherwise, if $\omega_i = 1$, C computes $V_i = \delta_i \alpha_i MS_{pub} + r_i MS_{pub} + \beta_i \vartheta apk_i$. It returns $\sigma_i = (R_i, V_i)$ to A_2 as the signature of the message m_i on user's identity ID_i with the public key pk_i.

(vii) *Forgery:* Finally, A_2 outputs a forged aggregate signature $\sigma^* = (R^*, V^*)$ from message-identity-public key pairs (m_i^*, ID_i^*, pk_i^*), where $1 \le i \le n$. If all $\omega_i = 0$ hold, A_2 aborts; otherwise, without loss of generality, let $ID_{ts} = ID_1$; that is, $\omega_1 = 1$, $\omega_i = 0$ $(2 \le i \le n)$, and then the forged aggregate signature should satisfy:

$$e(V^*, P) = e\left(\sum_{i=1}^{n} \alpha_i^* Q_i^* + R_i^*\right), MS_{pub}) e\left(\sum_{i=1}^{n} \beta_i^* pk_i^*, U\right) \tag{12}$$

where $Q_i^* = \delta_i^* P$, $pk_1^* = x_1^* bP$, $pk_i^* = x_i^* P$ $(2 \le i \le n)$, $U = \vartheta aP$, $V^* = \sum_{i=1}^{n} V_i^*$, and $R^* = \{R_1^*, R_2^*, \ldots, R_n^*\}$.

Furthermore, the derivation process is shown as below:

$$\begin{aligned}
&e(V^*, P) = e\left(\sum_{i=1}^{n} (\alpha_i^* Q_i^* + R_i^*), MS_{pub}\right) \\
&\quad \cdot e\left(\sum_{i=1}^{n} \beta_i^* pk_i^*, U\right) \\
&\Longrightarrow e(V^*, P) = e\left(\sum_{i=1}^{n} (\alpha_i^* Q_i^* + R_i^*), MS_{pub}\right) \\
&\quad \cdot e(\alpha_1^* pk_1^*, U)\, e\left(\sum_{i=2}^{n} \beta_i^* pk_i^*, U\right) \\
&\Longrightarrow e(\beta_1^* pk_1^*, U) = e(V^*, P) \\
&\quad \cdot \left[e\left(\sum_{i=1}^{n} (\alpha_i^* Q_i^* + R_i^*), MS_{pub}\right)\right. \\
&\quad \left. \cdot e\left(\sum_{i=2}^{n} \beta_i^* pk_i^*, U\right)\right]^{-1} \\
&\Longrightarrow e(\beta_1^* \vartheta abP, P) = e(V^*, P) \\
&\quad \cdot \left[e\left(\sum_{i=1}^{n} (\alpha_i^* Q_i^* + R_i^*), MS_{pub}\right)\right. \\
&\quad \left. \cdot e\left(\sum_{i=2}^{n} \beta_i^* pk_i^*, U\right)\right]^{-1} \\
&\Longrightarrow \beta_1^* \vartheta abP = V^* - \left[\left(\sum_{i=1}^{n} (\delta_i^* \alpha_i^* + r_i^*)\right) MS_{pub}\right. \\
&\quad \left. + \sum_{i=1}^{n} \beta_i^* x_i^* \vartheta\right]
\end{aligned} \tag{13}$$

Table 1: Security comparisons.

	B_{11}	B_{12}	B_{13}	SP	B_{21}	B_{22}	B_{23}	SP
Gong's Scheme [9]	*No*	*Yes*	*No*	*L*	*Yes*	*No*	*No*	*L*
liu's Scheme [10]	*Yes*	*Yes*	*Yes*	*H*	*No*	*No*	*No*	*L*
kumar's Scheme [8]	*Yes*	*Yes*	*Yes*	*H*	*No*	*No*	*No*	*L*
Our proposed Scheme	*Yes*	*Yes*	*Yes*	*H*	*Yes*	*Yes*	*Yes*	*H*

Table 2: symbol-operation-execution time.

Symbol	Operation	Time (ms)
T_{mts}	The time of performing a general hash operation in Z_q^*	0.0002
T_{mtp}	The time of performing a map-to-point operation in G_1	9.773
T_{ecc-pa}	The time of performing a point addition operation in G_1	0.022
T_{ecc-pm}	The time of performing a point multiplication operation in G_1	3.740
T_{bp}	The time of performing a bilinear pairing operation	11.515

$$\Longrightarrow abP = \left(V^* - \left(\sum_{i=1}^{n}\left(\delta_i^*\alpha_i^* + r_i^*\right)\right)MS_{pub} - \sum_{i=1}^{n}\beta_i^* x_i^* \vartheta\right)\left(\beta_1^*\vartheta\right)^{-1} \tag{14}$$

However, this contradicts the CDH assumption; thus the single signature and aggregate signature generated by the new scheme are unforgeable.

8. Security Comparisons and Performance Analysis

In this section, we first compare the security of the newly proposed CL-AS scheme with the other three CL-AS schemes and further analyze the performance of the new CL-AS scheme by evaluating the computation overhead.

8.1. Security Comparisons. In this subsection, we compare the security of the newly proposed CL-AS scheme with the other three CL-AS schemes [8–10]. For the convenience of description, let A_1 and A_2 denote the type1 and the type2 adversaries, respectively. Furthermore, the two types of adversaries are divided into three levels [31], where B_{i1} denotes general adversary, B_{i2} denotes strong adversary, B_{i3} denotes super adversary, respectively, and $i \in \{1, 2\}$; the value of i corresponds to the type i adversary. *Yes* denotes that it can satisfy the corresponding security requirement and *No* denotes that it cannot satisfy the corresponding security requirement. L denotes the weaker security and H denotes the stronger security under the corresponding attack types. SP denotes the security performance. The security comparisons of the various schemes are listed in Table 1.

As shown in Table 1, we can find that the first three schemes (i.e., Gong's scheme [9], liu's scheme [10], and kumar's scheme [8]) cannot satisfy all the security requirements. Especially for Gong's two CL-AS schemes [9], under the attacks of the type1 and the type2 adversaries, none of them can meet the security levels of B_3. liu and kumar's schemes cannot resist the malicious KGC attack (B_3 level). In contrast, our CL-AS scheme can meet all the security requirements. Hence, our proposed CL-AS scheme has better security than that of the other three schemes.

8.2. Performance Analysis. In this section, we analyze the performance of our CL-AS scheme by evaluating the computation overhead. Compared with that of kumar *et al.*'s scheme, our implementation shows that the new proposal can satisfy the security requirement and provide an improved security while reducing the computation cost.

In order to achieve a credible security level, we choose q and p as 160-bits prime number and 512-bits prime number, respectively. A ate pairing $e : G_1 \times G_1 \rightarrow G_2$ is used in our experiments, where G_1 and G_2 are cyclic groups with the same order q, defined on the super singular elliptic curve $E(F_p) : y^2 = x^3 + 1$.

We have implemented kumar *et al.*'s scheme and the newly proposed scheme with the MIRACL library [32] on a personal computer (Lenovo with Intel I5-3470 3.20G Hz processor, 4G bytes memory and Window 7 operating system). For the sake of simplicity, we firstly define the corresponding relations related symbol-operation-execution time as shown in Table 2.

Because *Setup*, *Partial-Private-Key-Gen*, and *Private-Key-Gen* phases are executed by MS or user and all of them are one-time operation, we laid stress on the comparisons of the computation cost in *Sign*, *Verify*, *Aggregate*, and *Aggregate-Verify* phases.

In *Sign* phase, the user in kumar *et al.*'s scheme needs to perform one general hash operation in Z_q^*, one map-to-point hash operation in G_1, two-point addition operations in G_1 and three-point multiplication operations in G_1. Therefore, the running time of the *Sign* phase is $T_{mts} + T_{mtp} + 2T_{ecc-pa} + 3T_{ecc-pm}$, whereas the user in the new proposal needs to perform two general hash operations in Z_q^*, one map-to-point hash operation in G_1, two-point addition operations in G_1, and four-point multiplication operations in G_1. Therefore,

Table 3: Computation cost comparisons (millisecond).

	kumar's Scheme [8]	**Our Proposed Scheme**
Sign	$T_{mts} + T_{mtp} + 2T_{ecc-pa} + 3T_{ecc-pm} \approx 21.0372$	$2T_{mts} + T_{mtp} + 2T_{ecc-pa} + 4T_{ecc-pm} \approx 24.7774$
Verify	$T_{mts} + T_{mtp} + T_{ecc-pa} + T_{ecc-pm} + 3T_{bp} \approx 48.0802$	$2T_{mts} + T_{mtp} + T_{ecc-pa} + 2T_{ecc-pm} + 3T_{bp} \approx 51.8204$
Agrregate	$99T_{ecc-pa} \approx 2.178$	$99T_{ecc-pa} \approx 2.178$
Aggregate–Verify	$100T_{mts} + 200T_{mtp} + 298T_{ecc-pa} + 100T_{ecc-pm} + 3T_{bp} \approx 2369.721$	$200T_{mts} + 101T_{mtp} + 298T_{ecc-pa} + 200T_{ecc-pm} + 3T_{bp} \approx 1776.214$

the running time of the *Sign* phase in our proposed scheme is $2T_{mts} + T_{mtp} + 2T_{ecc-pa} + 4T_{ecc-pm}$ milliseconds.

In *Verify* phase, the verifier in kumar *et al.*'s scheme needs to perform one general hash operation in Z_q^*, one map-to-point hash operation in G_1, one-point addition operation in G_1, one-point multiplication operation in G_1, and three-bilinear pairing operations. Therefore, the running time of the *Verify* phase is $T_{mts} + T_{mtp} + T_{ecc-pa} + T_{ecc-pm} + 3T_{bp}$, whereas the verifier in the new proposal needs to perform two general hash operations in Z_q^*, one map-to-point hash operation in G_1, one-point addition operation in G_1, two-point multiplication operation in G_1, and three-bilinear pairing operations. Therefore, the running time of the *Verify* phase in our proposed scheme is $2T_{mts} + T_{mtp} + T_{ecc-pa} + 2T_{ecc-pm} + 3T_{bp}$ milliseconds.

In *Aggregate* phase, the aggregator in kumar *et al.*'s scheme needs to perform $n - 1$ point addition operations in G_1, whereas the aggregator in the new proposal needs to perform $n - 1$ point addition operations in G_1. We can find that the running time of the *Aggregate* phase in the two schemes is equal to $(n-1)T_{ecc-pa}$ milliseconds.

In *Aggregate – Verify* phase, the aggregate verifier in kumar *et al.*'s scheme needs to perform n general hash operations in Z_q^*, $2n$ map-to-point hash operations in G_1, $3n-2$ point addition operations in G_1, n point multiplication operations in G_1, and three-bilinear pairing operations. Therefore, the running time of the *Aggregate – Verify* phase is $nT_{mts} + 2nT_{mtp} + (3n-2)T_{ecc-pa} + nT_{ecc-pm} + 3T_{bp}$ milliseconds, whereas the verifier in the new proposal needs to perform $2n$ general hash operations in Z_q^*, $n+1$ map-to-point hash operations in G_1, $3n-2$ point addition operations in G_1, $2n$ point multiplication operations in G_1, and three-bilinear pairing operations. Therefore, the running time of the *Aggregate – Verify* phase in our proposed scheme is $2nT_{mts} + (n+1)T_{mtp} + (3n-2)T_{ecc-pa} + 2nT_{ecc-pm} + 3T_{bp}$ milliseconds.

Assuming that $n = 100$ in the *Aggregate* and *Aggregate–Verify* phases, the computation overhead comparisons are shown in Table 3 and Figure 3. As can be seen from the results in Table 3 and Figure 3, the computation overhead of our proposed CL-AS scheme is slightly higher than that of kumar *et al.*'s scheme for *Sign* and *Verify* phases. In *Aggregate* phase, the computation overheads of the two schemes are equal, whereas in the *Agrregate – Verify* phase, the computation overhead of our scheme is much lower than that of kumar *et al.*'s scheme. However, compared with the total computation overheads of these four phases, our scheme's computation overhead is reduced by 24 percentage points compared with the that of kumar *et al.*'s scheme [8]. That is, we enforce the security in a large extent with the efficiency increased by 24 in computation overhead.

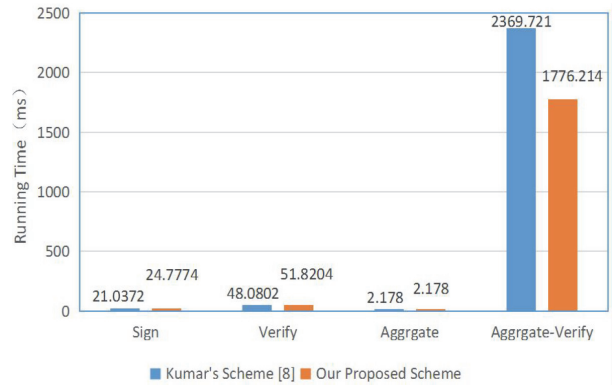


Figure 3: Computation cost comparisons.

9. Conclusion

To ensure the privacy and integrity of patients medical information, several CL-AS schemes have been put forward recently. In this paper, we first investigate the techniques of the data signature. Then we show that Pankaj Kumar *et al.*'s scheme is vulnerable against the malicious attack. This attack is a serious threat from the inside attacker acting as a MS, because it allows the adversary to forge a signature of message m_j using the signature of the message m_i on signer ID_i.

To overcome this security flaw, we put forward a new CL-AS scheme for the issues of integrity and privacy in HMSN. The security analysis shows that our proposed CL-AS scheme is provably secure and can meet the security requirements in HMSN. In addition, the detailed performance analysis and evaluation demonstrate that our CL-AS scheme can achieve a novel security level while reducing the computation cost. Our CL-AS scheme is robust against all types of attacks, making it more useful for protecting the integrity and privacy of patients medical information.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] D. J. Cook, J. C. Augusto, and V. R. Jakkula, "Ambient intelligence: Technologies, applications, and opportunities," *Pervasive and Mobile Computing*, vol. 5, no. 4, pp. 277–298, 2009.

[2] Z. Zhang and K. Wang, "A trust model for multimedia social networks," *Social Network Analysis and Mining*, vol. 3, no. 4, pp. 969–979, 2013.

[3] M. Al Ameen, J. Liu, and K. Kwak, "Security and privacy issues in wireless sensor networks for healthcare applications," *Journal of Medical Systems*, vol. 36, no. 1, pp. 93–101, 2012.

[4] M. R. Yuce, S. W. P. Ng, N. L. Myo, J. Y. Khan, and W. Liu, "Wireless body sensor network using medical implant band," *Journal of Medical Systems*, vol. 31, no. 6, pp. 467–474, 2007.

[5] C. Gao, Q. Cheng, X. Li, and S. Xia, "Cloud-assisted privacy-preserving profile-matching scheme under multiple keys in mobile social network," *Cluster Computing*, pp. 1–9, 2018.

[6] Z. Huang, S. Liu, X. Mao, K. Chen, and J. Li, "Insight of the protection for data security under selective opening attacks," *Information Sciences*, vol. 412-413, pp. 223–241, 2017.

[7] P. Li, J. Li, Z. Huang, C.-Z. Gao, W.-B. Chen, and K. Chen, "Privacy-preserving outsourced classification in cloud computing," *Cluster Computing*, pp. 1–10, 2017.

[8] P. Kumar, S. Kumari, V. Sharma, A. K. Sangaiah, J. Wei, and X. Li, "A certificateless aggregate signature scheme for healthcare wireless sensor network," *Sustainable Computing*, 2017.

[9] G. Zheng, L. Yu, H. Xuan, and C. Kefei, "Two certificateless aggregate signatures from bilinear maps," in *Proceedings of the SNPD 2007: 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, pp. 188–193, chn, August 2007.

[10] H. Liu, S. Wang, M. Liang, and Y. Chen, "New construction of efficient certificateless aggregate signatures," *International Journal of Security and Its Applications*, vol. 8, no. 1, pp. 411–422, 2014.

[11] W. Diffie, W. Diffie, and M. E. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[12] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in cryptology (Santa Barbara, Calif., 1984)*, vol. 196 of *Lecture Notes in Comput. Sci.*, pp. 47–53, Springer, Berlin, 1985.

[13] J. Li, J. Li, X. Chen, C. Jia, and W. Lou, "Identity-based encryption with outsourced revocation in cloud computing," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 64, no. 2, pp. 425–437, 2015.

[14] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," *Asiacrypt*, vol. 2894, pp. 452–473, 2003.

[15] X. Huang, W. Susilo, Y. Mu, and F. Zhang, "On the security of certificateless signature schemes from Asiacrypt 2003," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Preface*, vol. 3810, pp. 13–25, 2005.

[16] J. Li, X. Huang, Y. Mu, and W. Wu, "Cryptanalysis and improvement of an efficient certificateless signature scheme," *Journal of Communications and Networks*, vol. 10, no. 1, pp. 10–17, 2008.

[17] W. Yap, S. Heng, and B. Goi, "An Efficient Certificateless Signature Scheme," in *Emerging Directions in Embedded and Ubiquitous Computing*, vol. 4097 of *Lecture Notes in Computer Science*, pp. 322–331, 2006.

[18] M. H. Au, J. Chen, J. K. Liu, Y. Mu, D. S. Wong, and G. Yang, "Malicious KGC attacks in certificateless cryptography," in *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security, ASIACCS '07*, pp. 302–311, March 2007.

[19] A. W. Dent, B. t. Libert, and K. . Paterson, "Certificateless encryption schemes strongly secure in the standard model," in *Public key cryptography*, vol. 4939 of *Lecture Notes in Comput. Sci.*, pp. 344–359, Springer, Berlin, 2008.

[20] X. Li, K. Chen, and L. Sun, "Certificateless signature and proxy signature schemes from bilinear pairings," *Lithuanian Mathematical Journal*, vol. 45, no. 1, pp. 95–103, 2005.

[21] J. K. Liu, M. H. Au, and W. Susilo, "Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model," in *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security (ASIACCS '07)*, pp. 273–283, March 2007.

[22] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Lecture Notes in Computer Science*, vol. 2656 of *Lecture Notes in Comput. Sci.*, pp. 416–432, Springer, Berlin, 2003.

[23] L. Zhang and F. Zhang, "A new certificateless aggregate signature scheme," *Computer Communications*, vol. 32, no. 6, pp. 1079–1085, 2009.

[24] H. Xiong, Z. Guan, Z. Chen, and F. Li, "An efficient certificateless aggregate signature with constant pairing computations," *Information Sciences*, vol. 219, pp. 225–235, 2013.

[25] D. He, M. Tian, and J. Chen, "Insecurity of an efficient certificateless aggregate signature with constant pairing computations," *Information Sciences*, vol. 268, pp. 458–462, 2014.

[26] L. Cheng, Q. Wen, Z. Jin, H. Zhang, and L. Zhou, "Cryptanalysis and improvement of a certificateless aggregate signature scheme," *Information Sciences*, vol. 295, pp. 337–346, 2015.

[27] F. Zhang, L. Shen, and G. Wu, "Notes on the security of certificateless aggregate signature schemes," *Information Sciences*, vol. 287, pp. 32–37, 2014.

[28] Y. Zhang and C. Wang, "Comment on new construction of efficient certificateless aggregate signatures," *International Journal of Security and Its Applications*, vol. 9, no. 1, pp. 147–154, 2015.

[29] D. He and S. Zeadally, "Authentication protocol for an ambient assisted living system," *IEEE Communications Magazine*, vol. 53, no. 1, pp. 71–77, 2015.

[30] D. He, S. Zeadally, and L. Wu, "Certificateless public auditing scheme for cloud-assisted wireless body area networks," *IEEE Systems Journal*, no. 99, pp. 1–10, 2015.

[31] D. He, S. Zeadally, B. Xu, and X. Huang, "An Efficient Identity-Based Conditional Privacy-Preserving Authentication Scheme for Vehicular Ad Hoc Networks," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2681–2691, 2015.

[32] M. Scott, *Miracla multiprecision integer and rational arithmetic c/c++ library. shamus software ltd*, Dublin, Ireland, 2003, http://indigo.ie/mscott.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 4351603, 17 pages
https://doi.org/10.1155/2018/4351603

Review Article

Survey of Authentication and Authorization for the Internet of Things

Michal Trnka,[1] **Tomas Cerny**,[2] **and Nathaniel Stickney**[2]

[1]*Department of Computer Science, FEE, Czech Technical University in Prague, Prague, Czech Republic*
[2]*Department of Computer Science, Baylor University, Waco, TX, USA*

Correspondence should be addressed to Michal Trnka; trnkami1@fel.cvut.cz

Received 30 January 2018; Revised 20 April 2018; Accepted 10 May 2018; Published 12 June 2018

Academic Editor: Ilsun You

The Internet of Things is currently getting significant interest from the scientific community. Academia and industry are both focused on moving ahead in attempts to enhance usability, maintainability, and security through standardization and development of best practices. We focus on security because of its impact as one of the most limiting factors to wider Internet of Things adoption. Numerous research areas exist in the security domain, ranging from cryptography to network security to identity management. This paper provides a survey of existing research applicable to the Internet of Things environment at the application layer in the areas of identity management, authentication, and authorization. We survey and analyze more than 200 articles, categorize them, and present current trends in the Internet of Things security domain.

1. Introduction

Computer networks trace back to the early 1960s [1, 2]. TCP/IP, perhaps the most widely known protocol, was initially proposed in 1974 [3] and widely adopted in the early 1980s, allowing the widespread adoption of the Internet and its commercial use in the late 1980s [4]. In the beginning, the Internet connected only computers together, but in the 2000s, mobile devices began connecting to the Internet [5], and today network connection is available in more and more devices, which are called "smart objects" [6]. The current state is known as the "Internet of Things" (IoT) [7]. The forecast for the year 2020 is that the IoT will connect 20.5 billion devices with over three trillion US dollars spent on the hardware alone [8].

Security and privacy are considered the most crucial IoT challenges [7, 9]. Gartner [8] states, "security and risk concerns will continue to be the greatest impediment to IoT adoption. The market for IoT-specific security solutions will dramatically expand in 2017 as existing security providers aggressively retool existing capabilities to address IoT security risks."

This paper presents an overview of existing research in the areas of authentication, authorization, and identity management accomplished since 2013. The main focus areas in this work are security at the application layer, device management, and access rule enforcement. It excludes network and communications security. Candidate papers are identified not only by manual survey, but also by a systematic search [10] through multiple research indexing sites and portals. The resulting papers are analyzed to provide a survey and classification of existing work.

Other surveys of IoT security research exist. Sicari et al. [11] present existing solutions in seven categories: authentication, confidentiality, access control, privacy, trust, secure middleware, mobile security, and policy enforcement. Their survey summarizes state of the art in 2014 and does not include the most recent findings. Roman et al. [12] focus on security issues in a comparison of centralized and distributed architectures for IoT systems, both listing issues of the architectures, and outlining promising solutions to those issues as of 2013. Yang et al. [13] describe authentication and access control at the application layer, and security at the perception, network, and transport layers. We focus specifically on authentication and authorization at the application layer. Our survey employs a systematic search within the most prominent indexers. This way, our results are repeatable and not subjectively biased.

This paper provides an overview of basic authentication, authorization, and identity management techniques in Section 2. We define research goals and research questions to be answered in Section 3, and in Section 4 we describe our methods for paper discovery and selection, as well as the number of papers found. We categorize the research in Section 5 and further describe our categorization findings in Sections 5-8. In Section 10 we discuss threats to the validity of our survey.

2. Background

Initially computers were used as advanced machines to process various calculations or other processes without storing input or output data. While the systems supported multiple users, no data were stored, so security issues were not prevalent. However, when computers began to be used for data management and storage with multiple users accessing the system, the problem of access control emerged.

From the 1970s on, two predominant access control models were used: Mandatory Access Control (MAC) and Discretionary Access Control (DAC) [57]. MAC is predominantly used in applications with strict, centralized access control. Access rules are set by administrators and enforced by the system; users are not allowed to set or modify access policies for system resources. DAC is the opposite; no central element is needed and each user determines for the access policy for resources which they own.

As the complexity of applications increased, and they evolved into complex information systems with hundreds or thousands of users, a conceptual framework for easier access management was needed. Role-Based Access Control (RBAC) [58] allows grouping users together into groups, known as roles; each user may be assigned multiple roles. Access rules are further defined for the roles, and not single users. Roles often follow the organizational structure of the institution using the information system and are therefore easy to understand for business owners of the application. RBAC was introduced in the early 1990s and quickly became the predominant access control model.

As application user base sizes have continued to grow, the limitations of RBAC have become more apparent, including its unsuitability for context-aware applications [59] or for applications at a scale where the number of roles or role sets needed to cover different access right combinations is too extensive for manual management. Researchers have moved in two directions to address these issues. One direction is to extend the RBAC model in creative and numerous ways [60–65]. The other is to develop a more general access control model. Specifically, there is a growing interest in Attribute-Based Access Control (ABAC) [66]. It bases access rules on the user's attributes, rather than on predefined roles. ABAC can preserve all of the benefits of MAC, DAC, and RBAC while adding more flexibility; it can be used to support, or be implemented under, any of these access control paradigms.

The access control methods described above deal predominantly with authorizing users to access specific resources or take specific actions, rather than describing how the user should be authenticated; authentication is considered a prerequisite for authorization. This authentication may be accomplished using three basic credential categories. The first category, "Something I am", represents properties about the user, including their location or biometric characteristics. "Something I have" stands for credentials that were given to a user; the user possesses the credential. This category includes all types of keys, tokens, cards, or even personal devices like phones. The last and most familiar category is "Something I know", most often represented by passwords, but not limited to them; it also includes the user's knowledge of security questions, their interaction history, and other information.

Authorization credential categories may be combined together for increased security or to improve the user experience. Multifactor authentication is a common practice to increase security and prevent a breach in the event that a single credential is compromised. Some authentication frameworks provide single sign-on functionality where users sign into a trusted authentication provider using their credentials (often just a password) and receive provisional tokens which are then used to authenticate against other services. Subsequently, those services verify the token with authentication provider and log the user in. Often, the usage of the token is automated, and the user only needs to log in once.

Identity management is closely related to authentication and authorization. A virtual identity must exist against which users may authenticate and which stores user attributes (unique identification, attributes as understood in ABAC, RBAC roles, and other required information) used for authorization.

At the most basic level, applications each manage identity independently, using as little information as possible; generally this includes both a principal (identity unique identifier) and credentials used for authentication. As applications become more complex, the information required for user authorization grew to include roles or identity attributes. As the number of applications per user and the number of users per service increase, it becomes difficult both for the user and service administrators to manage the growing amount of identity information required. These developments led to the need for federated identity management—a way of providing identity services for multiple applications, often tied to authentication mechanisms. Currently, several implementations of federated identity management exist, including using LDAP [67] for identity management or using OpenID [68] as an identity service.

3. Goals

The motivation of this survey is to provide an overview of current progress on research in the domain of IoT security. This is a broad discipline and therefore we focus particularly on authorization, authentication, and identity management papers, specifically at the highest layer of the network stack, typically the application layer. While "network stack" is not the precise model used for the IoT, we use the term in lieu of a more standard vocabulary to describe the IoT technology and communication architecture; there does not yet appear to be common agreement on such a term. We are interested in architectures, projects, solutions, proposals,

and frameworks dealing with user-to-machine and machine-to-machine authentication and authorization. We are also interested in identity management for IoT devices.

In this paper, we raise the following specific research questions:

RQ1 What is the taxonomy of security solutions?

RQ2 How does context-awareness extend security?

RQ3 Are existing approaches and standards adapted and extended for IoT security, or are novel methods proposed?

RQ4 Which approaches are applicable to distributed or centralized architectures?

RQ5 Does existing research focus on user-to-machine or machine-to-machine interactions?

The questions above are examined further in their respective sections. Each section provides a list of the research found, a summary of its content, and an answer to the particular question.

4. Search

In order to systematically review existing research and answer our research questions, we performed searches at the following indexing sites and portals: IEEE Xplore, ACM Digital Library (ACM DL), Web of Science (WoS), SpringerLink, and ScienceDirect.

To show that our search queries provide results relevant for this survey, we evaluated our search query results against a control set of papers identified as matching our scope through manual search before we performed the search queries. When a search query returned papers from the control set, this is evidence of the usefulness of the search query.

The search query consists of two parts. The first part targets terms and keywords to be included in the paper and the second part removes papers that contain terms we are not interested in. Naturally, we are interested in research about the IoT so we include "Internet of Things" or "IoT" as one of the main groups. Another crucial term is "Security" as we target only those papers that deal with security. Further restriction terms refine the results to include only papers with "Authentication", "Authorization", "Access Control", or identity management, which is shortened to "Identity". The second portion of the query is to limit the amount of the articles in the result set. We removed papers that deal with the security at the lower levels of the network stack. This translates to the terms "Network", "Hardware", "RFID", and "protocol". Cryptography is not a particular focus of this survey, so we also remove research with this keyword. Finally, we remove papers that are surveys themselves, containing "Survey" or "Study" in their title.

The query syntax differs for each indexing site, but we aim to search through abstracts or keywords/topics where applicable. The queries are constructed as similarly as possible. The exact queries used, including the general query we used as a template, are listed in Table 1.

We encountered an issue with the search function in SpringerLink. The search system is not able to process a refined query such as the one we designed. We used a simpler query that returned 383 papers and processed these results by constructing a short script that opens the particular page for every exported paper, extracts the abstract, and performs the refined query locally on our machine.

Running the query across all five indexing services gives us a set of 387 papers, from which we exclude those that have less than 4 pages or are from year 2012 or earlier. Since WoS indexes papers that appear at other sites, it contains 16 duplicate papers, which we also remove. As a final filter, we read the abstract of each article and removed those papers not within the designed scope; this gives us 86 prefiltered candidate papers.

These remaining papers we read one by one, with some exceptions. The full text of one paper could not be downloaded; this was removed from the results set. Three of the papers were highly-similar extensions of another paper in the results set. In this case, we used the extended paper and discarded the shorter versions. We also removed papers that did not fit into the scope of this survey—those where the abstract initially indicated connection to our research questions but the full text did not. The complete statistics of papers found, prefiltered, and included for every indexing site can be seen in Table 2.

5. Taxonomy

To find candidate categories based on the most prevalent keywords we employ the RAKE [69] algorithm for keyword extraction. First, we transform the PDF documents using *pdftotxt* (5) and strip references or appendices. Then, we apply the RAKE algorithm with the following parameters for the keyword extraction: at least five characters, a maximum of two words for the keyword, and at least four occurrences in the text. For each keyword, we then find matching articles. Only keywords present in at least two papers are taken into consideration. We then group synonymous keywords into categories. As a consequence of this approach a paper may fall into multiple categories.

The results (excluding general terms) suggest the following categories of the papers. They are also illustrated in Figure 1.

(i) **authentication:** papers that address authentication [14–35]

(ii) **authorization:** articles dealing with authorization [18, 20, 21, 23, 26, 28, 30, 32–34, 36–45]

(iii) **service:** solutions that can be used in both IoT and SOA [15–17, 21–23, 25, 27, 30, 32, 35, 36, 46–49]

(iv) **token:** articles that use any form of token as an information bearer in their proposal [19, 21, 23, 29, 35–37, 40, 41, 43, 46, 50]

(v) **context:** papers using or proposing context-aware methods [14, 23, 33, 35, 36, 38, 46, 51–53]

(vi) **cloud:** research addressing security issues of cloud-based IoT devices [14, 16, 20, 41, 45, 51, 53, 54]

(vii) **identity management:** solutions discussing identity management [15, 18, 22, 34, 35, 46, 47, 50]

Table 1: Queries used for the search.

Indexer	Query
General query	(”Internet of Things” OR ”IoT”) AND ”Security” AND (”Authentication” OR ”Authorization” OR ”Identity” OR ”Access control”) AND NOT (”Network” OR ”Hardware” OR ”RFID” OR ”Protocol” OR ”Cryptography” OR ”Survey” OR ”Study”)
IEEE Xplore	((”Abstract”: ”Internet of Things” OR ”Abstract”: ”IoT”) AND (”Abstract”: ”Authentication” OR ”Abstract”: ”Authorization” OR ”Abstract”: ”Identity” OR ”Abstract”:”Access Control”) AND ”Index Terms”: ”Security” AND NOT(Search_Index_Terms: ”Network” OR ”Abstract”: ”Hardware” OR ”Abstract”: ”Cryptography” OR ”Abstract”: ”Protocol” OR ”Document Title”: ”Survey” OR ”Abstract”: ”RFID” OR ”Document Title”:”Study”))
ACM DL	recordAbstract:(IoT ”Internet Of Things”) AND recordAbstract:(Authentication Authorization Identity ”Access Control” -Hardware -Cryptography -Protocol -RFID) AND acmdlTitle:(-Study -Survey) AND keywords.author.keyword:(-Hardware -Physical -Network)
WoS	(”internet of things” OR IoT) AND TOPIC: (Security) AND TOPIC: (Authentication OR Authorization OR Identity OR ”Access Control”) NOT TOPIC: (Hardware OR Cryptography OR Protocol OR RFID OR Physical OR Network) NOT TOPIC: (Study OR Survey)
SpringerLink	(Authentication OR Authorization OR Identity OR ”Access Control”) + title (”Internet of Things” OR IoT)
ScienceDirect	TITLE-ABSTR-KEY(”Internet of Things” OR ”IoT”) AND TITLE-ABSTR-KEY(Authentication OR Authorization OR Identity OR ”Access Control”) AND KEY(Security) AND NOT (TITLE-ABSTR-KEY(Hardware OR Cryptography OR Protocol OR RFID) OR title(study OR survey) OR key(Physical OR Network))

Table 2: Number of articles processed in the survey.

Indexer	Results	Prefiltered	Relevant
IEEE Xplore	120	29	15
ACM DL	84	9	7
WoS	67	31	13
SpringerLink	33	8	6
ScienceDirect	27	9	2
Total	331	86	43

(viii) **healthcare:** projects that specifically address the healthcare domain [18, 22, 28, 32, 41, 53]

(ix) **attribute-based** subset of authorization proposals that involve ABAC [15, 28, 32, 34, 38, 53]

(x) **roles:** subset of authorization proposals that involve RBAC [18, 28, 29]

Two of the papers do not fit into any of the above categories [55, 56]. One article [55] is likely too short for RAKE to perform any meaningful analysis; with the second article [56], we do not identify any obvious reason for not being categorized. Nevertheless, both of the papers address authentication and we have included them in this category.

In total, over 50% of the articles get two or three keywords. Significant number of research papers fit into one or four categories. Two papers did not fit any category, and another three fit to five categories. This statistic is shown at Figure 2. As illustrated in Figure 3, research covered by this survey shows evident increase of interest in IoT security based on the amount of articles published, from a single paper in year 2013 through 4 in 2014 up to 11 in the 2015. There is a small decrease to 10 research publications in 2016. Year 2017 exhibits again growth to 17 papers for only 3 quarters of the year.

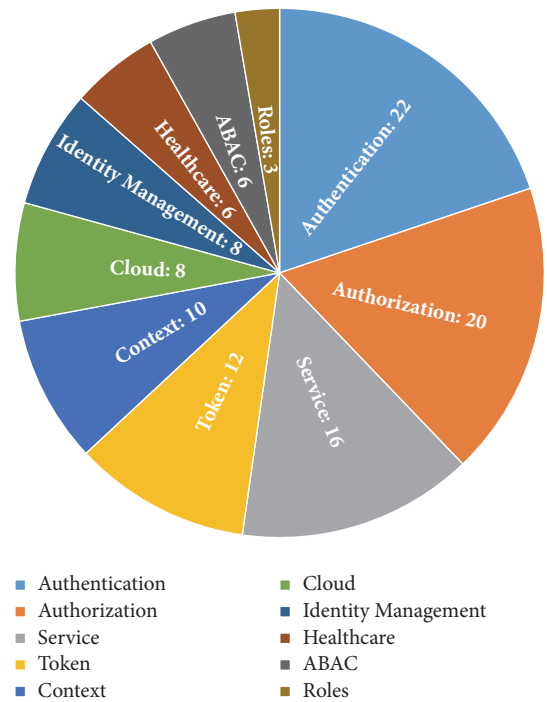


Figure 1: Number of keywords found across all articles.

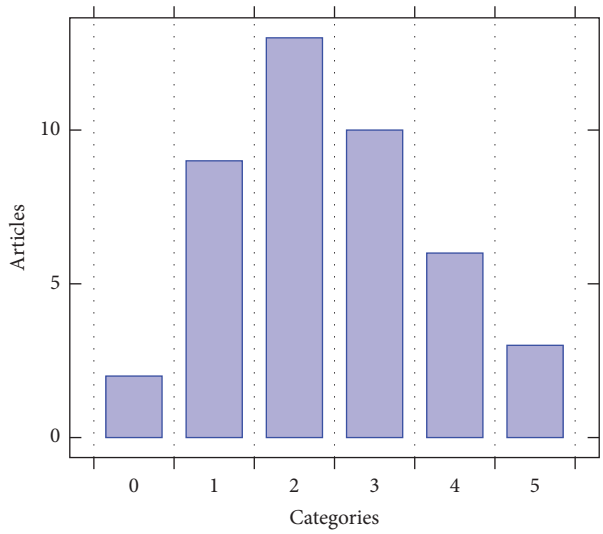


Figure 2: Number of categories suggested by RAKE per article.

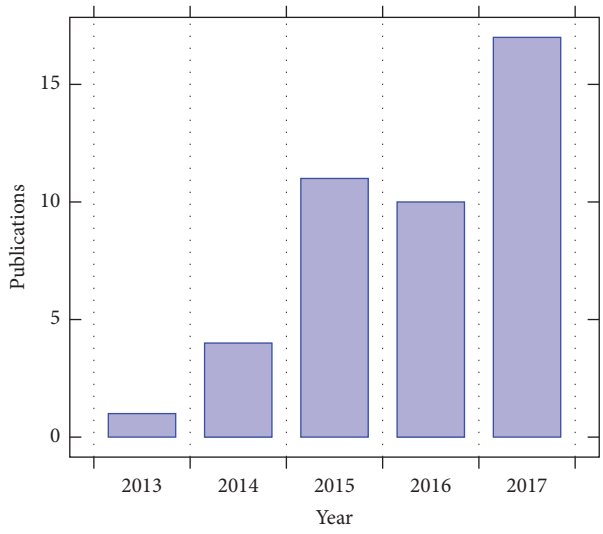


Figure 3: Number of publications per year. *Note that 2017 data is only through September.*

The authentication, authorization, identity management, and services categories are described in the subsequent subsections as they are the most populous categories. Articles with context-awareness elements are further described in their own section, which answers RQ2.

5.1. Authentication. Authentication is addressed by 22 papers from our pool—more than half of the articles in the survey. Authentication is the process of confirming an attribute claimed by an entity. In the vast majority of cases, it is confirmation of identity that the entity claims using credentials.

Traditional authentication methods, enhanced with multifactor authentication based on a location, are described in [14]. Their system considers user location, and they develop an additional factor for multifactor authentication which ascertains the physical possibility of a user being in a particular location; e.g., a user cannot possibly be in Los Angeles if they just logged in from New York. This adds additional security without requiring the user to perform extra actions.

In [15], the authors suggest enhancing privacy during authentication by basing authentication on attributes, rather than identities. A trusted authority issues certificates which prove that an entity possesses a particular attribute; these certificates are used for authentication when communicating with other services. This scheme preserves both entity privacy and the advantages of centralized identity management.

The authentication model for cloud-based IoT is elaborated by Barreto et al. [16]. Their solution supports two stages of authentication: one for basic and a second one for advanced access, e.g., administrative purposes. They do not describe specifically how the authentication should be done; rather they specify methods that cloud services should provide for authentication.

To achieve efficient and smart authentication of IoT devices, Cagnazzo et al. [17] suggest using Quick Response (QR) codes, specifically XignQR [70]. Every device has a printed QR code that contains important information about it, e.g., an ID representing its service provider, authentication server address, and digital signature. Scanning the QR code and sending it to the authentication manager allow the manager to decide which authentication method it should enforce on the user. This approach can be useful when physically managing large amounts of devices at the same location, e.g., in a hospital or in a factory.

A security framework following the Architecture Reference Model (ARM) [71] is described in [21]. It bases authentication on the Extensible Authentication Protocol (EAP) over LAN [72]. EAP is widely used and recognized as a mechanism to provide flexible authentication through different EAP methods. Those methods allow an EAP peer to be authenticated by an EAP server through EAP authentication for network access. While their work proposes interesting solutions, they do not provide any case study or usability study.

Kumar et al. [22] assume that the best authentication method for wearables and nearables (devices which are not worn, but are generally close to the user) is the biometric information of their owner. The proposed solution requires the user to register their biometric characteristic(s) in person with the authentication provider. Later, access points close to the user—wearables or nearables—capture the user's biometric information and authenticate them by comparing those characteristics with the registered characteristics. However, there is an issue with privacy as many users are reluctant to share their personal information.

Two almost identical works proposed the OpenID protocol as the method of authentication in the IoT environment [23, 29]. They describe a central service issuing tokens and communicating through a RESTful API [73] over the HTTP(s) [74] protocol, allowing rapid development and acceptance among IoT devices as all technologies used are proven, well-documented, and widely supported. A downside is that the OpenID protocol was not designed with IoT usage in mind and can be more demanding of computation and network resources than specialized protocols.

Another framework [24] for authentication is formally described using process algebra, specifically CSP [75]. The framework contains three authentication forms. An *entity* authentication is the capability of verifying the identity that the entity claims. An *action* authentication refers to authentication of the actions of devices and whether they are allowed. A *claim* authentication verifies the authenticity of devices' claims about previous actions. It also has three strength levels for each form—weak level, noninjective level, and injective level. The paper does not provide any proof of concept or other kinds of demonstration of their solution.

A mechanism of HTTP(s)-based authentication for IoT devices using a hash-chain generated between server and the client is explained in [25]. This hash-chain is generated during the login process and serves as a One Time Password for the client to authenticate against services. If a device does not have the required capabilities (battery lifetime, computational power, network connection, etc.) to generate the hash-chain, or those capabilities are in use for other functions, another device acting as a proxy may be used to generate the hash-chain.

Continuous authentication of personal IoT devices is addressed by Shazad et al. [26]. Current practice is to authenticate an entity just once when a session is established and keep them authenticated until some timeout occurs, or the session is otherwise closed. This session persistence presents a potential security risk. The authors divide devices into two categories: those which maintain physical contact with the user and those which do not. Devices that keep contact can be authenticated using various biometric information, both direct (blood flow rhythm) and indirect (using inertia measurement unit to check a user's gait). For devices that are not in physical contact with the user, the authors propose using radio frequency signals. For example, Wi-Fi signals are reflected by the human body and the resulting distortions can be measured and used to determine users' walking speed, gait cycle, and other physical properties.

Advanced authentication methods better than the current approaches are suggested in [27]. Most of the traditional methods have flaws or were not designed to be frequently used (e.g., passwords—almost no one can memorize strong and unique passwords for every service or device they use, so users reuse their passwords). Their proposal is based on users' digitized memories. Users would authenticate themselves against their digitized memories based on date and time, place, people or pets, devices, habits, audio, or ownership recognition. They map different suitable methods, including choice selection, alphanumeric input, image part selection, or interactive categorization.

Wiseman et al. [31] present a niche but interesting problem along with a solution. They address the issue of pairing an IoT device with its "master" account. Connecting from devices using a password can be difficult or even impossible because of the lack of a proper input method. One method to avoid this is to let the device display an access code and add the access code to the master account. They examine this process from a user experience perspective and compare convenience between alphanumeric codes and codes generated from human-readable words.

A privacy-preserving, decentralized identity management framework for the IoT is presented in [35]. Identity in the IoT is extended not only to users but also to IoT devices themselves using an ARM-compliant, claims-based approach built on top of Identity Mixer technology [76]. They define partial identities as subsets of user or device virtual identities that preserve privacy while being sufficient to provide identity confirmation. They show a use of their framework with Distributed Capability-Based Access Control [21]. Identity attributes are disclosed by specific proof and are employed during authorization based on XACML rules to obtain capability tokens used to access a service.

Finally, there is a group of papers [18–20, 28, 30, 32–34] that address authentication tangentially either as part of a broader and more complex framework or project, or to solve authentication issues as a side effect of dealing with another problem.

Table 3 presents an overview of authentication research, reflecting the information we extracted from the papers. It shows which solutions support centralized and decentralized architectures, which are oriented for user-to-machine (U2M) or for machine-to-machine (M2M) communication, which possess at least some elements of context-awareness, the paper's primary domain, or which authentication factor is used for the primary authentication ("Inherence" is "Something I am"; "Possession" is "Something I have"; "Knowledge" is "Something I know").

5.2. Authorization. Authorization is the process of granting permissions on specific actions to given entities—in our scenario specifically to users, devices, or applications. There are a total of 20 articles in the identified pool addressing this topic. Authorization ties with services as the second most populous category.

Access control based on trust in an ARM-compliant model is proposed by [36]. It describes various levels of trust, a multidimensional attribute which describes various concerns in the network the authors call dimensions: quality of service (including network availability and throughput), security (authentication and authorization protocols, encryption, etc.), reputation (recommendations from other devices), and social relationship (the group or groups of IoT devices to which an individual device belongs, e.g., those made by a certain manufacturer or currently in a particular location). This trust is used for final authorization within the environment.

The authors of [18] describe a complex framework for use in the healthcare field. They employ a version of RBAC where a user, specifically a patient, grants permission to access his data based on a particular role—a group of doctors and nurses. A centralized authentication server enforces the resulting security rules.

Another paper [37] develops an authorization architecture based on IoT-OAS [77], authenticating users using tokens similar to those used in OpenID. Every device has a designated owner and a set of actions or permissions. Users may request and share permissions with one another; multiple operational cases are described in the paper.

Gerdes et al. [20] tackle the problem of authorization and authentication for devices with constrained computational

Table 3: Summary of authentication articles.

Article	Centralized	Decentralized	U2M	M2M	Context-aware	Factor	Domain	Specifics
[14]	Yes	Yes	Yes	No	Yes	Inherence	Any	Service answering whether user can be in the given location
[15]	Yes	Yes	Yes	Yes	No	Possession	Any	Use of attributes for authentication
[16]	Yes	No	Yes	Yes	No	N/A	Cloud	Authentication through cloud
[17]	Yes	Yes	Yes	No	No	N/A	Any	Reading QR codes physically present on a device
[18]	Yes	Yes	N/A	N/A	No	Knowledge	Healthcare	Framework designed to preserve patient privacy
[19]	Yes	No	Yes	Yes	No	Knowledge	Any	Adjustment of Web API management; OpenID Connect
[20]	No	Yes	Yes	Yes	No	Possession	Any	Authentication for devices with constrained computational power
[21]	Yes	No	No	Yes	No	Knowledge	Any	ARM compliant; EAPoL; RADIAL
[22]	No	Yes	Yes	No	No	Inherence	Healthcare	Biometric from wearable and nearables
[23]	Yes	No	Yes	Yes	No	N/A	Healthcare	OpenID Connect
[24]	Yes	No	Yes	Yes	No	N/A	Any	Authentication framework mathematical description using CSP algebra
[25]	Yes	No	Yes	Yes	No	N/A	Any	HTTPS-based device authentication using hash chain as One Time Password
[26]	N/A	N/A	Yes	No	Yes	Inherence	Any	Biometric; continuous authentication
[27]	Yes	No	Yes	No	Yes	Knowledge	Any	User's electronical history
[28]	Yes	No	Yes	Yes	No	N/A	Healthcare	Authentication based on attributes
[29]	Yes	No	Yes	Yes	No	Knowledge	Any	OpenID Connect
[30]	N/A	N/A	No	Yes	No	N/A	Any	WS-Security adaptation for IoT
[31]	N/A	N/A	Yes	No	No	Knowledge	Any	One time passwords using words chosen by a user
[32]	Yes	No	Yes	Yes	No	Possession	Healthcare	Full security framework
[33]	No	Yes	Yes	Yes	No	N/A	Any	Blockchain access control framework
[34]	N/A	N/A	No	Yes	No	Possession	No	Authentication on perception level
[35]	No	Yes	Yes	Yes	Yes	Knowledge	Any	Privacy preserving based on partial identities

power. The authors divide IoT devices into the categories "constrained" and "less-constrained" based on resource availability and allow less-constrained devices to perform some authorization functions on behalf of the constrained devices. The paper includes basic methods for these authorization management tasks, and "principal actors", which represent the person or company that owns the specific device or the data on the device, must set appropriate policies for each situation about which tasks can or cannot be offloaded.

One solution to the problem of data access control across a shared network is developed in [38]. The authors use Ciphertext-Policy Attribute-Based Encryption [78] and enhance it with a set of policy descriptions in a XML file. Access policies are based on entity attributes and structured as a binary tree with "And" and "Or" operations available. Entities present a keyserver with a list of their attributes, and the keyserver generates a key which can only decrypt data to which the listed attributes allow access.

A framework introduced in [21] supports not only authentication but also authorization, enabled by creating an Authorization Server which issues access tokens according to security rules stored in XACML [79], an XML schema for representing authorization and entitlement policies. Entities request authorization tokens based on their attributes and then use the tokens to access services provided by or data stored on another server or device.

Kurniawan et al. find classic security strategies unsuitable because they are centralized and scale poorly in the IoT environment. They propose a trust-based model [39] based on Bayesian decision theory. The authors compute Bayesian trust values based on three inputs: experience (the history of interactions between the actors), knowledge (what is already known about the entity and the context), and recommendation (how much trusted peers trust the entity in question) and use these trust values as input to a loss function that determines the cost of an action. Access control decisions are made based on the output of the loss function, given a particular trust value.

Two proposals based on the existing OAuth protocol [80] use tokens that encode the access rights (e.g., roles or attributes) of the token owner and a configurable lifespan. The first method [41] uses JSON Web Tokens [81]; the second proposal [23] uses a special token format which allows a limited number of accesses. Both proposals communicate through a RESTful API.

Another framework for securing API-enabled IoT devices in smart buildings [43] is also inspired by OAuth and uses JSON Web Tokens. The proposed security manager is split into two services to enable better scalability. The first service is an authentication manager which authenticates users or services with a process similar but not identical to OAuth and issues a JSON Web Token. The second service is an access control manager that verifies whether the access is allowed, based on XACML rules set by the system administrator and the identity of the requesting side (which is provided by the token).

Blockchain technology is used in [40] to store, distribute, and verify authorization rules. Every node in the network has a full database of all access control policies for each resource-requester pair in the form of transactions. Access is granted by giving a token to the requester entity and propagating it in the blockchain. The blockchain also serves as an auditing and logging tool. Trust in the network is based on the distributed nature and large size of that network; it is very difficult to gain unauthorized access or disable the network by attacking a central element. A slightly different approach using blockchain is presented in [33]. Rules based on OrBAC [82] are distributed through a block chain, and based on the history of the communication, the rules are updated with reinforced learning algorithms.

Tasali et al. [28] discuss current standards for healthcare devices, including Integrated Clinical Environment (ICE) [83] and Medical Application Platform (MAP) [84]. The conclusion is that they barely address authorization and authentication (if they address it at all). Their solution is based on ABAC, enhanced with attribute inheritance inspired by RBAC. Attribute inheritance allows "plug-and-play" configuration of new devices based on device types represented as attributes preset on the devices.

Another option is to isolate each function of the device and provide access just to that functionality [28]. Functionalities are slightly similar to the concept of micro services. The proposed functionality-centric access control framework also reduces application level attacks on "misused functionality" or "reduced functionality".

A proposal for energy constrained devices called Time Division Multiple Access is described in [44]. The schema is well suited for sensors with known communication patterns, such as a repeating communication schedule in which sensors periodically report data. The proposed communication scheme optimizes the tradeoff between device lifetime and distortion of the data transmitted. Another different application of ABAC focused on reducing storage and communication overhead is described in [34].

Sicari et al. provide a full specification for a security framework for smart healthcare [32]. It describes three main points (locations) for policy enforcement, a policy administration point, a policy enforcement point, and a policy decision point. The access roles are described using XML in a format inspired by ABAC.

Another access control model for IoT running in the cloud [45] secures data using hierarchical attribute-based encryption. The encryption is done in two steps. The first part of encryption is done on the device; the secondary encryption is done on the gateway. This reduces the load on the device. Decryption is likewise split between the cloud and the device in order to save application resources. The hierarchical nature of the encryption scheme allows updating security policies using an update key based on information from the data source, without the device itself needing to reencrypt the data.

Two of the reviewed papers [26, 30] discuss authorization only tangentially. The complete overview of authorization research can be seen in Table 4.

5.3. Identity Management. Identity can be viewed as a set of user's attributes, both virtual or real. Identity management is the mechanism of storing and retrieving user identities. Typically, users are forced to have more unconnected identities for

Table 4: Summary of authorization articles.

Article	Centralized	Decentralized	U2M	M2M	Context-aware	Policies creator	Domain	Policies based on	Specifics
[18]	Yes	Yes	N/A	N/A	No	Data creator	Healthcare	Roles	Rules tied to the data
[20]	No	Yes	Yes	Yes	No	Resource owner	Any	N/A	Constrained devices
[21]	Yes	No	No	Yes	No	Administrator	Any	N/A	ARM compliant; describes access control generally
[23]	Yes	No	Yes	Yes	No	Administrator	Any	N/A	OAuth; tokens
[26]	N/A	N/A	Yes	No	Yes	N/A	Any	Data itself	Biometric information used
[28]	Yes	No	Yes	Yes	Yes	Administrator	Healthcare	Attributes	Supports with attribute inheritance
[30]	N/A	N/A	No	Yes	No	Administrator	Any	N/A	WS-Security adaptation for IoT
[32]	Yes	No	Yes	Yes	Yes	Administrator	Healthcare	Attributes	Full security framework
[33]	No	Yes	Yes	Yes	No	Resource owner	Any	OrBAC	Reinforced learning to update rules
[36]	Yes	Yes	Yes	Yes	No	N/A	Any	Attributes	ARM compliant; Attributes extended with trust based on various concerns in the network
[37]	No	Yes	Yes	No	No	Resource owner	Any	Direct grants	Tokens; Possible to share permissions
[38]	No	Yes	N/A	N/A	Yes	Data creator	Any	Attributes	Data decryption only with correct attributes
[39]	No	Yes	N/A	Yes	Yes	System	Any	Bayesian decision	Bayesian decision theory for authorization
[40]	No	Yes	Yes	Yes	No	Resource owner	Any	Direct grant	Propagation through blockchain
[41]	Yes	No	Yes	Yes	No	N/A	Healthcare	N/A	OAuth; tokens
[34]	N/A	N/A	Yes	Yes	Yes	Resource owner	Any	Attributes	Perception layer framework
[42]	Yes	Yes	Yes	Yes	No	Resource owner	Any	Direct grant	Access control specified for functionalities
[43]	Yes	Yes	No	Yes	No	Resource owner	Smart building	Attributes	OAuth; XACML; tokens
[44]	N/A	N/A	No	Yes	No	N/A	Any	N/A	Constrained devices
[45]	Yes	Yes	No	Yes	No	Data creator	Cloud	Attributes	Gateway, device and cloud share data encryption

various services. In the IoT environment, the identity should be available for the whole IoT network, while preserving user's privacy, although it does not mean that each user must have single identity. The identity concept is also extended from users to include sensor identities in the IoT. Identity management is closely connected to authentication, which is process of verifying that a user (or a device) actually is the owner of that identity, as well as to authorization, which is the process of granting access to a resource based on user attributes (i.e., identity). Eight of the articles address identity or identity management at least partially.

Traditionally, user identity contains the principal along with credentials used for authentication. This renders a privacy risk, especially if the identity is shared with multiple services whose operators are not known in advance, and that might appear on and disappear from the network at any time in the dynamic IoT concept. Many of the articles tackle the problem of privacy by limiting a user's identity to only their attributes, without any unique information that could lead to disclosure of their identity. One of the proposals is for a trusted party to issue cryptographic containers containing user attributes [15]. It is not specified that the trusted party must be single entity in a network, so we can assume that multiple trusted parties can exist simultaneously. Also [50] proposes using attributes instead of identity for authorization. Gusmeroli et al. propose a slightly different approach with using capabilities instead of attributes [46]. This proposal also supports anonymous capabilities which allows authentication without disclosing identity.

The problem of assigning identity to devices is described in [47]. An IoT device inherits the identity of its user through various methods based on a relationship between the user and the device. They formulate methods for devices that are strictly connected to a single user, as well as identity extensions from users to devices that change users frequently.

A complete framework for decentralized identity management to enhance user privacy is introduced in [35]. It defines partial identities as the least sufficient subsets of full identities for a requesting service that do not disclose any unnecessary information about a user.

The principle of storing a user's biometric information in access points, serving like identity servers, and thus linking a real user's identity with his virtual identity through wearables is described in [22]. Two final articles [18, 34] deal partially with privacy and data transmission encryption.

5.4. Services. This section presents an overview of the solutions that either support IoT-as-a-service or provide security-as-a-service. This means that at a minimum the security client (an entity) or security provider follows the principles of Service Oriented Architecture (SOA) [85]. Frequently, both of the actors can be viewed as services. In this survey we have 16 research publications that include SOA compatibility, although not every paper in this category uses the term SOA or "service"; instead, they are frequently called by synonyms, e.g., "central entity", "authorization or authentication server", and so forth. Most of the centralized security approaches can be viewed as a service.

Most of the surveyed proposals contain an identity management, authentication, or authorization service. An application in the IoT environment may offload the authentication process to such a security service [16, 17, 21, 23, 27, 32, 35, 47]. A few proposals also allow the distribution of access rights or other properties used for authorization from the service to its clients [15, 21, 46]. Some of the services also provide additional features like enhanced user privacy [15, 25, 46]. They anonymize entity identities by hiding identity details from the service provider and guarantee the entity's identity by the trustworthiness of the identity management service itself.

Two of the papers in this category stand out. The first adapts the Web Service (WS) Security specification [86], which is intended for loosely coupled distributed systems, to the IoT environment by extending it to allow identity management functions to be offloaded from computationally "weak" devices to "strong" ones [30]. The proposed method, termed DPWSec, also simplifies the original WS-Security specification by removing unneeded portions: multihop security, statelessness, hosting and hosted devices, and the device profile communication model. The second paper describes a security framework within the scope of the Device Profile for Web Services using the XACML standard for rule description [49]. It describes three parts of the framework—the policy enforcement point (where the policies are enforced), policy decision points (where the policies are evaluated), and policy information points (where the audit logs are kept).

6. Context-Awareness

One trend in contemporary application development is a movement towards context-awareness. Context is defined by Abowd [59] as any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves. In the context of the IoT it can be extended to not only interaction between a user and an application but also between two applications.

Solutions using context-aware security can provide a much better user experience as well as increased security [87]; often both can be achieved at the same time. Nevertheless, the level of interest in context-awareness from a security perspective has not reached the same level as interest from the user experience perspective, likely because computer security is traditionally a more conservative domain of computer science. In this section we focus on research that does speak to an interest in context-aware security.

Most common approach to achieve context-aware security is using ABAC. It differs from RBAC in that an entity (a user or a device) performing an action is not authorized based on matching the roles it is assigned to roles which allow certain actions. In ABAC, every action is mapped to a specific set of attributes an entity must possess in order to take that action. An example of such a rule for reading a document is that the entity must be from the same department as the creator of the document, must be employed in a management position, and must be located in the same building or complex.

One option is to specify access rules using ABAC for every piece of data at creation time and join those rules with the data so that during network transportation, updates, or copying, the rules stay consistent. In order to manipulate the data, an entity must possess the specified attributes [38]. Another method is to use a three-module architecture. The first module, a policy enforcement point, is responsible for invoking checks on access rules. The second, a policy information point, gathers information about an entity's attributes, including their context. Finally, a policy decision point compares security rules with the information gathered about the entity and decides whether the action is allowed or declined [28, 53]. Security rules can be written in XML using XACML [28] or using the Ontology Web Language [53]. While [32, 34, 48, 50] do not mention context information specifically, the ABAC implementations in those papers could also utilize context-aware attributes.

Instead of extending ABAC, another option is to adapt the well-described Capability-Based Access Control (CBAC) [88] architecture. A capability (known in some systems as a key) is a communicable, unforgeable token of authority. It refers to a value that references an object along with an associated set of access rights. This token may contain additional contextual rules, defined in XACML format, which must be satisfied for the token to be valid [46]. Variation on this is using Distributed Capability-Based Access Control [21] as described in [35].

A novel authorization architecture based on Bayesian decision theory [39] also considers context. The trust parameters of history, knowledge, and reputation (described in the Authorization section) may include contextual elements which are either acquired directly by the device itself or provided indirectly by a peer device. Machine learning techniques used to enhance access rights [33] also consider context in terms of a history of the previous interaction.

Biometric information may be considered "contextual" by definition, so biometric authorization is context-aware [22, 26]. Many devices, especially wearables, directly measure the user's physical traits (e.g., heart rhythm or body temperature). Other "nearable" devices can provide additional information such as weight or gait, both of which can be measured by video sensors. All of this information can be compared to a user's known physical or kinesiological properties.

Beyond simple biometric data, a user's digital life may be considered as context for identity management. A user's photos, videos, blog posts, and browsing history can be used to authenticate that user [27]. Given sufficient digital history, security questions can be devised which no one but the authentic user can answer. This has the benefit that the user does not need to memorize passwords or carry other credential material; their own memories are sufficient. Another similar proposal, which restricts context to information from network traffic, authenticates using contextual information provided by a smart home [51].

7. Existing versus Novel Approaches

Existing research projects in IoT security that propose an actual solution or method can be roughly aligned to two categories: those which extend or adjust existing architectures or programs to better suit the IoT environment and those which propose entirely new ideas to solve environment-specific problems. However, the classification is not strictly binary, and it is often difficult to judge the novelty of any particular proposal. The reader will note that all research is meant to be "novel"; we use the word here in a narrower sense to mean an entirely new approach which does not make use of existing technologies or standards.

The works we considered that apply or adapt existing technologies and methods from other security domains to the IoT environment often consider OAuth 2 technology [19, 24, 29, 41, 43]. Two proposals also adopt the WS-Security specification to IoT devices and communication between them [30, 49].

The most innovative solutions share some common properties. All of them are suitable for distributed use and none require administrator interaction. They can handle device connection and disconnection as well as security rule distribution and validation. Often the responsibility for creation of access rules is moved from administrators to data owners. Two papers show operation with trust between devices and dynamic calculation of trust among various communication partners [36, 39, 50]. One proposal adjusts ABAC to be more dynamic and allow a device to pick its own attributes; other devices must subsequently confirm that the device really does possess the claimed attribute. Security rules are set during data creation using ABAC and then connected to those data for its whole life-cycle [48]. Other innovative approaches suggest propagating all security rules through a blockchain in the network [40] and possibly update them based on reinforced learning algorithms [33]. Access would be granted for a specific entity to a given resource and distributed using blockchain as described earlier. One of the researches proposes access control based not on roles or attributes, but rather on functionalities of the IoT node [42]. Access control for cloud applications based on attributes [45] using computational power of sensor gateways and the cloud itself is suitable for constrained devices.

In summary, there are various novel proposals [33, 36, 39, 40, 42, 45, 48, 50], especially focusing on distributed solutions [33, 39, 40, 45, 48, 50], that potentially suit the IoT environment better in terms of scalability, maintainability, and flexibility, but due to their novelty it is difficult or impossible to predict which ideas might be adopted or see wide use. A significant amount of research [19, 23, 29, 30, 34, 41, 43, 49] is focused on adoption of existing technologies; all exhibit promising results.

8. Distribution versus Centralization

The IoT is a diverse, complex, and fast changing environment. It comprises a large number of devices that interact autonomously. Objects also appear and disappear autonomously and with high frequency. Given these differences from a more standard network environment, we focus in this section on what paradigms are used in the security solutions.

A conventional, centralized approach is very easy to set up, maintain, and audit for system administrators. It also

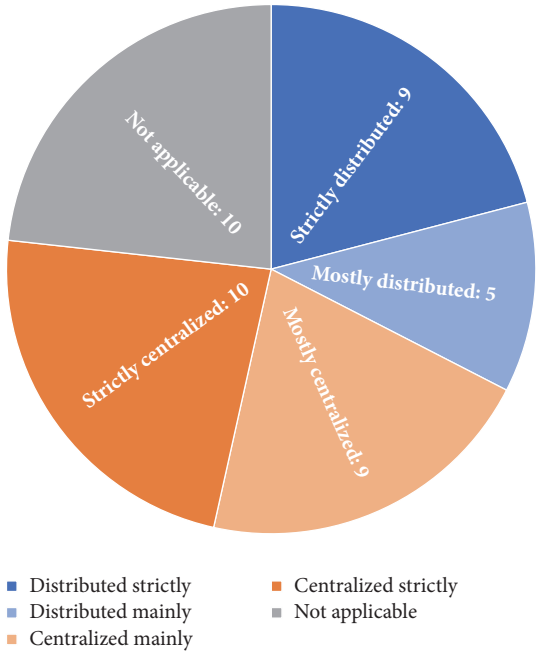


Figure 4: Categorization of distributed and centralized solutions.

presents a stable point in the network from which users and application can build trust. Implementing centralized solutions is simpler both for the central server and for applications using it. Many of the existing centralized solutions for networks and application can be extended to operate in the IoT environment without overly costly adjustments. However, using a centralized architecture in the IoT presents several drawbacks, including limited flexibility and scalability.

By contrast, the attributes of distributed architectures are completely opposite. They scale well and are built with flexibility as a main goal. However, synchronization, maintenance, and auditing present serious difficulties. There is also the issue that no single trusted central entity stands behind them, which may be required by business users, legal entities, or others.

To further complicate matters, the line between distributed and centralized solution is often not clear. While some solutions can be considered exclusively in one category, there are a significant number of proposals that may work under both paradigms. Figure 4 shows a chart of distributed and centralized solutions.

Requiring a central server for identity management prevents distributed operation for obvious reasons. Sometimes this limitation is imposed for domain-specific reasons (e.g., in the healthcare domain [18, 22, 32, 41]); other times it arises simply as a function of the technologies or methods employed [19, 23, 29, 51, 52]. In one proposal, the authentication method requires having as much historical data about an entity as possible, to the point that authentication data storage requirements make it impractical to host such data at multiple locations [27].

At the other end of the spectrum, the technologies used in some proposals specifically preclude centralization. For instance, methods which rely on the creators of data to specify security rules, or which grant access selectively, do not operate with a central server [20, 35, 37, 39, 48, 55]. Blockchain-based access rule verification [33, 40] also can not be centralized, and the same applies to extensions of the ABAC system which rely on peer devices to confirm an entity's attributes over the network [50].

Most of the ideas in the papers surveyed can be used in both centralized and decentralized architectures. Centralized solutions can be often decentralized by multiplying central elements [14–16, 21, 25, 28, 36, 42, 43], and decentralized proposals can be centralized by limiting the number of security control elements to single node [17, 38, 45, 46, 50]. Similarly, some of the research we reviewed [26, 30, 31, 34, 36, 44, 49, 52, 54, 56] cannot be categorized in either category. They work equally well for either architecture without modification and can be seen as complementary extensions for complex security solutions, helping with particular issues (e.g., authentication, auditing, context-awareness).

9. User versus Device-Centrism

In IoT two basic communication patterns exist: either users interact with devices, or devices interact among themselves. The first type is designated U2M category. The other scheme of communication is designated M2M. Some of the proposals fit both patterns; this section explains how the security models support particular communication models, and the limitations of those models.

One important restrictive factor is the need for human input to the interaction. In some cases, various information about the actual user is required for security reasons: biometric information [22, 26, 56], a user's digital history [27], or a user's location [14]. Other approaches require direct user interaction such as scanning QR codes [17] or using words to generate a password which connects a master account with the particular device [31]. Any of these cases requires U2M communication.

Generally a device is capable of constant and repetitive tasks, but its decision capabilities are limited: goals or objectives can only be set by a user. Users, on the other hand, may find monotonous or continual-load requirements onerous at best and impossible at worst. Given these differences in capability, the adaptation of existing M2M security technologies [21, 30, 34, 43, 55] works well for IoT scenarios where a user is not required. Proposals exist for M2M authentication even with low-resource devices [20, 44, 45].

Finally, many of the solutions described in U2M research can be used for M2M identity management with little to no modification [14, 18, 24, 32, 35, 37, 53] and vice versa [15, 28, 33, 36, 39, 40, 42, 49, 50]. Some of the research even includes existing U2M technologies being used for M2M purposes [23, 29, 41], and many of the papers surveyed are useful for either communication model [16, 19, 25, 38, 46–48, 51, 52, 54].

Figure 5 shows that research contributions in the U2M communication model occur with similar frequency to those in the M2M model. The vast majority of projects can be used for either communication scheme, which demonstrates the versatility of the security solutions and proposals.

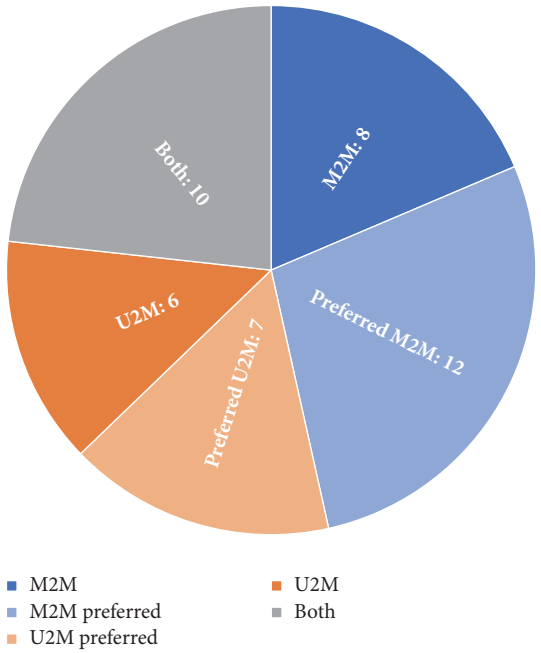


Figure 5: Categorization of U2M and M2M solutions.

10. Threats to Validity

Surveys are a highly subjective type of research and therefore suffer from threats to validity. We have identified several threats that need to be addressed or at least mentioned. In order to eliminate most of them, we have followed recommended guidelines for conducting systematic studies [10].

Our evidence selection is based on professional indexing sites. We could miss some articles published in other sources (e.g., journals not indexed in WoS). Also, the queries we use to search for articles explore only abstracts. This means that articles which should have been included may not have been because they contained some of the excluded words or because they did not contain any of the included words. We tried to eliminate this by testing our queries against the manually selected control set.

Data extraction bias is another possible threat to validity. We addressed this primarily by ensuring that each paper received several individual reviews focused on each research question. Using the RAKE algorithm to extract paper keywords also mitigated data extraction bias somewhat because the same extraction method was applied to each paper, apart from any human factors.

Exclusion and inclusion of the papers due to their scope are also potential threat. To mitigate this threat we followed methods for the selection criteria suggested in [10]. We have read numerous related works and spent considerable time reading the selected papers to assure they fit within our considered scope. We removed papers that focus specifically on cryptography, networking, and low level device security. We have also excluded papers that do not provide specific results, that list only suggestions or opinions without solution proposals.

All of the papers were treated equally in the survey, although not all of the published research has the same quality or impact on the community. We provide some overview of each article's impact in Table 5, including metadata about the impact of the paper and possible quality of the source of the publication. To measure community impact we have chosen two sources: data from publishers and Google Scholar [89]. Publishers generally provide their own list of citing works and a number of downloads (or views) of the work. One disadvantage of using this publisher-provided data is that it may often miss citations from sources unknown to it. Therefore, Google Scholar was chosen as a universal, most fully populated article aggregator. It provides its own list of citations, but they include self-citations and it may take up to few months for articles or citations to appear there. To quantify quality of the publishing media we chose two methods. For journals we use Impact Factor [90] from Web of Science as it is the most prominent and possibly oldest journal indexing tool. Ranking conferences proves to be more difficult. The most appropriate measure for our needs seems to be the Computing Research Education (CORE) Association of Australasia conference ranking [91] as it presents independent rankings of conferences with any sponsor. It ranks conferences with letters C, B, A, and A$*$ for its quality (A$*$ is the best, C is the worst). A disadvantage is that not all conferences are included in the ranking and the ranking itself is managed by a small group of scientists from a particular geographic area.

11. Conclusion

This paper provides an overview of the accomplished research and challenges in the security domain of IoT, especially for authentication and authorization. It contains the most recent research and categorizes it from multiple perspectives. It shows how context-awareness extends security and what approaches exist to incorporate context-awareness into IoT security. It shows how existing and current, widely adopted technologies are adapted for the IoT and surveys new security proposals designed specifically for that environment. We discussed whether security solutions for centralized or distributed architectures are favored and analyzed whether machine-to-machine or user-to-machine security is more prevalent in the current research.

To our best knowledge there is no similar study or survey of IoT security or any other study containing the latest IoT security research. We believe that this overview will help readers gain an overall picture about the state of IoT security research, allowing them to reapply existing knowledge and deal with the security issues that are preventing IoT popularity and adoption from increasing among end users.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Fulbright Program and the Grant Agency of the Czech Technical University in Prague, Grant no. SGS16/234/OHK3/3T/13.

Table 5: Community impact of articles.

Article	Published in	IF or CORE ranking	Year	Source citations	Source citations per year	Google citations	Google citations per year	Views	Views per year
[14]	Conference	A	2016	0	0	2	1	244	122
[15]	Conference	N/A	2016	0	0	6	3	342	171
[16]	Conference	N/A	2015	3	1	15	5	246	82
[17]	Journal	N/A	2016	2	1	1	0.5	409	204.5
[18]	Journal	1.405	2017	0	0	0	0	374	374
[19]	Conference	A	2015	1	0.33	7	2.33	766	255.33
[20]	Book chapter	N/A	2015	0	0	N/A	N/A	N/A	N/A
[21]	Journal	8.085	2015	45	15	57	19	1701	567
[22]	Conference	B	2017	0	0	4	4	207	207
[23]	Journal	1.239	2017	0	0	0	0	355	355
[24]	Conference	B	2014	4	1	0	0	945	236.25
[25]	Conference	N/A	2016	2	1	0	0	359	179.5
[26]	Journal	1.521	2017	2	2	1	1	1269	1269
[27]	Conference	C	2015	3	1	6	2	183	61
[28]	Conference	C	2017	0	0	0	0	160	160
[29]	Journal	N/A	2017	0	0	0	0	407	407
[30]	Conference	N/A	2015	0	0	2	0.67	195	65
[31]	Conference	A∗	2016	1	0.5	1	0.5	252	126
[32]	Journal	N/A	2017	0	0	5	5	N/A	N/A
[33]	Journal	N/A	2017	N/A	N/A	2	2	N/A	N/A
[34]	Journal	1.232	2014	N/A	N/A	73	18.25	N/A	N/A
[35]	Journal	0.849	2017	1	1	1	1	313	313
[36]	Journal	2.472	2016	17	8.5	30	15	1100	550
[37]	Conference	N/A	2015	1	0.33	4	3.33	180	60
[38]	Conference	C	2015	0	0	4	1.33	164	54.66
[39]	Conference	N/A	2015	1	0.33	1	0.33	297	99
[40]	Conference	N/A	2017	2	2	12	12	359	359
[41]	Conference	N/A	2016	0	0	0	0	394	197
[42]	Conference	C	2017	0	0	1	1	165	165
[43]	Conference	B	2016	2	1	3	1.5	253	126.5
[44]	Journal	4.951	2017	0	0	1	1	N/A	N/A
[45]	Journal	1.405	2017	2	2	5	5	370	370
[46]	Journal	2.02	2013	65	4.33	111	22.2	N/A	N/A
[47]	Conference	N/A	2017	1	1	0	0	511	511
[48]	Conference	N/A	2016	0	0	1	0.5	732	366
[49]	Conference	N/A	2014	5	1.25	10	2.5	97	24.25
[50]	Journal	10.435	2017	2	2	4	4	693	693
[51]	Conference	B	2015	11	3.67	43	14.33	1638	548
[52]	Conference	C	2015	2	1.67	1	0.33	272	90.67
[53]	Conference	N/A	2014	1	0.25	3	0.75	337	84.25
[54]	Conference	B	2017	1	1	1	1	238	238
[55]	Conference	N/A	2017	0	0	0	0	363	363
[56]	Conference	N/A	2016	0	0	1	0.5	315	157.5

References

[1] P. Baran, "On distributed communications networks," *IEEE Transactions on Communications*, vol. 12, no. 1, pp. 1–9, 1964.

[2] J. C. R. Licklider, "Memorandum for members and affiliates of the intergalactic computer network, Technical report," Tech. Rep., Advanced Research Projects Agency, April 1963.

[3] V. G. Cerf and R. E. Kahn, "A Protocol for Packet Network Intercommunication," *IEEE Transactions on Communications*, vol. 22, no. 5, pp. 637–648, 1974.

[4] B. M. Leiner, V. G. Cerf, D. D. Clark et al., "A brief history of the internet," *Computer Communication Review*, vol. 39, no. 5, p. 22, 2009.

[5] C.-L. Hsu, H.-P. Lu, and H.-H. Hsu, "Adoption of the mobile Internet: an empirical study of multimedia message service (MMS)," *Omega*, vol. 35, no. 6, pp. 715–726, 2007, Special Issue on Telecommunications Applications.

[6] G. Wu, S. Talwar, K. Johnsson, N. Himayat, and K. D. Johnson, "M2M: from mobile to embedded internet," *IEEE Communications Magazine*, vol. 49, no. 4, pp. 36–43, 2011.

[7] L. Atzori, A. Iera, and G. Morabito, "The internet of things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[8] Gartner Inc., "Hype cycle for the internet of things 2017," Technical report, July 2017.

[9] G. Jayavardhana, B. Rajkumar, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," in *Future Generation Computer Systems*, vol. 29 of *Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications - Big Data, Scalable Analytics, and Beyond*, pp. 1645–1660, 7 edition, 2013.

[10] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: an update," *Information and Software Technology*, vol. 64, pp. 1–18, 2015.

[11] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: the road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2015.

[12] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.

[13] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A Survey on Security and Privacy Issues in Internet-of-Things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250–1258, 2017.

[14] I. Agadakos, P. Hallgren, D. Damopoulos, A. Sabelfeld, and G. Portokalidis, "Location-enhanced authentication using the IoT because you cannot be in two places at once," in *Proceedings of the 32nd Annual Computer Security Applications Conference, ACSAC 2016*, pp. 251–264, USA, December 2016.

[15] G. Alpár, L. Batina, L. Batten et al., "New directions in IoT privacy using attribute-based authentication," in *Proceedings of the ACM International Conference on Computing Frontiers, CF 2016*, pp. 461–466, NY, USA, May 2016.

[16] L. Barreto, A. Celesti, M. Villari, M. Fazio, and A. Puliafito, "An authentication model for IoT clouds," in *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2015*, pp. 1032–1035, August 2015.

[17] M. Cagnazzo, M. Hertlein, and N. Pohlmann, *An Usable Application for Authentication, Communication and Access Management in the Internet of Things*, Springer International Publishing, Cham, Switzerland, 2016.

[18] F. Chen, Y. Luo, J. Zhang et al., "An infrastructure framework for privacy protection of community medical internet of things: Transmission protection, storage protection and access control," *World Wide Web*, pp. 1–25, 2017.

[19] P. Fremantle, J. Kopecký, and B. Aziz, *Web API Management Meets the Internet of Things*, Springer International Publishing, Cham, Switzerland, 2015.

[20] S. Gerdes, C. Bormann, and O. Bergmann, "Chapter 11 - keeping users empowered in a cloudy internet of things," in *The Cloud Security Ecosystem*, R. Ko and K.-K. R. Choo, Eds., pp. 231–247, Syngress, Boston, MA, USA, 2015.

[21] J. L. H. Ramos, M. P. Pawlowski, A. J. Jara, A. F. Skarmeta, and L. Ladid, "Toward a lightweight authentication and authorization framework for smart objects," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 4, pp. 690–702, 2015.

[22] T. Kumar, A. Braeken, M. Liyanage, and M. Ylianttila, "Identity privacy preserving biometric based authentication scheme for Naked healthcare environment," in *Proceedings of the IEEE International Conference on Communications, ICC*, May 2017.

[23] S.-H. Lee, K.-W. Huang, and C.-S. Yang, "TBAS: Token-based authorization service architecture in Internet of things scenarios," *International Journal of Distributed Sensor Networks*, vol. 13, no. 7, 2017.

[24] L. Liu, B. Fang, and B. Yi, *A General Framework of Nonleakage-Based Authentication Using CSP for the Internet of Things*, Springer International Publishing, Cham, Switzerland, 2014.

[25] A. Pinto and R. Costa, *Hash-Chain Based Authentication for IoT Devices and REST Web-Services*, Springer International Publishing, Cham, Switzerland, 2016.

[26] M. Shahzad and M. P. Singh, "Continuous authentication and authorization for the internet of things," *IEEE Internet Computing*, vol. 21, no. 2, pp. 86–90, 2017.

[27] N. Shone, C. Dobbins, W. Hurst, and Q. Shi, "Digital memories based mobile user authentication for IoT," in *Proceedings of the IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, pp. 1796–1802, October 2015.

[28] Q. Tasali, C. Chowdhury, and E. Y. Vasserman, "A flexible authorization architecture for systems of interoperable medical devices," in *Proceedings of the 22nd ACM Symposium on Access Control Models and Technologies, SACMAT 2017*, pp. 9–20, USA, June 2017.

[29] M. Trnka and T. Cerny, "Authentication and authorization rules sharing for internet of things," *Software Networking*, no. 1, pp. 35–52, 2017.

[30] S. Unger and D. Timmermann, "DPWSec: devices profile for web services security," in *Proceedings of the 10th IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing, ISSNIP 2015*, pp. 1–6, April 2015.

[31] S. Wiseman, G. S. Mino, A. L. Cox, S. J. J. Gould, J. Moore, and C. Needham, "Use your words: Designing one-time pairing codes to improve user experience," in *Proceedings of the 34th Annual Conference on Human Factors in Computing Systems, CHI 2016*, USA, May 2016.

[32] S. Sicari, A. Rizzardi, L. Grieco, G. Piro, and A. Coen-Porisini, "A policy enforcement framework for Internet of Things applications in the smart health," *Smart Health*, vol. 3-4, pp. 39–74, 2017.

[33] A. Outchakoucht, H. Es-samaali, and J. Philippe, "Dynamic access control policy based on blockchain and machine learning

for the internet of things," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 1, 2017.

[34] N. Ye, Y. Zhu, R.-C. Wang, R. Malekian, and Q.-M. Lin, "An efficient authentication and access control scheme for perception layer of internet of things," *Applied Mathematics & Information Sciences*, vol. 8, no. 4, pp. 1617–1624, 2014.

[35] J. B. Bernabe, J. L. Hernandez-Ramos, and A. F. S. Gomez, "Holistic privacy-preserving identity management system for the internet of things," *Mobile Information Systems*, vol. 2017, 2017.

[36] J. Bernal Bernabe, J. L. Hernandez Ramos, and A. F. Skarmeta Gomez, "Taciot: multidimensional trust-aware access control system for the internet of things," *Soft Computing*, vol. 20, no. 5, pp. 1763–1779, 2016.

[37] S. Cirani and M. Picone, "Effective authorization for the Web of Things," in *Proceedings of the 2nd IEEE World Forum on Internet of Things, WF-IoT 2015*, pp. 316–320, December 2015.

[38] W. Han, Y. Zhang, Z. Guo, and E. Bertino, "Fine-grained business data confidentiality control in cross-organizational tracking," in *Proceedings of the 20th ACM Symposium on Access Control Models and Technologies, SACMAT 2015*, pp. 135–145, June 2015.

[39] A. Kurniawan and M. Kyas, "A trust model-based Bayesian decision theory in large scale Internet of Things," in *Proceedings of the 10th IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing, ISSNIP 2015*, April 2015.

[40] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, "Towards a novel privacy-preserving access control model based on blockchain technology in IoT," *Advances in Intelligent Systems and Computing*, vol. 520, pp. 523–533, 2017.

[41] P. Solapurkar, "Building secure healthcare services using OAuth 2.0 and JSON web token in IOT cloud scenario," in *Proceedings of the 2nd International Conference on Contemporary Computing and Informatics, IC3I 2016*, pp. 99–104, December 2016.

[42] S. Lee, J. Choi, J. Kim et al., "FACT: Functionality-centric access control system for IoT programming frameworks," in *Proceedings of the 22nd ACM Symposium on Access Control Models and Technologies, SACMAT 2017*, pp. 43–54, USA, June 2017.

[43] S. Bandara, T. Yashiro, N. Koshizuka, and K. Sakamura, "Access control framework for API-enabled devices in smart buildings," in *Proceedings of the 22nd Asia-Pacific Conference on Communications, APCC 2016*, pp. 210–217, August 2016.

[44] A. Biason, C. Pielli, A. Zanella, and M. Zorzi, "Access control for IoT nodes with energy and fidelity constraints," *IEEE Transactions on Wireless Communications*, vol. 17, no. 5, pp. 3242–3257, 2018.

[45] Q. Huang, L. Wang, and Y. Yang, "DECENT: Secure and fine-grained data access control with policy updating for constrained IoT devices," *World Wide Web*, pp. 1–17, 2017.

[46] S. Gusmeroli, S. Piccione, and D. Rotondi, *A Capability-Based Security Approach to Manage Access Control in the Internet of Things*, vol. 58 of *Mathematical and Computer Modelling*, 5 edition, 2013, The Measurement of Undesirable Outputs: Models Development and Empirical Analyses and Advances in mobile, ubiquitous and cognitive computing.

[47] A. Majeed and A. Al-Yasiri, *Formulating A Global Identifier Based on Actor Relationship for the Internet of Things*, Springer International Publishing, Cham, Switzerland, 2017.

[48] D. Schreckling, J. David Parra, D. Charalampos, and J. Posegga, *Data-Centric Security for the IoT*, Springer International Publishing, Cham, Switzerland, 2016.

[49] K. Fysarakis, I. Papaefstathiou, C. Manifavas, K. Rantos, and O. Sultatos, "Policy-based access control for DPWS-enabled ubiquitous devices," in *Proceedings of the 19th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2014*, September 2014.

[50] D. Hussein, E. Bertin, and V. Frey, "A community-driven access control approach in Distributed IoT environments," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 146–153, 2017.

[51] V. Sivaraman, H. H. Gharakheili, A. Vishwanath, R. Boreli, and O. Mehani, "Network-level security and privacy control for smart-home IoT devices," in *Proceedings of the 11th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob 2015*, pp. 163–167, October 2015.

[52] I. Bouij-Pasquier, A. Ait Ouahman, A. Abou El Kalam, and M. Ouabiba De Montfort, "SmartOrBAC security and privacy in the Internet of Things," in *Proceedings of the 12th IEEE/ACS International Conference of Computer Systems and Applications, AICCSA 2015*, November 2015.

[53] M. Poulymenopoulou, F. Malamateniou, and G. Vassilacopoulos, "A virtual PHR authorization system," in *Proceedings of the 2014 IEEE-EMBS International Conference on Biomedical and Health Informatics, BHI 2014*, pp. 73–76, June 2014.

[54] J. Wilson, R. S. Wahby, H. Corrigan-Gibbs, D. Boneh, P. Levis, and K. Winstein, "Trust but verify: Auditing the secure Internet of Things," in *Proceedings of the 15th ACM International Conference on Mobile Systems, Applications, and Services, MobiSys*, USA, 2017.

[55] C.-Y. Chen, *Efficient Authentication for Tiered Internet of Things Networks*, Springer International Publishing, Cham, Switzerland, 2017.

[56] H. Ren, Y. Song, S. Yang, and F. Situ, "Secure smart home: A voiceprint and internet based authentication system for remote accessing," in *Proceedings of the 11th International Conference on Computer Science and Education, ICCSE 2016*, pp. 247–251, August 2016.

[57] R. Sandhu, "Access control: The neglected frontier," in *Information Security and Privacy*, J. Pieprzyk and J. Seberry, Eds., vol. 1172, Springer, Berlin, Heidelberg, Germany, 1996.

[58] D. Ferraiolo and R. Kuhn, "Role-based access control," in *Proceedings of the 15th National Computer Security Conference*, pp. 554–556, April 1992.

[59] D. Gregory, K. Abowd Anind, J. Peter, N. Davies, M. Smith, and P. Steggles, *Towards a Better Understanding of Context and Context-Awareness*, Springer, Berlin, Heidelberg, Germany, 1999.

[60] M. Trnka and T. Cerny, "On security level usage in context-aware role-based access control," in *Proceedings of the the 31st Annual ACM Symposium*, NY, USA, April 2016.

[61] M. A. Al-Kahtani and R. Sandhu, "A model for attribute-based user-role assignment," in *Proceedings of the 18th Annual Computer Security Applications Conference, ACSAC 2002*, pp. 353–362, USA, December 2002.

[62] M. Ge and S. L. Osborn, "A design for parameterized roles," in *Research Directions in Data and Applications Security XVIII*, C. Farkas and P. Samarati, Eds., Springer, Boston, MA, USA, 2004.

[63] J. Fischer, D. Marino, R. Majumdar, and T. Millstein, "Fine-grained access control with object-sensitive roles," in *ECOOP 2009 – Object-Oriented Programming*, S. Drossopoulou, Ed., Springer, Berlin, Heidelberg, Germany, 2009.

[64] M. J. Covington, W. Long, S. Srinivasan, A. K. Dey, M. Ahamad, and G. D. Abowd, "Securing context-aware applications using

environment roles," in *Proceedings of the sixth ACM Symposium on Access Control Models and Technologies (SACMAT 2001)*, pp. 10–20, USA, May 2001.

[65] G. Sladić, B. Milosavljević, and Z. Konjović, "Context-sensitive access control model for business processes," *Journal of Organizational Computing and Electronic Commerce*, vol. 10, no. 6, pp. 939–972, 2013.

[66] X. Jin, R. Krishnan, and R. Sandhu, "A unified attribute-based access control model covering dac, mac and rbac," in *Data and Applications Security and Privacy XXVI*, N. Cuppens-Boulahia, F. Cuppens, and J. Garcia-Alfaro, Eds., pp. 41–55, Springer, Berlin, Heidelberg, Germany, 2012.

[67] K. Zeilenga, "Lightweight directory access protocol (ldap): technical specification road map," RFC 4510, RFC Editor, 2006, http://www.rfc-editor.org/rfc/rfc4510.txt.

[68] M. Jones, B. de Medeiros, C. Mortimore, N. Sakimura, and J. Bradley, "Openid connect core," OpenID Community, 2014.

[69] S. Rose, D. Engel, N. Cramer, and W. Cowley, "Automatic keyword extraction from individual documents," *Text Mining: Applications and Theory*, pp. 1–20, 2010.

[70] N. Pohlmann, M. Hertlein, and P. Manaras, *Bring Your Own Device For Authentication (BYOD4A) – The Xign–System*, Springer Fachmedien Wiesbaden, Germany, 2015.

[71] A. Bassi, M. Bauer, M. Fiedler et al., *Enabling Things to Talk: Designing IoT Solutions with the IoT Architectural Reference Model*, Springer Publishing Company, Incorporated, 1st edition, 2016.

[72] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz, "Extensible Authentication Protocol (eap)," RFC 3748, RFC Editor, June 2004, http://www.rfc-editor.org/rfc/rfc3748.txt.

[73] R. T. Fielding, *Architectural Styles and the Design of Network-Based Software Architectures*, chapter Representational State Transfer (REST), University of Californi, 2000.

[74] E. Rescorla, "Http over tls," RFC 2818, RFC Editor, May 2000, http://www.rfc-editor.org/rfc/rfc2818.txt.

[75] A. W. Roscoe, *The Theory and Practice of Concurrency*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1997.

[76] J. Camenisch and E. V. Herreweghen, "Design and implementation of the idemix anonymous credential system," in *Proceedings of the 9th ACM Conference on Computer and Communications Security CCS '02*, pp. 21–30, ACM, New York, NY, USA, November 2002.

[77] S. Cirani, M. Picone, P. Gonizzi, L. Veltri, and G. Ferrari, "IoT-OAS: an oauth-based authorization service architecture for secure services in IoT scenarios," *IEEE Sensors Journal*, vol. 15, no. 2, pp. 1224–1234, 2015.

[78] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the IEEE Symposium on Security and Privacy (SP '07)*, pp. 321–334, IEEE Computer Society, Washington, DC, USA, 2007.

[79] "The extensible access control markup language (xacml) version 3.0. OASIS Standard," 2017.

[80] D. Hardt, "The oauth 2.0 authorization framework," RFC 6749, RFC Editor, October 2012, http://www.rfc-editor.org/rfc/rfc6749.txt.

[81] M. Jones, J. Bradley, and N. Sakimura, "Json web token (jwt)," RFC 7519, RFC Editor, May 2015, http://www.rfc-editor.org/rfc/rfc7519.txt.

[82] A. A. E. Kalam, R. E. Baida, P. Balbiani et al., "Organization based access control," in *Proceedings of the 4th IEEE International Workshop on Policies for Distributed Systems and Networks*, pp. 120–131, 2003.

[83] Astm f2761-09, *Medical Devices And Medical Systems - Essential Safety Requirements for Equipment Comprising The Patient-Centric Integrated Clinical Environment (Ice) - Part 1: General Requirements And Conceptual Model*, ASTM International, PA, USA, 2009.

[84] J. Hatcliff, A. King, I. Lee et al., "Rationale and architecture principles for medical application platforms," in *Proceedings of the 2012 IEEE/ACM 3rd International Conference on Cyber-Physical Systems, ICCPS 2012*, pp. 3–12, April 2012.

[85] T. Cerny, M. J. Donahoo, and M. Trnka, "Contextual understanding of microservice architecture: Current and future directions," *Applied Computing Review*, vol. 17, no. 4, 2018.

[86] Web services security: Soap message security 1.1. OASIS Standard, 2006.

[87] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: a survey," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 414–454, 2014.

[88] B. C. Neuman, "Proxy-based authorization and accounting for distributed systems," in *Proceedings of the 1993 IEEE 13th International Conference on Distributed Computing Systems*, pp. 283–291, May 1993.

[89] "Google scholar," https://scholar.google.com/.

[90] E. Garfield, "The history and meaning of the journal impact factor," *Journal of the American Medical Association*, vol. 295, no. 1, pp. 90–93, 2006.

[91] "Core conference ranking," http://portal.core.edu.au/conf-ranks/.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 7254305, 7 pages
https://doi.org/10.1155/2018/7254305

Research Article

A Secure Ciphertext Retrieval Scheme against Insider KGAs for Mobile Devices in Cloud Storage

Run Xie,[1] Chanlian He,[2] Dongqing Xie,[3] Chongzhi Gao ,[3] and Xiaojun Zhang[4]

[1]*School of Mathematics, Yibin University, Yibin, China*
[2]*School of Computer and Information Engineering, Yibin University, Yibin, China*
[3]*School of Computer Science, Guangzhou University, Guangzhou, China*
[4]*School of Computer Science, Southwest Petroleum University, Chengdu, China*

Correspondence should be addressed to Chongzhi Gao; czgao@gzhu.edu.cn

Received 13 February 2018; Revised 24 April 2018; Accepted 10 May 2018; Published 6 June 2018

Academic Editor: Ilsun You

With the advent of cloud computing, data privacy has become one of critical security issues and attracted much attention as more and more mobile devices are relying on the services in cloud. To protect data privacy, users usually encrypt their sensitive data before uploading to cloud servers, which renders the data utilization to be difficult. The ciphertext retrieval is able to realize utilization over encrypted data and searchable public key encryption is an effective way in the construction of encrypted data retrieval. However, the previous related works have not paid much attention to the design of ciphertext retrieval schemes that are secure against inside keyword-guessing attacks (KGAs). In this paper, we first construct a new architecture to resist inside KGAs. Moreover we present an efficient ciphertext retrieval instance with a designated tester (dCRKS) based on the architecture. This instance is secure under the inside KGAs. Finally, security analysis and efficiency comparison show that the proposal is effective for the retrieval of encrypted data in cloud computing.

1. Introduction

With the development and deployment of cloud computing, more and more mobile devices are connected to the cloud and receiving services provided by the cloud servers. Cloud storage service is one of the most critical applications of cloud computing, which offers great convenience to users, including the storage of data and sharing of data [1]. However, the cloud service providers are usually cannot be completely trusted because they are managed and controlled by a third party such as Google or Amazon. Thus, the users have to take the risk that the cloud servers may find and leak their sensitive information.

To tackle the challenge of security in cloud storage, a lot of research has been performed to address the security and privacy issues in cloud computing, such as the cloud data access control [2–4], cloud data outsourcing computation [5–8], and privacy in data processing [9–13].

To protect data privacy, data are usually encrypted before they are uploaded to the cloud server. Nevertheless, this way generates the new obstacles that the cloud server is not able to carry out data retrieval over ciphertext data [14]. When users would like to access the part of encrypted data, they have to get entire data back or share the keys with the cloud server. As a result, users have to pay more for the bandwidth or give up their data privacy.

To overcome this obstacle, the concept of searchable encryption [15] was first proposed by Song et al. at 2000. Meanwhile, based on symmetric encryption, they proposed the keyword search scheme based on symmetric encryption, namely, Searchable Symmetric Encryption (abbreviation SSE). The searchable encryption permits users to retrieve a particular keyword over encrypted data by sending a trapdoor to the cloud server. However, the SSE involves detailed secret key management.

To overcome weakness and improve security, a new searchable encryption primitive was presented by Boneh and Boyen, which is called Public key Encryption with Keyword Search (PEKS in short) [16]. In their solution, with public information, the sender can encrypt the keyword associated

with encrypted data and store its ciphertext to cloud server. To achieve keyword retrieval over the encrypted data, the receiver creates a trapdoor corresponding to the keyword, then he delivers the trapdoor to server. By the testing procedure, the server can find the ciphertext of keyword associated with the trapdoor. Then it sends corresponding data to receiver. Yet, there is the requirement of secure channel in their PEKS [16]. Usually, it is difficult to fulfill this requirement. This weakness limits the applications of their scheme. Following Boneh's PEKS, Baek et al. presented a new PEKS solution, which is called Secure Channel Free Public Key Encryption with Keyword Search [17] (denoted as SCF-PEKS). However, Baek's scheme was proved to be insecure by Yau et al. [18] with the following reasons. When the outside adversary acquires a trapdoor in channel, he can launch keyword-guessing attacks. This attack is called outside KGAs. So far, many of existing solutions concentrate on building up the security to resist outside KGAs [19–23]. Only a few schemes [24–28] are secure against this attacks.

Additionally, the most difficult issue is to resist keyword-guessing attacks launched by the cloud server, namely, inside KGAs. The inside KGAs are that the test server launched keyword-guessing attacks. Actually, such kind of attack also has been considered in deduplication system [30] and other security protocols [3, 31–33]. Specifically, the malicious server can create their ciphertext since the production of a PEKS ciphertext of keyword involves only public parameters. Given the trapdoor, the malicious cloud server can perform the test procedure with the guessing keyword. Therefore the server is able to know if the guessing keyword matches given trapdoor. Repeating guessing-then-testing process, the server can find the correct keyword. Because of the weakness of the small size of keyword space, this attack is available. Based on dual-server, Chen et al. [34] presented a new PEKS scheme which is considered to be secure against insider KGAs. However, in their solution, the front server can test whether the ciphertext of keyword relates to given trapdoor. Hence, under the original framework of [16], designing a secure scheme against inside KGAs is out of the question.

Recently, a new scheme [29] has been proposed by Jiang et al. based on slightly different architecture. With the aid of TTP (trusted third party), their solution can resist inside KGAs. In [29], TTP delivers its secret key to the sender from a safe channel. With his own secret key, the sender produces the legal ciphertext of keywords. Without the sender's secret key, the server is not able to generate a correct ciphertext of keyword as inputs of the test procedure. Hence, the server is not able to launch inside KGAs.

1.1. Our Contributions. In this paper, we present a new ciphertext retrieval system with a designated tester (dCRKS) based on the new security model. Compared with some analogous works, such as the cloud data retrieval schemes [35–37], the advantages of this system can be summarized as follows.

Firstly, we build security model of ciphertext retrieval system. Here, the server will not be considered as special attacker. So this model is more simple.

Second, we design an instance of dCRKS. This dCRKS instance can resist inside KGAs. In the instance, the server can not produce a correct ciphertext of keywords without the secret key of sender. Meanwhile the server can not generate a valid trapdoor without the secret key of receiver. Therefore, the malicious server is not able to launch inside KGAs. Most of the existing literatures (as [25, 28]) can not resist inside KGAs. Although the [29] is secure against inside KGAs, the TTP (trusted third party) is required in their scheme.

Thirdly, in this dCRKS instance, only a specified server is able to test whether given trapdoor relates to a dCRKS ciphertext. So, the proposal is stronger than [29].

Last, the analysis proves that the generation method of trapdoor and the testing algorithm are more effective than those of [29].

2. Preliminaries

Here, we will build the framework of dCRKS and its security model. Next, we introduce the hard assumptions which are used to prove the security of the instance of dCRKS system. In security model, let $\mathcal{F}$ be an adversary. The challenger denoted by $\mathcal{G}$. The dCRKS ciphertexts refer to the list of encrypted keywords.

2.1. Framework of dCRKS and Security Model

2.1.1. Framework of dCRKS. The dCRKS system is a ciphertext retrieval approach. In this system, only the specified server can carry out the testing procedure with the correct dCRKS ciphertext. The framework of dCRKS consists of the four algorithms. They are defined as follows.

Setup. Here n is the essential parameter. Let $\mathcal{PP}$ be the set public parameter. Inputting n, this algorithm outputs $\mathcal{PP}$.

KeyGen ($\mathcal{PP}$). When the $\mathcal{PP}$ are imported, this algorithm outputs (P_d, K_d), (P_r, K_r), and (P_s, K_s). The P_d (or K_d) is the sender's public (or private) key. Similarly, the receiver's public (private) is the P_r (K_r). The server's public (private) is the P_s (K_s).

EndCRKS ($\mathcal{PP}, K_d, P_d, P_r$). Let w be a keyword. Taking P_d, K_d, P_r, w, and the public parameter $\mathcal{PP}$ as input, this EndCRKS produces C_w corresponding to w, where C_w is dCRKS ciphertext.

dTrapdoor ($P_d, K_r, P_s, \mathcal{PP}$). When the keywords w', K_r, P_s, and $\mathcal{PP}$ are imported, this algorithm produces a trapdoor $T_{w'}$ corresponding to w'.

dTest ($K_s, C_w, T_{w'}, \mathcal{PP}$). In this algorithm, the server takes a trapdoor $T_{w'}$, $\mathcal{PP}$, a dCRKS ciphertext C_w, and its private key K_s as input. If $w' = w$, it replies "yes"; otherwise it replies "no".

2.1.2. Security Model. Here, we construct the security architecture of the dCRKS. The security of dCRKS ciphertext and trapdoor are defined by this architecture. The security of dCRKS is based on the two games.

In game 1, the adversary can be a malicious server or a malicious receiver or other attackers. So, the adversary can know the server's secret key or the receiver's secret key. The only limitation is that the adversary can not query the trapdoors corresponding to the challenge keywords w_0, w_1. The dCRKS ciphertext is secure. That means the adversary $\mathcal{F}$ is unable to differentiate between the dCRKS ciphertext of w_1 and the dCRKS ciphertext of keyword w_0 when the coupling trapdoor has not be obtained.

In game 2, the adversary can be a malicious server or a malicious sender or other attackers. Clearly, he can obtain the server's secret key or the receiver's secret key. The security of trapdoor requires that the $\mathcal{F}$ is unable to differentiate between a trapdoor of w_1 and a trapdoor of w_0 when the coupling trapdoor has not be obtained, where w_0 and w_1 are the challenge keywords.

The game 1 is described as follows.

Game 1. In this game, the $\mathcal{F}$ can enquire for private key and trapdoor. Yet $\mathcal{F}$ is not permitted to enquire the coupling trapdoors of the challenge keywords w_0, w_1, where both w_0 and w_1 are his choice. It requires that $\mathcal{F}$ differentiates between the dCRKS ciphertext of keyword w_1 and the dCRKS ciphertext of w_0. If the $\mathcal{F}$ can not win this game with nonnegligible probability, the dCRKS scheme is secure to resist the chosen keyword attacks.

Init. In this phase, $\mathcal{F}$ issues P_d as the challenge public key P_d.

Setup. Running the setup procedure, $\mathcal{G}$ generates the public parameters $\mathcal{PP}$ and gives the public parameters $\mathcal{PP}$ to adversary $\mathcal{F}$.

Phase 1. $\mathcal{F}$ performs repeatedly inquiries. The restriction is that the number of enquiries is no more than polynomially bounded.

Pk-Query (Private Key Query). For $d_i \neq d^*$, $\mathcal{F}$ sends P_{d_i} to $\mathcal{G}$, then $\mathcal{G}$ replies $\mathcal{F}$ with K_{d_i}.

T-Query (Trapdoor Query). $\mathcal{F}$ issues P_{r_i} and w_i to $\mathcal{G}$. $\mathcal{G}$ runs the trapdoor procedure and replies the trapdoor T_{w_i} for P_{r_i}, w_i to $\mathcal{F}$.

Challenge. $\mathcal{F}$ chooses the pair keywords (w_0, w_1) and P_{d^*} as the challenge keywords for P_{d^*}. The restriction is that the private key corresponding to P_{d^*} or the trapdoors corresponding to w_0 and w_1 have not been enquired by $\mathcal{F}$. $\mathcal{G}$ generates the challenge ciphertext $C^*_{w_b}$ and replies $C^*_{w_b}$ to $\mathcal{F}$, where $b \in \{0, 1\}$ is a random bit.

Phase 2. In this phase, $\mathcal{F}$ can still enquire the secret keys ($d \neq d^*$) and the trapdoors ($d \neq d^*$) or the trapdoor for $d = d^*$ with $w_i \neq w_0, w_1$. $\mathcal{G}$ replies as Phase 1.

Outputs. In the end, $\mathcal{F}$ guesses $b \in \{0, 1\}$. When $b' = b$, it means that $\mathcal{F}$ wins this game.

Game 2. Here, $\mathcal{F}$ can enquire for dCRKS ciphertext and secret key. Yet, $\mathcal{F}$ is not allowed to enquire the dCRKS ciphertexts corresponding to the challenge keywords w_0 and w_1, where both w_0 and w_1 are his choice. It requires that $\mathcal{F}$ differentiates between the trapdoor of keyword w_1 and the trapdoor of w_0. If the $\mathcal{F}$ can not win this game with nonnegligible advantage probability, the dCRKS system can resist the chosen keyword attacks.

Init. $\mathcal{F}$ issues the challenge public key P_r.

Setup. Running setup procedure, the $\mathcal{G}$ produces public parameters $\mathcal{PP}$ and delivers the parameters $\mathcal{PP}$ to $\mathcal{F}$.

Phase 1. $\mathcal{F}$ performs repeatedly inquiries. The restriction is that the number of inquiries is no more than polynomially bounded.

Pk-Query. $\mathcal{F}$ sends P_{r_i} to $\mathcal{G}$, $r_i \neq r^*$, and $\mathcal{G}$ responds K_{r_i} to $\mathcal{F}$ with r_i.

dc-Query (dCRKS Ciphertext Query). $\mathcal{F}$ sends P_{d_i} and w_i to $\mathcal{G}$. Running the EndCRKS procedure, $\mathcal{G}$ replies the ciphertext C_{w_i} for P_{d_i}, w_i to $\mathcal{F}$.

Challenge. $\mathcal{F}$ chooses a pair keywords w_0, w_1, and P_{r^*} as the challenge keywords for P_{r^*}. The restriction is that the secret key for P_{r^*} or the dCRKS ciphertext of w_0, w_1 for P_{r^*} has not been inquired by $\mathcal{F}$. $\mathcal{G}$ generates $T^*_{w_b}$ as challenge trapdoor and replies $T^*_{w_b}$ to $\mathcal{F}$, where $b \in \{0, 1\}$ is a random bit.

Phase 2. In this phase, $\mathcal{F}$ can still enquire the dCRKS ciphertexts and the secret key for $r \neq r^*$ or the dCRKS ciphertexts for $r = r^*$ and $w_i \neq w_0, w_1$. $\mathcal{G}$ replies as the first phase.

Outputs. In the end, $\mathcal{F}$ guesses $b' \in \{0, 1\}$. When $b' = b$, it means that $\mathcal{F}$ wins this game.

2.2. Complexity Assumptions

2.2.1. Bilinear Map. Let G and G_T be multiplicative cyclic groups with the order p (prime). Let $e : G \times G \to G_T$ be a bilinear map. e has the following properties:

(1) $g_1, g_2 \in G$ exist, such that $e(g_1, g_2)$ is not equal to 1_{G_T}.

(2) For all $\eta, \pi \in G$ and $a, t \in Z$, the $e(\eta^a, \pi^t) = e(\eta, \pi)^{at}$ is true.

(3) For all $\eta, \pi \in G$, the $e(\eta, \pi)$ can be calculated in polynomial time.

2.2.2. Complexity Assumptions. According to [38], the Computational Diffie-Hellman (CDH) problem is considered to be hard on G and G_T. Meanwhile, we know that the Decision Diffie-Hellman (DDH) problem [39] is hard on G_T.

In additional, to prove the security of dCRKS system, we need to introduce a new hard problem on G and G_T, namely, the Strong Decisional Diffie-Hellman assumption (in short SDDH). The SDDH problem is defined as follows.

The SDDH problem in (G, G_T) is as follows.

Given $\mathcal{L}' = (g, h, g^a, h^c, e(g, h)^{1/a}, e(g, g)^t, e(g, h)^q)$ as input, output "yes" if $q = ct/a$ and "no" otherwise, where a, c, t, $q \in Z^*_p$.

As is known, g^t is not able to infer from $e(g,g)^t$ because the e is considered to be one-way functions. Moreover, $e(g,h)^{ct/a}$ can not be calculated from g^a, h, and h^c. In fact, even the DDH problem is easy, the SDDH problem is seemingly still intractable.

Additionally, the DLP (discrete logarithm problem) is assumed to hold over G and G_T.

3. dCRKS against Insider Attacks

3.1. The Instance of dCRKS. Now, we will describe the instance of dCRKS. Here, the G and G_T are given groups as the previous definition. Let $H : \{0,1\}^* \to Z_p^*$ be the hash function, which is considered as random oracle in security model.

Setup. Let e be a bilinear map on G. Let p be the order of G and G_T, where both G and G_T are multiplicative cyclic group. This procedure produces $\mathcal{PP} = (g, G_T, G, p, H, e, h)$, where $\mathcal{PP}$ is the set of public parameters. The generator of G is g.

KeyGen ($\mathcal{PP}$). Takes as input x, y, z, g, and h, where x, y, $z \in Z_p^*$. This procedure generates key as the following way.

$K_d = z$ and $P_d = h^z$, where K_d and P_d are the sender's private key and public key, respectively.

$K_r = x$ and $P_r = (P_{r1}, P_{r2}) = (e(g,h)^{1/x}, g^x)$, where $K_r = x$ and P_r are the receiver's private key and public key.

$K_s = y$ and $P_s = g^y$, where $K_s = y$ and $P_s = g^y$ are the server's private key and public key.

EndCRKS (P_d, K_d, P_r, w). The sender takes a keyword w, P_r, P_d, K_d, $\mathcal{PP}$, and a random $u \in Z_p^*$ as input. This procedure produces $C = (C_1, C_2, C_3)$ as the dCRKS ciphertext of w output. C_1, C_2, and C_3 are calculated as follows:

$$C_3 = e(g,g)^u$$
$$C_2 = (P_{r2})^u \cdot g^{-uH(w)} = g^{(x-H(w))u}$$
$$C_1 = (P_{r1})^{zu} = e(g,h)^{zu/x}$$

dTrapdoor (P_d, K_r, P_s, w'). The receiver takes a keyword w', $r \in Z_p^*$, P_d, K_r, and P_s as inputs. This procedure produces $T = (T_1, T_2)$ as the trapdoor of w' output. T_1 and T_2 are calculated as follows:

$$T_1 = r;$$
$$T_2 = (P_d^{1/x} \cdot P_s^{-r})^{1/(x-H(w'))}.$$

dTest. Receiving a trapdoor T, the server runs dTest algorithm over the dCRKS ciphertexts with his private key y. Let C be a dCRKS ciphertext. The retrieving operation is executed by checking

$$e(C_2, T_2)\, C_3^{yT_1} = C_1 \quad (1)$$

If the above equation is true, the algorithm returns 1; otherwise it returns 0.

3.2. Correctness of dCRKS. Now, we show that the above instance is correct. Let C be a dCRKS ciphertext which matches the trapdoor T. By the following equations, we can verify the correctness of dTest.

$$T_1 = r;$$
$$T_2 = (P_d^{1/x} \cdot P_s^{-r})^{1/(x-H(w'))};$$
$$T = (T_1, T_2);$$
$$C_2 = P_{r2}^u \cdot g^{-uH(w)};$$
$$e(C_2, T_2) = e(g^{(x-H(w))u}, (h^{z/x} g^{-yr})^{1/(x-H(w'))}).$$

When $w' = w$, we can obtain the equations

$$e(C_2, T_2) = e(g^u, h^{z/x} g^{-yr}) = e(g^u, h^{z/x}) e(g^u, g^{-yr})$$

As a result, the correctness of dTest is verified as follows:

$$C_1 = e(g,h)^{zu/x} = e(C_2, T_2) C_3^{yT_1}.$$

3.3. Security of dCRKS

3.3.1. Security of dCRKS Ciphertext. In this section, we demonstrate that the ciphertexts of keywords are secure under the chosen keyword attack in the instance.

Theorem 1. *Suppose the SDDH problem is hard; the dCRKS instance can achieve dCRKS ciphertext indistinguishability.*

Proof. Let $\mathcal{F}$ be polynomial-time adversary. If $\mathcal{F}$ can break the dCRKS instance with nonnegligible advantage probability, we construct an algorithm $\mathcal{G}$ as the challenger, who can solve the SDDH problem with nonnegligible advantage probability.

Init. The $\mathcal{F}$ issues the challenge public key $P_{d^*} = h^{z^*}$ and a keyword set w_i.

Setup. Let $\mathcal{L} = (g, h, g^a, h^c, e(g,h)^{1/a}, e(g,g)^t, e(g,h)^q)$ be a SDDH instance. $\mathcal{G}$ is given $\mathcal{L}$ and (G, G_T, $e(\cdot)$). The setup procedure produces parameters $\mathcal{PP}$, then $\mathcal{G}$ sends $\mathcal{PP}$ to $\mathcal{F}$.

Phase 1. $\mathcal{F}$ can carry out multiple queries. The restriction is that the number of enquiries is no more than polynomially bounded.

Pk-Query. To inquire d_i's private key, $\mathcal{F}$ transmits P_{d_i} to $\mathcal{G}$. If $d_i \neq d^*$, $\mathcal{G}$ returns $z_i = K_{d_i}$ to $\mathcal{F}$.

T-Query. To inquire the trapdoor of w_i, $\mathcal{F}$ issues w_i and $P_{r_i} = (e(g,h)^{1/x}, g^x)$ to $\mathcal{G}$. $\mathcal{G}$ responds the trapdoor T_{w_i} for P_{r_i}, w_i by calling the trapdoor oracle.

Challenge. Let d^* be the sender's identity. $\mathcal{F}$ selects w_0, w_1, and P_{d^*} as challenge. The restriction is that the secret key for P_{d^*} or the trapdoor for w_0, w_1 for P_{d^*} have not been inquired by $\mathcal{F}$. $\mathcal{G}$ replies the challenge ciphertext $C^*_{w_b}$ to $\mathcal{F}$, where $C^*_{w_b}$ is a tuple (C_1, C_2, C_3) and $C_3 = e(g,g)^t$, $C_2 = g^{at'} g^{-t'H(w_b)}$, and $C_1 = e(g,h)^q$.

Phase 2. $\mathcal{F}$ can still enquire the trapdoor and the secret key for $d \neq d^*$ or the trapdoor for $d = d^*$ and $w_i \neq w_0, w_1$. $\mathcal{G}$ replies as the front phase.

Outputs. In the end, $\mathcal{F}$ outputs $b' \in \{0, 1\}$.

Analysis. Because the actual dCRKS ciphertext C_{w_b} is $C_1 = e(g,h)^{zu/x}$, $C_2 = g^{x-H(w_b)u}$, and $C_3 = e(g,g)^u$, the distribution of challenge ciphertext $C^*_{w_b}$ is identical to that in the actual system. In fact, for the uniformly random $t', t \in Z_p^*$, $\mathcal{F}$ needs to differentiate between the tuple $(g^{t'(-H(w_b)+a)}, e(g,g)^t)$ and the tuple $(g^{t(-H(w_b)+a)}, e(g,g)^t)$. If $q = ct/a$, the simulation is perfect. ε denotes nonnegligible probability. As a result, if $\mathcal{F}$ has advantage probability $1/2 + \varepsilon$ to determine the bits b correctly, then the $\mathcal{G}$ can solve the SDDH problem with identical advantage probability ε.

This completes the proof of dCRKS ciphertexts indistinguishability. □

3.3.2. Security of Trapdoor. Now, we show that the trapdoor is secure under the chosen keyword attack.

Theorem 2. *The dCRKS instance can achieve the trapdoor indistinguishability to resist the chosen keyword attack under random oracle model in game 2.*

Proof. In this section, we show that the polynomial-time algorithm $\mathcal{F}$ is able to differentiate between the ciphertext of keyword w_0 and the ciphertext of keyword w_1 if and only if he can distinguish two uniform distributions on G.

Init. $\mathcal{F}$ issues $P_{r^*} = (g^{x^*}, e(g,h)^{1/x^*})$ as the challenge public key.

Setup. Running the setup procedure, $\mathcal{G}$ gives the public parameters to $\mathcal{F}$.

Phase 1. $\mathcal{F}$ implements multiple queries without exceeding polynomial bounded.

Pk-Query. To inquire r_i's private key, $\mathcal{F}$ sends $P_{r_i} = (e(g,h)^{1/x_i})$ to $\mathcal{G}$, $r_i \neq r^*$. Then $\mathcal{G}$ returns $K_{r_i} = x_i$.

dc-Query. $\mathcal{F}$ issues P_{d_i} and w_i to $\mathcal{G}$. Running EndCRKS oracle, $\mathcal{G}$ returns C_{w_i} for P_{d_i}, w_i to $\mathcal{F}$.

Challenge. $\mathcal{F}$ chooses keywords w_0, w_1, and P_{r^*} as his challenge. The restriction is that the ciphertext for w_0, w_1 for P_{r^*} or the private key for P_{r^*} has not be enquired by $\mathcal{F}$. $\mathcal{G}$ picks two random r', z' and computes the challenge trapdoor $T^*_{w_b}$, where $T^*_{w'_b} = (r', h^{z'} g^{-yr'})$. Then $\mathcal{G}$ replies the challenge trapdoor $T^*_{w'_b}$ to $\mathcal{F}$.

Phase 2. $\mathcal{F}$ can still enquire the ciphertext and the secret key for $r \neq r^*$ or the ciphertext for $r = r^*$ and $w_i \neq w_0, w_1$. $\mathcal{G}$ replies as first phase.

Outputs. $\mathcal{F}$ outputs $b' \in \{0, 1\}$.

Analysis. By enquiring, $\mathcal{F}$ can obtain g^{x^*}, $e(g,h)^{1/x^*}$, and $T^*_{w'_b} = (r', h^{z'} g^{-yr'})$. In scheme, $T^*_{w_b} = (r, (h^{z/x^*} g^{-yr})^{1/(x^*-H(w_b))})$, where r is uniform random value. Thus the distribution of $(h^{z/x^*} g^{-yr})^{1/(x^*-H(w_b))}$ is a uniform distribution on G. Meanwhile, the $T^*_{w'_b} = (r', h^{z'} g^{-yr'})$ is uniform distribution on G with taking uniform random r', z'. As a result, the simulation is perfect, namely, the distribution of $T^*_{w'_b}$ is identical to that in the actual system.

Moreover, as game 2, $\mathcal{F}$ can not know z and x^*. Even if $\mathcal{F}$ can calculate the following value:

$$e(g^{(x^*-H(w_b))}, T^*_{w_b}) = e(g,h)^{(z/x^*-yr)((x^*-H(w_{b'}))/(x^*-H(w_b)))}$$

$\mathcal{F}$ cannot distinguish

$$e(g,h)^{(z/x^*-yr)((x^*-H(w_{b'}))/(x^*-H(w_b)))} \text{ (where } b' \neq b)$$

from $e(g,h)^{z/x^*} \cdot e(g,h)^{-yr}$ (where $b' = b$).

Therefore, $\mathcal{F}$ can guess $b' = b$ with nonnegligible advantage probability, then $\mathcal{G}$ can distinguish between $(h^{z/x^*} g^{-yr})^{1/(x^*-H(w_b))}$ and uniformly distribution on G with identical advantage probability. □

3.3.3. Analysis of against Inside KGAs. In this section, we show that the dCRKS instance is secure against inside KGAs as follows.

First, given the trapdoor T^d_w, the server can generate the legal ciphertext C corresponding to T^d_w if and only if it can obtain the specified senders private key. Maybe the server can select z' and calculate $h^{z'}$ to produce ciphertext $C = (C'_1, C'_2, C'_3)$ corresponding to w', where

$$e(g, h^{z'})^{u/x} = (P_{r1})^{z'u} = C'_1$$
$$g^{(x-H(w'))u} = (P_{r2})^u \cdot g^{-uH(w')} = C'_2$$
$$e(g,g)^u = C'_3$$

Let $T^d_w = (T_1, T_2)$, then

$$e(C_2, T_2) = e(g^{(x-H(w))u}, (h^{z/x} g^{-yr})^{1/(x-H(w'))})$$

Based on the dTest, the $e(C_2, T_2) C_3^{yT_1} = C_1$ is true if and only if $w = w'$ and $z' = z$, where z and w correspond to T^d_w. However, given the trapdoor T^d_w, the probability of selecting $z' \in Z_p^*$ such that $z' = z$ is negligible, even $w' = w$. Therefore, the malicious cloud server is not able to launch keyword-guessing attacks by computing the dCRKS ciphertext of all possible keywords.

Second, given the ciphertext $C_w = (C_1, C_2, C_3)$, the cloud server can not produce a legal trapdoor T_w corresponding to C_w. Although the server may select a x', P_d, and w' to generate the trapdoor, the probability of selecting $x' \in Z_p^*$ such that $x' = x$ is negligible, where x is the receiver's private key associated with the C_w. Based on the same analysis, we know that the malicious cloud server is not able to launch

Table 1: A comparison of various schemes.

Schemes	Inside KGAs	Outside KGAs	ZC	ZT	TrC	TeC	CiC
[28]	NO	YES	$l_G + l_H$	$2l_G$	$2E_t$	$P_t + 2E_t$	$P_t + 2E_t$
[17]	NO	NO	$l_G + l_H$	l_G	E_t	$P_t + E_t$	$P_t + E_t$
[29]	YES	YES	$2l_G$	$2l_G + 2l_p$	$2E_t$	$P_t + 2E_t$	$2E_t$
[25]	NO	YES	$3l_G + 2l_{GT}$	l_s	$l_G + l_p$	E_t	$4P_t + 3E_t + tv$
[27]	NO	YES	$2l_G + l_{GT}$	$2l_{GT}$	E_t	$3P_t + E_t$	$P_t + 4E_t$
Ours	YES	YES	$l_G + 2l_{GT}$	$l_G + l_p$	E_t	$P_t + E_t$	$3E_t$

keyword-guessing attacks by creating the trapdoor of all possible keywords.

Lastly, taking g^x, $H(w)$, and $e(g,h)^{1/x}$, the server may build the following equation: $e(g^{(x-H(w_b))}, T_{w_b}) = e(g,h)^{(z/x)((x-H(w_{b'}))/(x-H(w_b)))} \cdot e(g,h)^{-yr((x-H(w_{b'}))/(x-H(w_b)))}$

However, this equation can not help to find correct trapdoor or launch KGAs since T_{w_b} contains a random number. Summarize these reasons; the proposal is secure under the inside KGAs.

4. Performance Analysis

Now, we demonstrate efficiency of the proposal by analyzing its security and calculation cost. With the analysis in Table 1, it shows that only [29] and our scheme are secure to resist inside KGAs. Furthermore, the TTP is removed in our scheme.

To compare performance, let E_t and P_t be the exponential operation and the pairing operation over a bilinear group, respectively. The size of Z_p's element is denoted by l_p. Similarly, the character l_G and l_{G_T} denote the size of G's element and the size of G_T's element, respectively. The size of hash value denotes l_H. For brevity, the calculation cost of creating trapdoor and keyword ciphertext denote TrC and CiC, respectively. The character ZC denotes the size of keyword ciphertext. The ZT denotes the size of trapdoor.

From Table 1, it shows that one E_t is required to create trapdoor in our scheme. Compared with [29], our solution is more efficient to create trapdoor. Meanwhile, the performance overhead of testing is one E_t and one P_t. In [29], it requires two E_t and one P_t.

Lastly, the test procedure can only be run by a specified server. This improves the security of the system.

5. Conclusion

With the wide application of cloud computing, data privacy has become one of critical security issues for mobile users. The ciphertext retrieval is one of the most useful approaches to achieve data privacy for cloud storage. In this paper, we first proposed a new architecture and security model to resist inside KGAs, which is a strong attack on the keyword search scheme. We also proposed an instance of the dCRKS system. Under the security model proposed in our paper, this new scheme has been proven secure to resist inside KGAs. Security analysis and efficiency comparison show that our scheme is effective for the retrieval of encrypted data in cloud computing.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by Scientific Research Fund of Sichuan Provincial Education Department (no. 18ZA0546) and Guangzhou Scholars Project for Universities of Guangzhou.

References

[1] J. Shen, Z. Gui, S. Ji, J. Shen, H. Tan, and Y. Tang, "Cloud-aided lightweight certificateless authentication protocol with anonymity for wireless body area networks," *Journal of Network and Computer Applications*, vol. 106, pp. 117–123, 2018.

[2] J. Xu, L. Wei, Y. Zhang, A. Wang, F. Zhou, and C. Gao, "Dynamic Fully Homomorphic encryption-based Merkle Tree for lightweight streaming authenticated data structures," *Journal of Network and Computer Applications*, vol. 107, pp. 113–124, 2018.

[3] H. Wang, Z. Zheng, L. Wu, and P. Li, "New directly revocable attribute-based encryption scheme and its application in cloud storage environment," *Cluster Computing*, vol. 20, no. 3, pp. 2385–2392, 2017.

[4] J. Cai, Y. Wang, Y. Liu, J.-Z. Luo, W. Wei, and X. Xu, "Enhancing network capacity by weakening community structure in scale-free network," *Future Generation Computer Systems*, 2017.

[5] X. Chen, J. Li, J. Weng, J. Ma, and W. Lou, "Verifiable computation over large database with incremental updates," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 65, no. 10, pp. 3184–3195, 2016.

[6] X. Chen, J. Li, X. Huang, J. Ma, and W. Lou, "New Publicly Verifiable Databases with Efficient Updates," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, pp. 546–556, 2015.

[7] Z. Huang, S. Liu, X. Mao, K. Chen, and J. Li, "Insight of the protection for data security under selective opening attacks," *Information Sciences*, vol. 412-413, pp. 223–241, 2017.

[8] B. Li, Y. Huang, Z. Liu, J. Li, Z. Tian, and S. Yiu, "HybridORAM: Practical oblivious cloud storage with constant bandwidth," *Information Sciences*, 2018.

[9] Z. Liu, Y. Huang, J. Li, X. Cheng, and C. Shen, "DivORAM: Towards a practical oblivious RAM with variable block size," *Information Sciences*, vol. 447, pp. 1–11, 2018.

[10] W. Meng, E. W. Tischhauser, Q. Wang, Y. Wang, and J. Han, "When Intrusion Detection Meets Blockchain Technology: A Review," *IEEE Access*, vol. 6, pp. 10179–10188, 2018.

[11] C. Yuan, X. Li, Q. Wu M J, J. Li, and M. Sun X, "Fingerprint Liveness Detection from Different Fingerprint Materials Using Convolutional Neural Network and Principal Component Analysis," *CMC: Computers, Materials and Continua*, vol. 53, no. 3, pp. 357–371, 2015.

[12] Y. Li, G. Wang, L. Nie, Q. Wang, and W. Tan, "Distance metric optimization driven convolutional neural network for age invariant face recognition," *Pattern Recognition*, vol. 75, pp. 51–62, 2018.

[13] W. Chen, H. Lei, and K. Qi, "Lattice-based linearly homomorphic signatures in the standard model," *Theoretical Computer Science*, vol. 634, pp. 47–54, 2016.

[14] C. Gao, Q. Cheng, P. He, W. Susilo, and J. Li, "Privacy-preserving Naive Bayes classifiers secure against the substitution-then-comparison attack," *Information Sciences*, vol. 444, pp. 72–88, 2018.

[15] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceedings of the IEEE Symposium on Security and Privacy (S&P '00)*, pp. 44–55, IEEE, Berkeley, Calif, USA, May 2000.

[16] D. Boneh and X. Boyen, "Efficient selective-ID secure identity-based encryption without random oracles," in *Advances in cryptology–EUROCRYPT 2004*, vol. 3027 of *Lecture Notes in Comput. Sci.*, pp. 223–238, Springer, Berlin, Germany, 2004.

[17] J. Baek, R. Safavinaini, and W. Susilo, "Public key encryption with keyword search revisited," in *Computational Science and Its Applications-ICCSA*, pp. 1249–1259, Springer, Berlin, Germany, 2008.

[18] W. C. Yau, S. H. Heng, and M. B. Goi, "Off-line keyword guessing attacks on recent public key encryption with keyword search schemes," in *International Conference on Autonomic and Trusted Computing*, pp. 100–105, Springer, 2008.

[19] P. Xu, H. Jin, Q. Wu, and W. Wang, "Public-key encryption with fuzzy keyword search: a provably secure scheme under keyword guessing attack," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 62, no. 11, pp. 2266–2277, 2013.

[20] H. Li, X. Lin, H. Yang, X. Liang, R. Lu, and X. Shen, "EPPDR: an efficient privacy-preserving demand response scheme with adaptive key evolution in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2053–2064, 2014.

[21] H. Li, D. Liu, Y. Dai, T. H. Luan, and X. S. Shen, "Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage," *IEEE Transactions on Emerging Topics in Computing*, vol. 3, no. 1, pp. 127–139, 2015.

[22] H. Li, D. Liu, Y. Dai, T. H. Luan, and S. Yu, "Personalized search over encrypted data with efficient and secure updates in mobile clouds," *IEEE Transactions on Emerging Topics in Computing*, 2016.

[23] L. Fan, X. Lei, N. Yang, T. Q. Duong, and G. K. Karagiannidis, "Secrecy Cooperative Networks with Outdated Relay Selection over Correlated Fading Channels," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7599–7603, 2017.

[24] J. Shen, T. Zhou, X. Chen, J. Li, and W. Susilo, "Anonymous and Traceable Group Data Sharing in Cloud Computing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 4, pp. 912–925, 2018.

[25] L. Fang, W. Susilo, C. Ge, and J. Wang, "Public key encryption with keyword search secure against keyword guessing attacks without random oracle," *Information Sciences*, vol. 238, pp. 221–241, 2013.

[26] R. Xie, C. Xu, C. He, and X. Zhang, "Lattice-based searchable public-key encryption scheme for secure cloud storage," *International Journal of Web and Grid Services*, vol. 14, no. 1, pp. 3–20, 2018.

[27] Y. Zhao, H. Ma, X. Chen, Q. Tang, and H. Zhu, "A new trapdoor-indistinguishable public key encryption with keyword search," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 3, no. 1-2, pp. 72–81, 2012.

[28] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Trapdoor security in a searchable public-key encryption scheme with a designated tester," *The Journal of Systems and Software*, vol. 83, no. 5, pp. 763–771, 2010.

[29] P. Jiang, Y. Mu, F. Guo, X. Wang, and Q. Wen, "Online/offline ciphertext retrieval on resource constrained devices," *The Computer Journal*, vol. 59, no. 7, pp. 955–969, 2016.

[30] J. Li, Y. K. Li, X. Chen, P. P. C. Lee, and W. Lou, "A Hybrid Cloud Approach for Secure Authorized Deduplication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 1206–1216, 2015.

[31] Q. Liu, Y. Guo, J. Wu, and G. Wang, "Effective query grouping strategy in clouds," *Journal of Computer Science and Technology*, vol. 32, no. 6, pp. 1231–1249, 2017.

[32] Y. Liu, J. Ling, Z. Liu, J. Shen, and C. Gao, "Finger vein secure biometric template generation based on deep learning," *Soft Computing*, vol. 21, no. 1, pp. 1–9, 2017.

[33] H. Wang, W. Wang, Z. Cui, X. Zhou, J. Zhao, and Y. Li, "A new dynamic firefly algorithm for demand estimation of water resources," *Information Sciences*, vol. 438, pp. 95–106, 2018.

[34] R. Chen, Y. Mu, G. Yang, F. Guo, and X. Wang, "Dual-server public-key encryption with keyword Search for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 789–798, 2016.

[35] Q.-A. Wang, C. Wang, K. Ren, W.-J. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 847–859, 2011.

[36] J. Li, X. Tan, X. Chen, D. S. Wong, and F. Xhafa, "OPoR: Enabling proof of retrievability in cloud computing with resource-constrained devices," *IEEE Transactions on Cloud Computing*, vol. 3, no. 2, pp. 195–205, 2015.

[37] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An ID-based linearly homomorphic signature scheme and its application in blockchain," *IEEE Access*, 2018.

[38] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586–615, 2003.

[39] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," *Journal of Cryptology. The Journal of the International Association for Cryptologic Research*, vol. 17, no. 4, pp. 297–319, 2004.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 4879496, 11 pages
https://doi.org/10.1155/2018/4879496

Research Article

A High-Security and Smart Interaction System Based on Hand Gesture Recognition for Internet of Things

Jun Xu, Xiong Zhang, and Meng Zhou

Display Center, School of Electronic Science and Engineering, Southeast University, Sipailou 2#, Nanjing, Jiangsu 210096, China

Correspondence should be addressed to Xiong Zhang; zhangxiongseu@163.com

Received 28 February 2018; Revised 12 April 2018; Accepted 9 May 2018; Published 6 June 2018

Academic Editor: Ilsun You

In this work, we propose a vision-based hand gesture recognition system to provide a high-security and smart node in the application layer of Internet of Things. The system can be installed in any terminal device with a monocular camera and interact with users by recognizing pointing gestures in the captured images. The interaction information is determined by a straight line from the user's eye to the tip of the index finger, which achieves real-time and authentic data communication. The system mainly contains two modules. The first module is an edge repair-based hand subpart segmentation algorithm which combines pictorial structures and edge information to extract hand regions from complex backgrounds. Second, the position which the user focuses on is located by an adaptive method of pointing gesture estimation, which adjusts the offsets between the target position and the calculated position due to lack of depth information.

1. Introduction

Internet of Things (IoT) which connects objects around us together to the Internet becomes one of the most important information technologies nowadays. One of the issues of IoT is how to make the object-object interaction and human-object interaction smart and safe. Data collected from users may include sensitive or private information of their daily activities; hence security protection and privacy preserving are vital for the development of IoT. According to the principle of information generation and the technical architecture, IoT consists of three layers, namely, perception [1, 2], network protocol [3, 4], and application layers [5, 6]. The perception layer is composed of various sensors and gateways, such as temperature sensor, two-dimensional barcode, camera, and Global Positioning System (GPS). The perception layer recognizes objects and collects information with Radio Frequency Identification (RFID) and Wireless Sensor Networks (WSNs); therefore, the security issue for the perception layer is essentially on node capture attacks of RFID and WSNs [7, 8]. The network protocol layer is the central pivot of IoT which transmits and processes the information from the perception layer. As one of the key technologies in the network layer, wireless mobile communication networks employ traditional encryption and authentication techniques to improve the security and privacy of IoT [9, 10]. The application layer realizes the intelligent services by connecting IoT and users, such as city management, intelligent transportation, telemedicine, and smart home. The application layer also faces the challenges of security threat. For example, the attackers can capture the unattended equipment and send data to the application system by identity impersonation and data modification. Therefore, this study is aimed at developing a secure and authentic interaction system based on computer vision for smart home in the application layer of IoT. As shown in Figure 1, our interaction system can control the cursor of a computer or a smart TV by recognizing users' pointing gestures. The cursor is located immediately according to the line from eye to fingertip when the user points to the target. Since the interaction data is produced only when the user's eye and fingertip are detected and the pointing direction is recognized, the system can effectively avoid unauthorized access of illegal users, which is more secure than the traditional point-touch devices.

Many studies have been conducted to recognize pointing gestures based on computer vision. When human interacts with an object by pointing gesture, the pointing direction is estimated by two points on the camera perspective line.

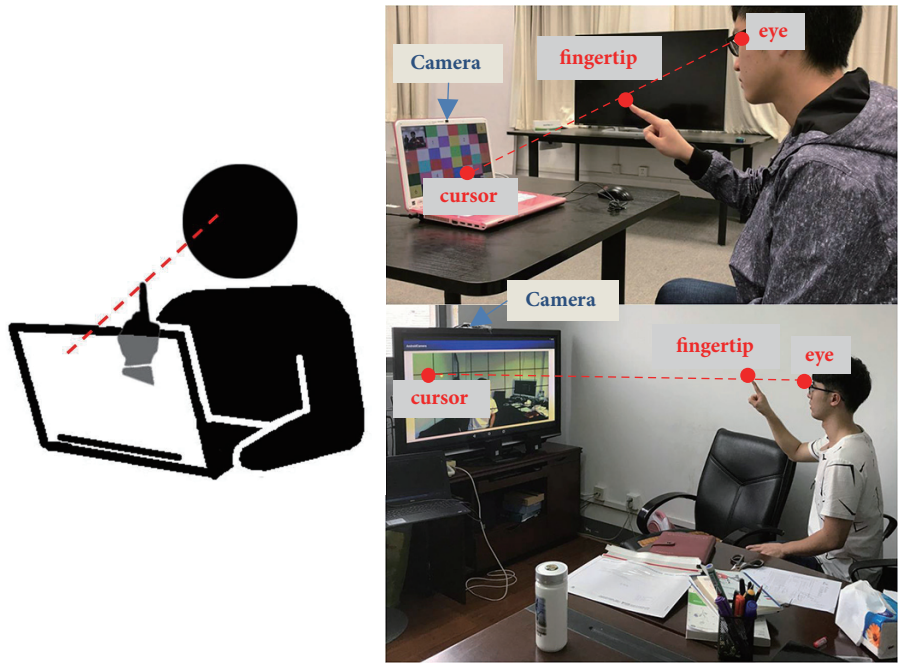


Figure 1: Applications of our interaction system based on pointing gesture recognition.

According to the different input cameras, pointing gesture recognition technologies can be classified into 3D methods [11–13] and 2D methods [14–16]. 3D methods rely on specific input devices, such as stereo camera or Kinect, while 2D methods use less expensive and more easily available cameras, with reduced computation cost for 2D information. However, pointing gesture recognition based on 2D methods is still challenging. First, since 2D information of the hand provides relatively weak and ambiguous characteristics, the discrimination of bare hand from background is easily influenced by the large variability of hand appearances and the presence of other skin-like objects. Some studies have been conducted to deal with this problem [17–19]. For example, Li and Wachs [17] proposed a weighted elastic graph matching method to detect and recognize ten hand postures under complex backgrounds. The average recognition accuracy reached 97%; however the performance was unclear when hand shape distorts. In [18], hand regions were extracted based on the Bayesian model of visual attention by combining shape, texture, and color cues. The long run time, 2.65 s, of this method clearly made it unsuitable for real-time application. Gonzalez et al. [19] proposed a hand segmentation method based on pixel color and edge orientation to deal with the overlap of hand and face, whereas the segmentation results were affected by thin lines under the chin or over the collar. The second challenge is to estimate the pointing position determined by intersection between the interaction plane and the pointing vector due to lack of depth information. In order to solve this problem, some interaction systems based on 2D pointing gesture methods only utilize information of pointing direction instead of pointing position [14, 15], and some other systems require users to make coordinate calibration before operating [16]. In this paper, we propose an edge repair-based hand subpart segmentation algorithm, which accurately and effectively segments the palm and finger regions from the background by using 2D information. Furthermore, on the basis of the hand segmentation algorithm, an adaptive method of pointing direction estimation is developed which can adjust the eye-fingertip line during operation.

2. System Model

We develop a vision-based interaction system using pointing gesture recognition as a node in the application layer of IoT. When the user points to the screen, a straight line from the eye to the fingertip determines the position on the screen where the cursor should locate. Figure 2 illustrates the flow chart of our system. First, 2D images are captured by a normal camera. Second, the user's eye is detected by the AdaBoost classifier based on Haar-like features [20]. Third, hand region is segmented from the complex backgrounds using the edge repair-based hand subpart segmentation algorithm. Fingertip of the index finger is detected by combining convex hull and convexity defect features [21]. If both the eye and the hand are located, an adaptive method of pointing direction estimation is proposed to obtain the pointing position according to the eye-fingertip line. Finally, the cursor is moved to the pointing position. Consequently, the system is secure and authentic because it can only be activated when the eye and hand of the user are both detected.

2.1. Methods of Eye Detection and Hand Segmentation. In our system, the face position is first coarsely determined by background subtraction and skin color detection. Then connected regions with large areas are extracted. In order to verify face from these regions, an ellipse model is employed

to select the approximately elliptical candidates because of the face shape [22]. For each candidate, the AdaBoost algorithm based on Haar-like features is employed for eye detection. The AdaBoost classifier is trained by the positive samples of eye images and the negative samples of all kinds of background images without human eyes. A result of eye detection is shown in Figure 2.

In order to segment hand regions efficiently, we propose an edge repair-based hand subpart segmentation algorithm which includes four procedures as illustrated in Figure 3.

Firstly, a hierarchical chamfer matching algorithm (HCMA) [23] is used to locate the whole hand region in the binary image produced by combining skin color detection and background subtraction. As shown in Figure 4, the chamfer distance image of the hand is searched for the optimal position which matches the hand template from the previous frame. After the distance image is traversed, the optimal position is figured out by calculating the minimum edge distance Ed.

$$\mathrm{Ed} = \frac{1}{3}\sqrt{\frac{1}{N}\sum_{i=0}^{N-1} v_i^2}, \tag{1}$$

where v_i is the pixel value that the template hits and N is the number of the contour pixels in the template. In order to accelerate the matching, a pyramid structure is built by halving the resolution of the distance image gradually. At the top level of the pyramid structure, a grid of positions is chosen to start the matching. Each position and its neighborhood are computed for Ed. If a smaller edge distance is found, the template is moved to the new position (X, Y) in the distance image of level n by

$$\begin{aligned} X &= \frac{X_r + 2^n - 1}{2^n} \\ Y &= \frac{Y_r + 2^n - 1}{2^n}, \end{aligned} \tag{2}$$

where

$$\begin{aligned} X_r &= c_x + \cos\theta \cdot s_x \cdot x - \sin\theta \cdot s_y \cdot y \\ Y_r &= c_y + \sin\theta \cdot s_x \cdot x + \cos\theta \cdot s_y \cdot y \end{aligned} \tag{3}$$

where (x, y) are the coordinates of the points in the template and (c_x, c_y), (s_x, s_y), and θ are translation, scaling, and rotation parameters, respectively. For each start position, the position with local minima is obtained and then used as the start position at the next level $n - 1$. When all the levels are traversed, the optimal position is finally identified, which is illustrated by the blue rectangle in Figure 4(b).

Secondly, the located hand region is analyzed to detect the palm and fingers separately by combining pictorial structures and Histogram of Oriented Gradient (HOG) features [24]. Figure 5(a) shows the hand model based on pictorial structures, which includes a root part of the palm and five-finger parts. Thus the hand configuration is denoted by $L = \{l_0, l_1, l_2, \ldots, l_n\}$ where the subscript 0 corresponds to the palm part and the subscripts $1 \sim n$ correspond to the finger parts.

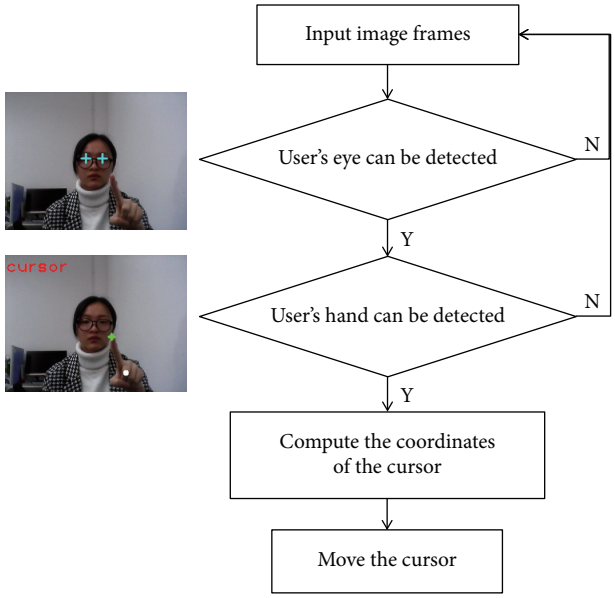


Figure 2: Flow chart of our proposed system.

Given an image I and a set of hand model parameters $\boldsymbol{\theta} = \{u_i, c_{ij}\}$, the maximum a posteriori (MAP) probability of L is represented by

$$\begin{aligned} P(L \mid I, \theta) &\propto P(I \mid l_0, u_0) \\ &\cdot \prod_{i=1}^{n} \left(P(I \mid l_i, u_i) \cdot P(l_i, l_0 \mid c_{i0})\right), \end{aligned} \tag{4}$$

where u_i is the appearance parameter of part i and c_{ij} is the connection parameter between part i and part j. Due to the different characteristics of the palm and finger in the model, two support vector machine (SVM) classifiers are employed to detect the subparts of the hand. On the one hand, the classifier for the palm part is trained by HOG features. On the other hand, the input feature vector of the classifier for the finger part considers both HOG features and the spatial relationship between the finger and palm. Let the state of the finger part i be l_i (x_i, y_i, θ_i). Assuming (x_j, y_j) and (x_0, y_0) are the coordinates of the joint and the palm center, the relative position (x_i, y_i) is computed by

$$\begin{aligned} x_i &= \frac{\left(x_j - x_0\right)}{w_0} \\ y_i &= \frac{\left(y_j - y_0\right)}{w_0}, \end{aligned} \tag{5}$$

where w_0 is the size of the palm part. θ_i is the absolute part orientation as shown in Figure 5(a). Figure 5(b) shows the detection result of the image in Figure 4.

Thirdly, since the blurry border of the hand and face probably leads to an incompletely connected hand silhouette, we propose an edge repair method to recover the contour of each subpart. The edge image of each subpart is extracted to

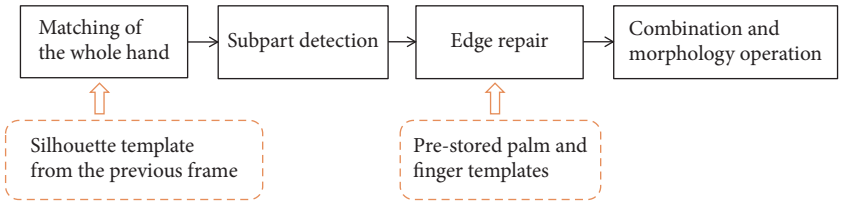


Figure 3: Architecture of the edge repair-based hand subpart segmentation algorithm.

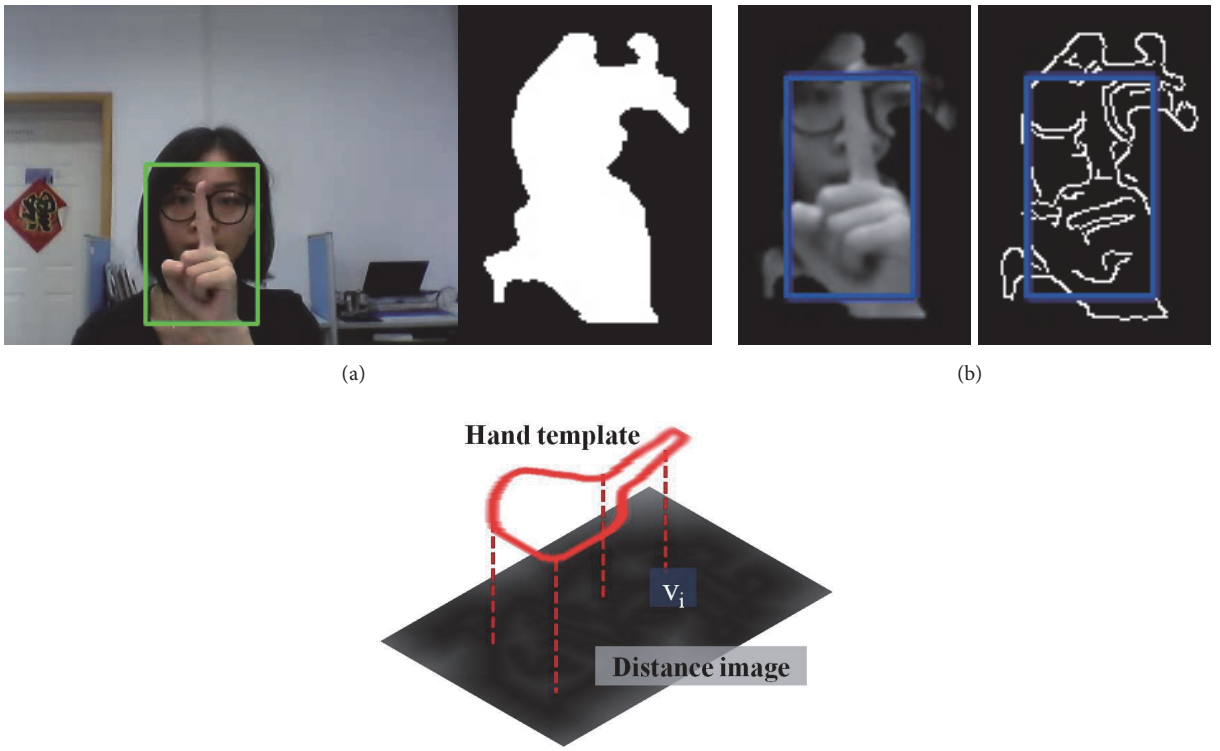


Figure 4: Template matching of the whole hand by HCMA. (a) Hand segmentation by skin color detection and background subtraction. (b) Edge extraction by Canny edge detector. (c) The edge distance is computed by the pixel values v_i, which the hand template hits.

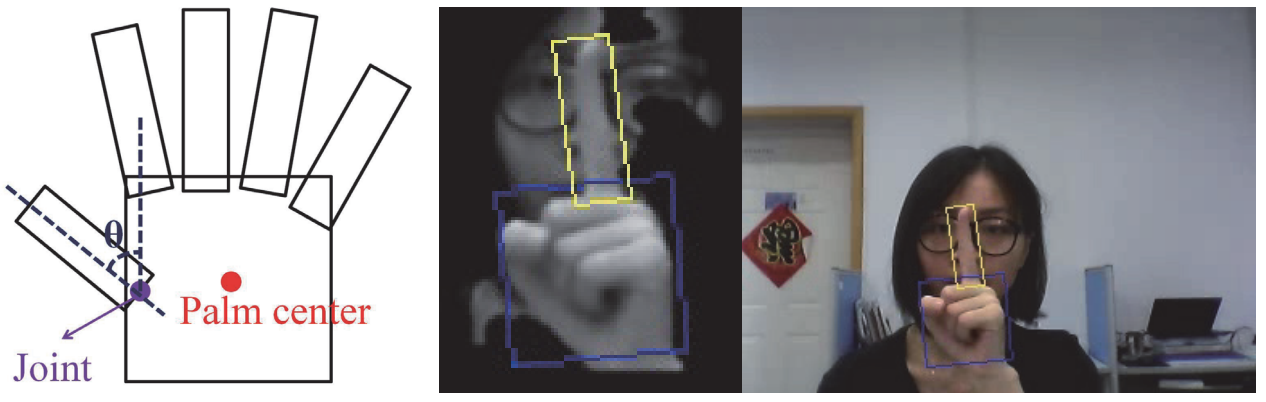


Figure 5: Palm and finger detection based on pictorial structure and HOG features. (a) Hand modeling. (b) Detection result.

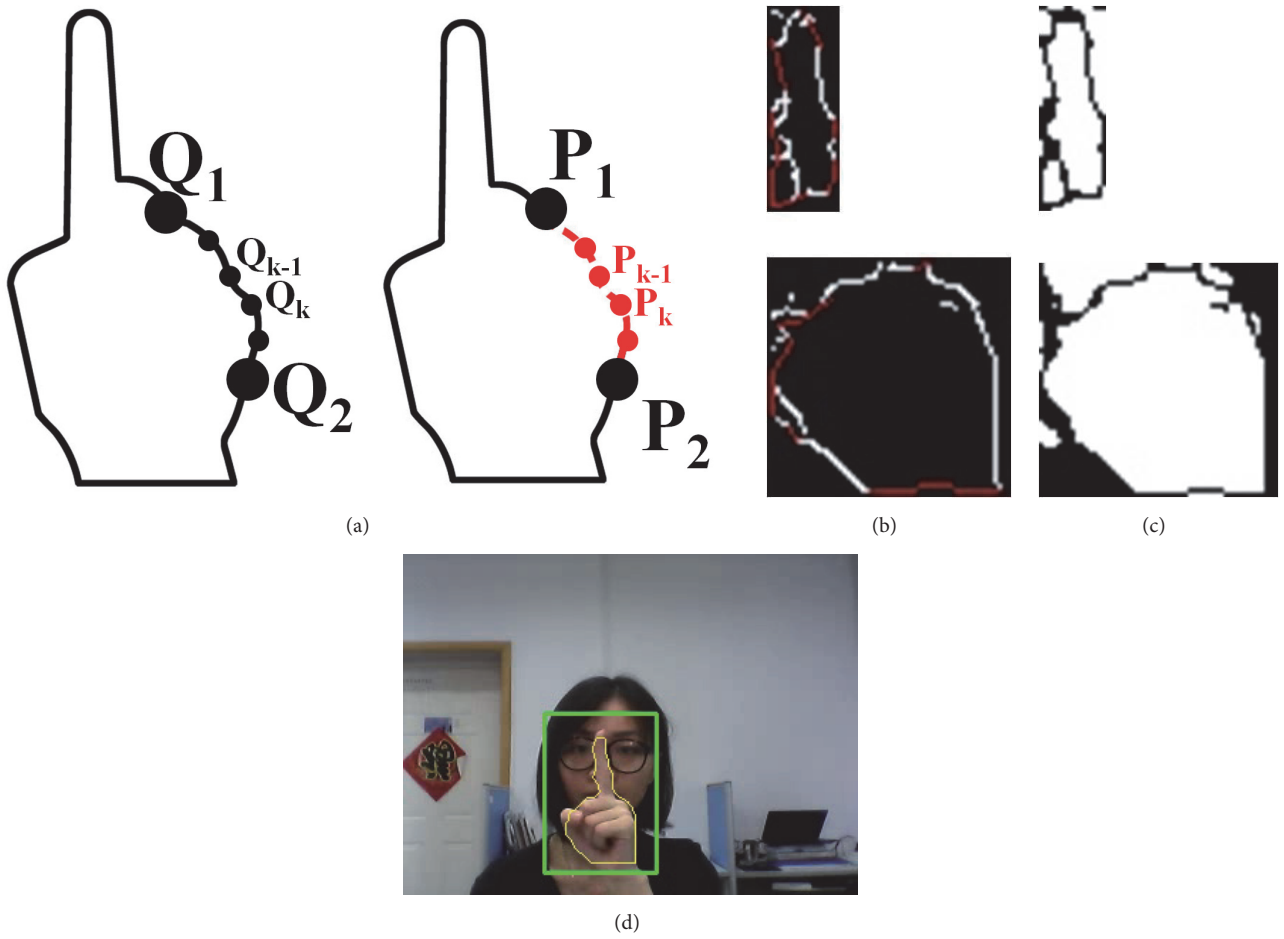


FIGURE 6: Edge repair. (a) Breakpoint connection. (b) Edge repair results. (c) Refined segmentation using edge repair results. (d) Combination of all subparts.

detect where the contour breaks. An edge point is determined as a breakpoint if one or two adjacent points among its eight-neighborhood are edge points. For each breakpoint P_i, a contour point Q_i in the prestored template is found, which is the closest to P_i based on Euclidean distance. As shown in Figure 6(a), in order to connect the adjacent breakpoints P_1P_2 depending on the template, Q_1Q_2 is divided into several subsegments by $\{Q_k \mid k = 1, \ldots, m\}$ and a set of $\{P_k \mid k = 1, \ldots, m\}$ is generated by $P_k = P_{k-1} + Q_k - Q_{k-1}$. Then, P_{k-1} and P_k are connected by the Catmull–Rom interpolation method [25]. Figure 6(b) shows the result of edge repair where the connections of breakpoints are illustrated in red.

Finally, the repaired edge images of all the subparts are used to extract the refined hand pixels from the coarse binary image as shown in Figure 6(c). All images of hand subparts are combined to generate the whole hand region in Figure 6(d).

2.2. Cursor Positioning. When both the eye and the fingertip are detected, the position on the screen where the user points to can be figured out by the line extending forward from the dominant eye to the fingertip. Since the depth information of the eyes and fingertips is unavailable and the pointing ways of different users are individually different, the offset exists between the target position and the calculated position. Thus, we propose an adaptive method of pointing direction estimation which can adjust the eye-fingertip line through a learning process.

As shown in Figure 7, a three-dimensional coordinate system with the camera position as the origin is established. The intersection point (x_i, y_i) of the screen and the eye-fingertip line can be calculated through similar triangle theory as

$$x_i = x_{E\_real} - \left(\frac{x_{E\_real} - x_{F\_real}}{z_{E\_real} - z_{F\_real}} \right) \cdot z_{E\_real}$$
$$y_i = y_{E\_real} - \left(\frac{y_{E\_real} - y_{F\_real}}{z_{E\_real} - z_{F\_real}} \right) \cdot z_{E\_real}, \quad (6)$$

where $(x_{E\_real}, y_{E\_real}, z_{E\_real})$ and $(x_{F\_real}, y_{F\_real}, z_{F\_real})$ represent the eye's coordinates and the fingertip's coordinates in the 3D coordinate system, respectively. Since the relationship of those coordinates in 3D coordinate system is similar to that

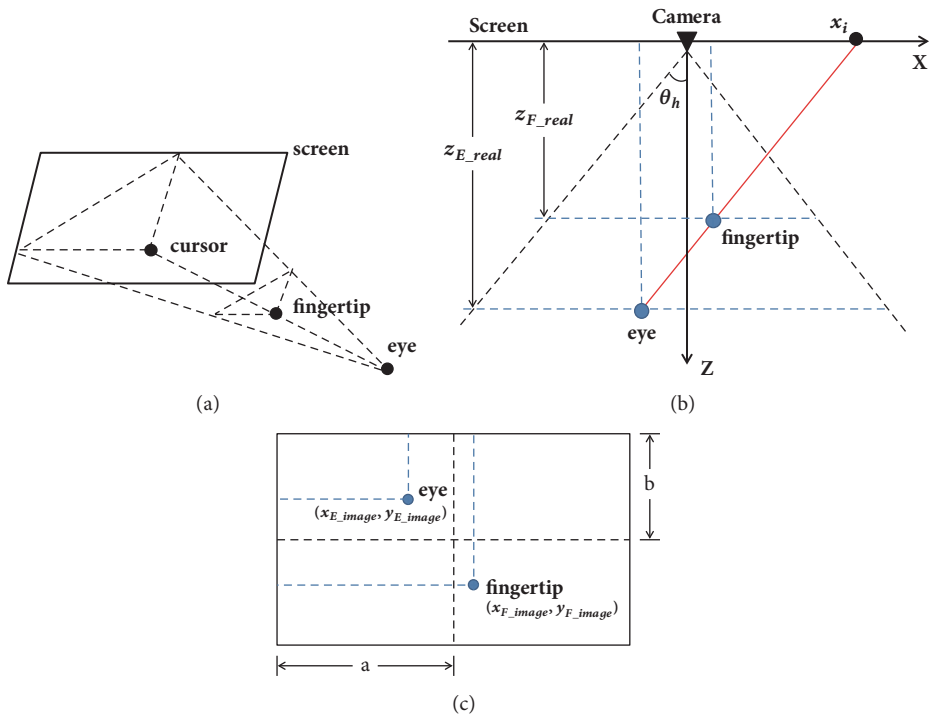


FIGURE 7: Cursor positioning based on the eye-fingertip line. (a) 3D coordinate system. (b) X-Z plane of (a). (c) The captured image with the resolution of 2a × 2b.

in the captured image (Figure 7(c)), $(x_{E\_real}, y_{E\_real}, z_{E\_real})$ can be computed by

$$x_{E\_real} = z_{E\_real} \cdot \tan\theta_h \cdot \frac{x_{E\_image} - a}{a}$$
$$y_{E\_real} = z_{E\_real} \cdot \tan\theta_v \cdot \frac{y_{E\_image} - b}{b}, \quad (7)$$

where $(x_{E\_image}, y_{E\_image})$ are the coordinates of the human eye in the image, θ_h and θ_v are the camera's angles of view in the horizontal and vertical direction, and (a, b) is half of the spatial resolution of the image. $(x_{F\_real}, y_{F\_real}, z_{F\_real})$ can be figured out by a similar equation to (7).

Then the cursor's coordinates $(x_{\text{cursor}}, y_{\text{cursor}})$ are computed by transforming (x_i, y_i) into the screen coordinates in pixels as

$$x_{\text{cursor}} = \left(x_i + \frac{1}{2}W\right) \cdot \frac{x_{\text{res}}}{W}$$
$$y_{\text{cursor}} = y_i \cdot \frac{y_{\text{res}}}{H}, \quad (8)$$

where $(x_{\text{res}}, y_{\text{res}})$ is the spatial resolution of the screen and W and H are the width and height of the screen.

It can be seen that the cursor's position is closely related to the distance between the fingertip and the screen z_{F_real} and the distance between the eye and the screen z_{E_real}. However, the two distances cannot be accurately obtained from our monocular vision-based system. The inaccurate distances will lead to an offset between the calculated position and the target position of the cursor. Hence, we propose a method to adjust the cursor's position through a learning process. Firstly, z_{E_real} is estimated according to the face's area and z_{F_real} is initialized to z_{E_real} minus 30 based on users' habits. Secondly, when the cursor is not located at the desired position, the user is allowed to alter the cursor's position by moving the fingertip slightly. The cursor's coordinate is adjusted from $(x_{\text{cursor}}, y_{\text{cursor}})$ to $(x'_{\text{cursor}}, y'_{\text{cursor}})$ by

$$x'_{\text{cursor}} = s_x \cdot d_x + x_{\text{cursor}}$$
$$y'_{\text{cursor}} = s_y \cdot d_y + y_{\text{cursor}}, \quad (9)$$

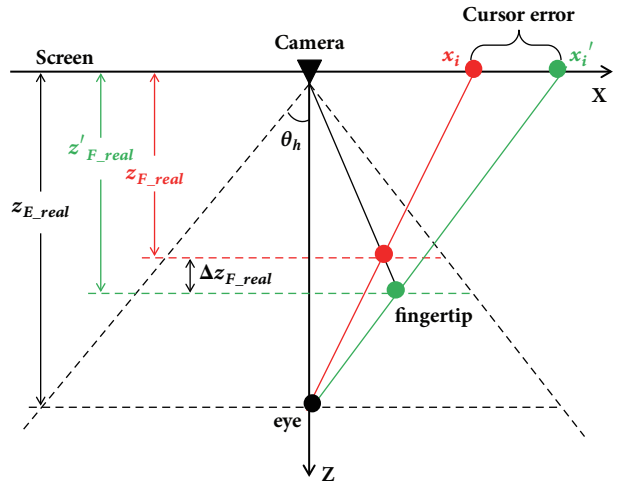


FIGURE 8: Error analysis model.

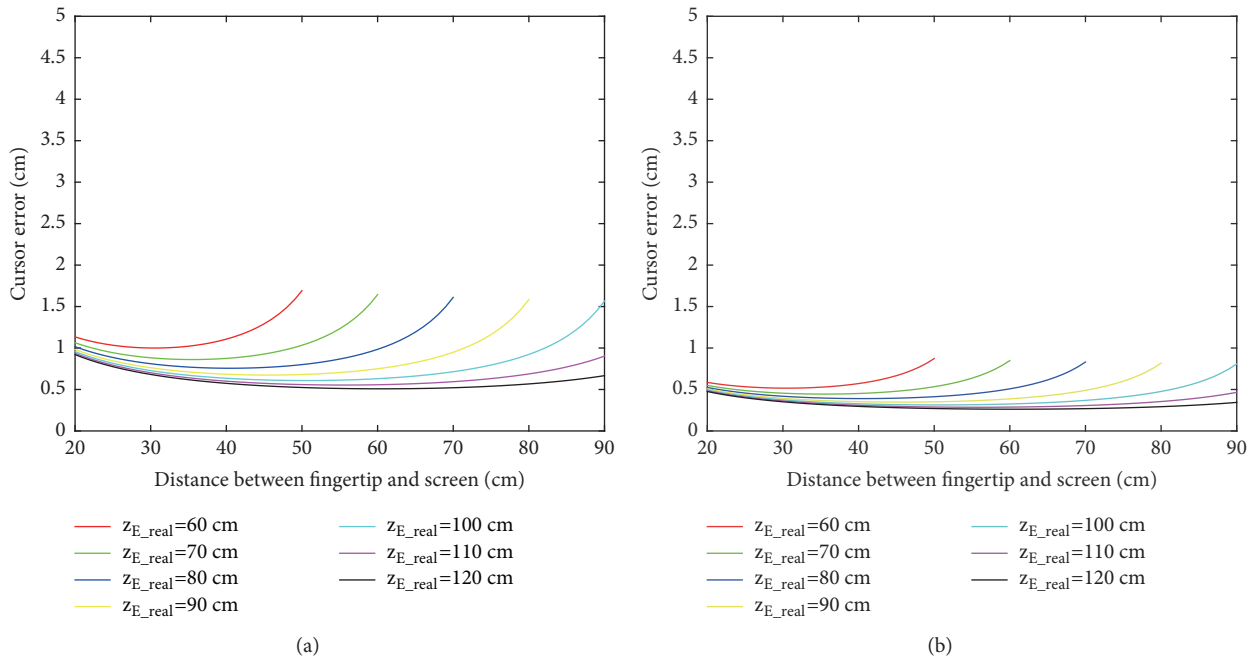


FIGURE 9: Cursor errors with different z'_{F_real} and z_{E_real} (a) when $x'_i = 15.5$ cm and (b) when $x'_i = 8$ cm.

where (d_x, d_y) represents the moving distance of the fingertip in the successive frames and (s_x, s_y) is the multiple coefficient. Our system monitors the fingertip's movement and records $(x'_{\text{cursor}}, y'_{\text{cursor}})$ when the fingertip moves a short distance after a pause. Then z_{F_real} can be adjusted by (10) from $z^{(n)}_{F\_real}$ to $z^{(n+1)}_{F\_real}$.

$$\begin{aligned} z^{(n+1)}_{F\_real} &= (1-\alpha)\, z^{(n)}_{F\_real} + \alpha \cdot z'_{F\_real} \\ &= (1-\alpha)\, z^{(n)}_{F\_real} + \alpha \cdot \frac{1}{2}\left(z^{x}_{F\_real} + z^{y}_{F\_real}\right), \end{aligned} \tag{10}$$

where α is the update rate. $z^x_{F\_real}$ and $z^y_{F\_real}$ are estimated based on x'_{cursor} and y'_{cursor}, respectively, according to the derivation of (6)–(9). After the adjusting process works several times, z_{F_real} will approach the real value.

2.3. Feasibility Verification of Our System. In order to verify whether our cursor positioning system is feasible or not, we perform error analysis to evaluate how much the cursor error depends on the eye-hand position in the direction of z-axis and the eye-hand position in the image. Because of the lack of depth information, the real locations of the fingertip and the eye in the direction of z-axis cannot be obtained, represented by z'_{F_real} and z'_{E_real}. Hence the estimated values of z_{F_real} and z_{E_real} are used in our pointing direction estimation method. Generally, the offsets of Δz_{F_real} and Δz_{E_real} exist between the real values and the estimated values, which lead to the errors of the cursor position. Similarly, assuming only X-Z plane is considered; Δx_{F_image} and Δx_{E_image} possibly exist due to the detection deviations of the fingertip's and eye's x-coordinates in the image. Note that the adjusting process of the cursor position is not activated here in order to analyze the error.

As shown in Figure 8, it is assumed that Δz_{F_real} exists and the eye is fixed on z-axis; the cursor's coordinate x_i is computed by

$$x_i = z_{E\_real} \cdot \tan\theta_h \cdot \frac{x_{F\_image} - a}{a} \cdot \frac{z_{F\_real}}{z_{E\_real} - z_{F\_real}}. \tag{11}$$

Then the cursor error Δx_i is calculated by subtracting the deviation value x_i from the real value x'_i as

$$\begin{aligned} \Delta x_i &= z_{E\_real} \cdot \tan\theta_h \cdot \frac{x_{F\_image} - a}{a} \\ &\quad \cdot \left(\frac{z'_{F\_real}}{z_{E\_real} - z'_{F\_real}} - \frac{z'_{F\_real} - \Delta z_{F\_real}}{z_{E\_real} - z'_{F\_real} + \Delta z_{F\_real}} \right), \end{aligned} \tag{12}$$

where $\Delta z_{F\_real} = z'_{F\_real} - z_{F\_real}$. $(x_{F\_image} - a)/a$ can be computed by using similar triangle principle in Figure 8. Therefore,

$$\begin{aligned} \Delta x_i = x'_i &- \frac{\left(z_{E\_real} - z'_{F\_real}\right) \cdot x'_i}{z'_{F\_real}} \\ &\cdot \frac{z'_{F\_real} - \Delta z_{F\_real}}{z_{E\_real} - z'_{F\_real} + \Delta z_{F\_real}}. \end{aligned} \tag{13}$$

Assuming the width of the screen is 31 cm, Figure 9 shows the errors of the cursor with different z'_{F_real} and z_{E_real} when the user points to the edge of the screen and a quarter of the screen. It is indicated that the cursor error becomes larger when the user points to the position closer to the screen edge. The cursor error increases as z_{E_real} decreases and as the hand

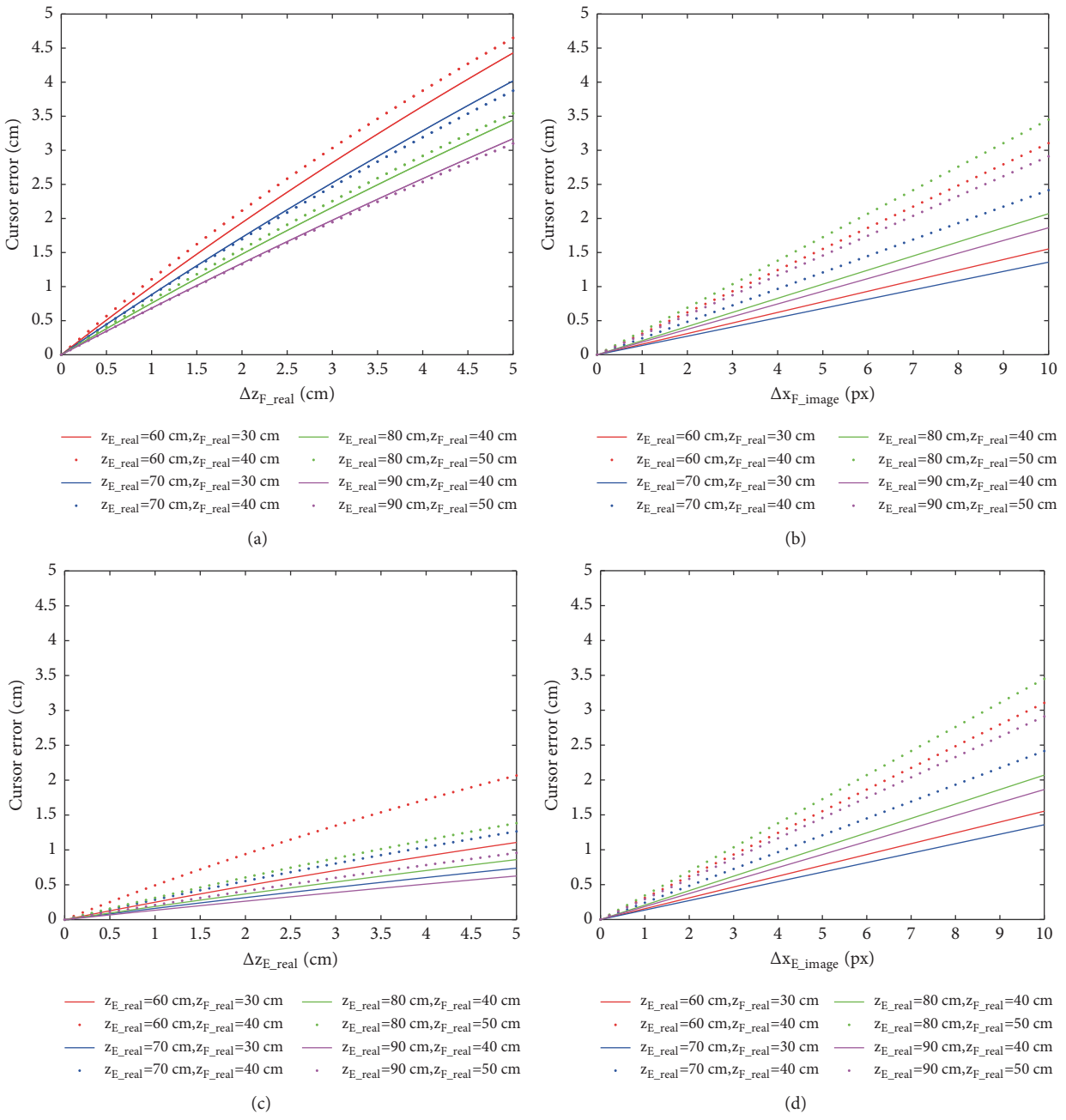


Figure 10: Maximum cursor errors effected by the four offsets of Δz_{F_real}, Δz_{E_real}, Δx_{F_image}, and Δx_{E_image}.

is closer to the head. Moreover, the cursor error is acceptable when operating the laptop in a close distance less than 100 cm and the error will be the smallest when the hand locates at the midpoint of the screen and the eye.

Figure 10 illustrates how much the cursor error depends on the four offsets of Δz_{F_real}, Δz_{E_real}, Δx_{F_image}, and Δx_{E_image} using the similar derivation of (13), where the two distances from the screen denoted by z_{E_real} and z_{F_real} are set according to users' operation habits. Not that the user is assumed to point to the edge of the screen, which causes the maximum error among the entire screen. As shown in Figure 10, the maximum cursor errors fall below 3 cm and 1.5 cm when the offsets of eye and hand positions on z-axis are less than 3 cm, and they fall below 3 cm when the offsets of eye and hand positions in the image are less than 9 pixels. The precision is acceptable for the block-level positioning and the following experiments will prove the attainability when the adjusting process is activated.

3. Experimental Results

Our proposed system was evaluated on a laptop with a 2D camera and the resolutions of the screen and the captured image were 1366 $*$ 768 and 320 $*$ 240, respectively. Besides,

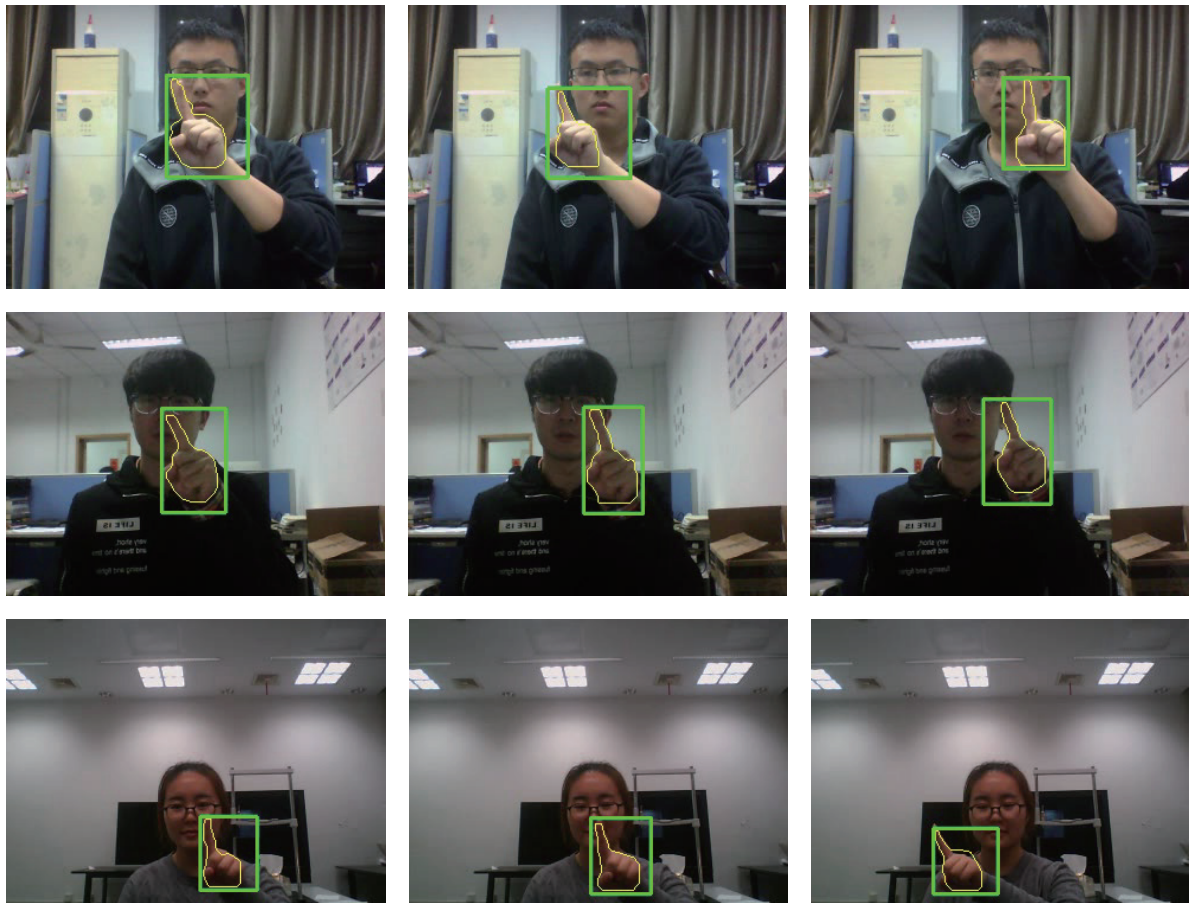


Figure 11: Segmentation results of the edge repair-based hand subpart segmentation algorithm under different situations.

on the basis of the proposed methods mentioned above, the user was not permitted to keep shaking his/her body back and forth which would affect the adjusting process of cursor positioning. Ten subjects were asked to operate the system by pointing gesture.

Firstly, some segmentation results of the edge repair-based hand subpart segmentation algorithm are shown in Figure 11, where yellow lines indicate the hand regions. It demonstrates that our method can extract hand pixels accurately under different complex backgrounds, including human faces with similar color. Moreover, the method is independent of users and robust to various hand appearances.

Then in order to evaluate the accuracy of cursor positioning, the computer screen was divided into several blocks as shown in Figure 12 and the subjects pointed to the four blocks marked with "1" to "4" repeatedly. Each block was pointed 15 times as a set of data.

Figure 12 shows the qualitative experimental results of cursor positioning by different subjects under different backgrounds. In each row, the same subject points to the different positions of the screen and the relative positions between the fingertip and the eye appear different. The cursors highlighted by the black circles are successfully located at the corresponding marked blocks. It is demonstrated that our cursor positioning method is robust to diverse individuals and different situations. Besides, it is also implied that our hand segmentation method works well when hand overlaps the face.

Table 1: Average errors of cursor positioning for the four marked blocks.

Error	d	d'(cm)
Block 1	0.707	2.38
Block 2	0.641	2.16
Block 3	0.680	2.29
Block 4	0.536	1.80
Average	**0.641**	**2.16**

Let the length of the blocks be 1. When the subject points to a block, the cursor error d between the calculated position and the desired position is computed by Euclidean distance. Figure 13 shows the errors of several sets of data. The horizontal axis represents the repeat times when the users point to the blocks. It is proved that the error decreases significantly after several times owing to the adjusting process in the adaptive pointing direction estimation method. Moreover, Table 1 shows the average errors of cursor positioning for the four marked blocks. The errors d' are estimated by $d' = d \times S^{1/2}$, where S represents the area of the block. Because the

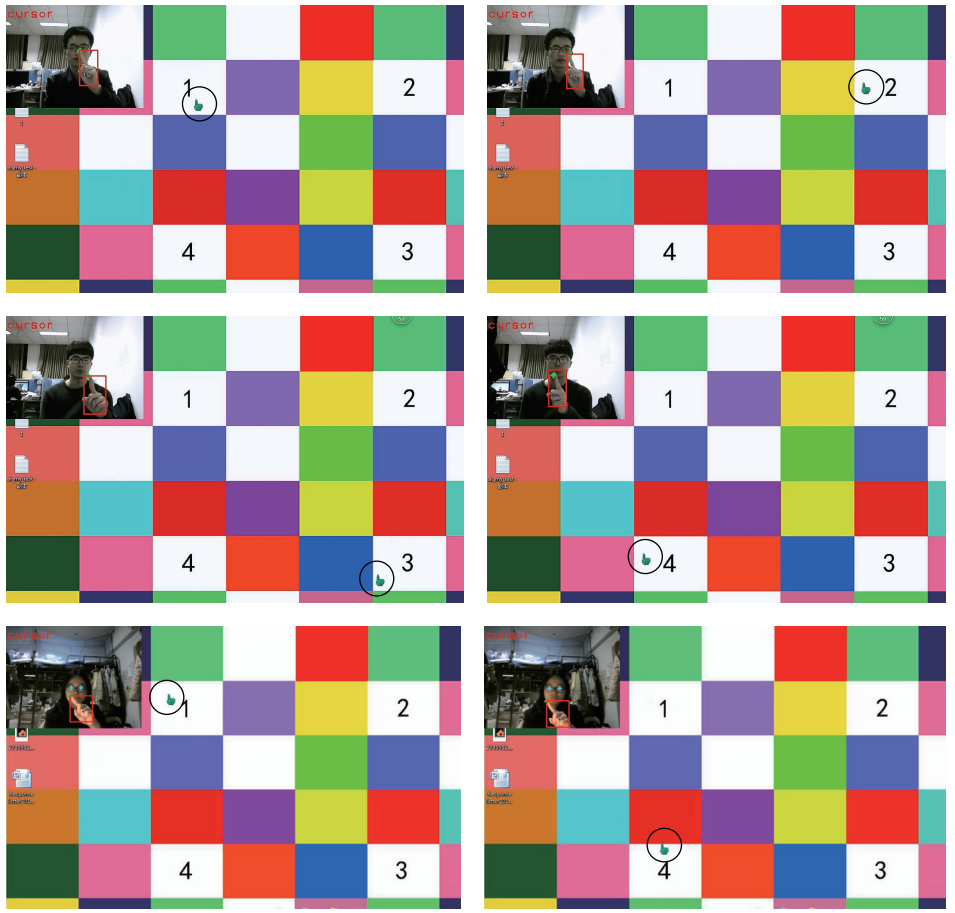


Figure 12: Qualitative experimental results of cursor positioning where black circles highlight the cursor's positions.

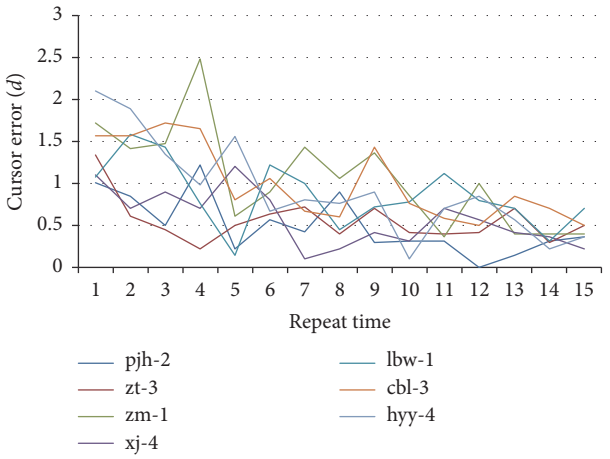


Figure 13: Cursor errors with the repeat times as the horizontal axis.

size of block icon in Windows 8/10 Metro can be set as 3 cm or 6 cm, it can be concluded that the positioning errors are small and acceptable for our application requirement.

Besides, our system can work in real time with an average speed of 131 ms per frame.

4. Conclusion

A security and smart Internet of Things interaction system based on hand gesture recognition is proposed in this work. When a user points to screen, the target position which the user points to is estimated by a straight line from the user's eye to the fingertip. Therefore, the interaction between human and computer should be activated by the coexisting of eye and hand. In our system, we employ a novel hand segmentation algorithm which combines the pictorial structure model, hierarchical chamfer matching algorithm, and curve fitting that segments hand regions accurately and efficiently. Furthermore, we propose an adaptive pointing direction estimation method for cursor calibration. An adjusting process is presented to correct the offsets between the target position and the calculated position arising from diverse individuals and lack of depth information. Experimental results show that our system provides a natural and friendly human-computer interaction and possesses satisfactory and accuracy of cursor positioning under complex backgrounds.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by Perspective Joint Research Project of Jiangsu Province Technology Project (BY2016076-07).

References

[1] Y. Liu, C. Cheng, T. Gu, T. Jiang, and X. Li, "A Lightweight Authenticated Communication Scheme for Smart Grid," *IEEE Sensors Journal*, vol. 16, no. 3, pp. 836–842, 2016.

[2] T. Zhang, L. Yan, and Y. Yang, "Trust evaluation method for clustered wireless sensor networks based on cloud model," *Wireless Networks*, pp. 1–21, 2016.

[3] X. Su, H. F. Yu, W. Kim, C. Choi, and D. Choi, "Interference cancellation for non-orthogonal multiple access used in future wireless mobile networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, article no. 231, 2016.

[4] C. R. Panigrahi, J. L. Sarkar, and B. Pati, "Transmission in mobile cloudlet systems with intermittent connectivity in emergency areas," *Digital Communications and Networks*, vol. 4, no. 1, pp. 69–75, 2018.

[5] E. Park, Y. Cho, J. Han, and S. J. Kwon, "Comprehensive Approaches to User Acceptance of Internet of Things in a Smart Home Environment," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2342–2350, 2017.

[6] K. M. A. Alheeti, A. Gruebler, and K. McDonald-Maier, "Using discriminant analysis to detect intrusions in external communication for self-driving vehicles," *Digital Communications and Networks*, vol. 3, no. 3, pp. 180–187, 2017.

[7] J. W. Ho, *Distributed Detection of Node Capture Attacks in Wireless Sensor Networks*, InTech, 2010.

[8] P. Tague and R. Poovendran, "Modeling adaptive node capture attacks in multi-hop wireless networks," *Ad Hoc Networks*, vol. 5, no. 6, pp. 801–814, 2007.

[9] P. Tague, D. Slater, J. Rogers, and R. Poovendran, "Vulnerability of network traffic under node capture attacks using circuit theoretic analysis," in *Proceedings of the INFOCOM 2008: 27th IEEE Communications Society Conference on Computer Communications*, pp. 664–672, usa, April 2008.

[10] M. Conti, R. Di Pietro, L. V. Mancini, and A. Mei, "Mobility and cooperation to thwart node capture attacks in MANETs," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, Article ID 945943, 2009.

[11] H. Kim, Y. Kim, D. Ko, J. Kim, and E. C. Lee, "Pointing Gesture Interface for Large Display Environments Based on the Kinect Skeleton Model," in *Future Information Technology*, vol. 309 of *Lecture Notes in Electrical Engineering*, pp. 509–514, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[12] Z. Li and R. Jarvis, "Visual interpretation of natural pointing gestures in 3D space for human-robot interaction," in *Proceedings of the 11th International Conference on Control, Automation, Robotics and Vision, ICARCV 2010*, pp. 2513–2518, Singapore, December 2010.

[13] M. J. Reale, S. Canavan, L. Yin, K. Hu, and T. Hung, "A multi-gesture interaction system using a 3-D iris disk model for gaze estimation and an active appearance model for 3-D hand pointing," *IEEE Transactions on Multimedia*, vol. 13, no. 3, pp. 474–486, 2011.

[14] M. Sugi, M. Nikaido, Y. Tamura, J. Ota, and T. Arait, "Development of gesture-based interface for deskwork support system," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2006*, pp. 5171–5176, chn, October 2006.

[15] R. C. Luo, S.-R. Chang, and Y.-P. Yang, "Tracking with pointing gesture recognition for human-robot interaction," in *Proceedings of the 2011 IEEE/SICE International Symposium on System Integration, SII 2011*, pp. 1220–1225, jpn, December 2011.

[16] Z. Černeková, C. Malerczyk, N. Nikolaidis, and I. Pitas, "Single camera pointing gesture recognition for interaction in edutainment applications," in *Proceedings of the 24th Spring Conference on Computer Graphics, SCCG 2008*, pp. 121–126, svk, April 2008.

[17] Y.-T. Li and J. P. Wachs, "Recognizing hand gestures using the weighted elastic graph matching (WEGM) method," *Image and Vision Computing*, vol. 31, no. 9, pp. 649–657, 2013.

[18] P. K. Pisharady, P. Vadakkepat, and A. P. Loh, "Attention based detection and recognition of hand postures against complex backgrounds," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 403–419, 2013.

[19] M. Gonzalez, C. Collet, and R. Dubot, "Head Tracking and Hand Segmentation during Hand over Face Occlusion in Sign Language," in *Trends and Topics in Computer Vision*, vol. 6553 of *Lecture Notes in Computer Science*, pp. 234–243, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[20] P. Viola and M. J. Jones, "Robust Real-Time Object Detection," in *Proceedings of the in International Workshop on Statistical and Computational Theories of Vision – Modeling, Learning, Computing*, 2001.

[21] P. D. Le and V. H. Nguyen, "Remote Mouse Control Using Fingertip Tracking Technique," in *AETA 2013: Recent Advances in Electrical Engineering and Related Sciences*, vol. 282 of *Lecture Notes in Electrical Engineering*, pp. 467–476, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[22] P. Gejguš and M. Šperka, "Face tracking in color video sequences," in *Proceedings of the Spring Conference on Computer Graphics, SCCG 2003 - Conference Proceedings*, pp. 245–249, svk, April 2003.

[23] G. Borgefors, "Hierarchical Chamfer matching: a parametric edge matching algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 849–865, 1988.

[24] H. Zhong, J. P. Wachs, and S. Y. Nof, "A collaborative telerobotics network framework with hand gesture interface and conflict prevention," *International Journal of Production Research*, vol. 51, no. 15, pp. 4443–4463, 2013.

[25] J. Tsao, "Interpolation artifacts in multimodality image registration based on maximization of mutual information," *IEEE Transactions on Medical Imaging*, vol. 22, no. 7, pp. 854–864, 2003.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 4296934, 14 pages
https://doi.org/10.1155/2018/4296934

Review Article

Study to Improve Security for IoT Smart Device Controller: Drawbacks and Countermeasures

Xin Su,[1] **Ziyu Wang**,[2] **Xiaofeng Liu**,[1] **Chang Choi**,[3] **and Dongmin Choi**[4]

[1]*College of IOT Engineering, Changzhou Key Laboratory of Robotics and Intelligent Technology, Hohai University, Changzhou 213022, China*
[2]*Nanjing Ivtime, Co., Ltd., Nanjing 210000, China*
[3]*Department of Computer Engineering and IT Research Institute, Chosun University, Gwangju 61452, Republic of Korea*
[4]*Division of Undeclared Majors, Chosun University, Gwangju 61452, Republic of Korea*

Correspondence should be addressed to Dongmin Choi; jdmcc@chosun.ac.kr

Received 20 February 2018; Accepted 7 May 2018; Published 31 May 2018

Academic Editor: Pino Caballero-Gil

Including mobile environment, conventional security mechanisms have been adapted to satisfy the needs of users. However, the device environment-IoT-based number of connected devices is quite different to the previous traditional desktop PC- or mobile-based environment. Based on the IoT, different kinds of smart and mobile devices are fully connected automatically via device controller, such as smartphone. Therefore, controller must be secure compared to conventional security mechanism. According to the existing security threats, these are quite different from the previous ones. Thus, the countermeasures applied should be changed. However, the smart device-based authentication techniques that have been proposed to date are not adequate in terms of usability and security. From the viewpoint of usability, the environment is based on mobility, and thus devices are designed and developed to enhance their owners' efficiency. Thus, in all applications, there is a need to consider usability, even when the application is a security mechanism. Typically, mobility is emphasized over security. However, considering that the major characteristic of a device controller is deeply related to its owner's private information, a security technique that is robust to all kinds of attacks is mandatory. In this paper, we focus on security. First, in terms of security achievement, we investigate and categorize conventional attacks and emerging issues and then analyze conventional and existing countermeasures, respectively. Finally, as countermeasure concepts, we propose several representative methods.

1. Introduction

As mobile-based technologies continue to develop and propel society further into the information age, our surroundings are rapidly becoming ubiquitous mobile environments. According to the changes in the mobile-based society, increasing numbers of people are becoming mobile-based. Thus, it is a well-known fact that mobile devices are pervasive in today's global environment. The smartphone, a representative mobile device, provides various applications related to our daily life, which expand our living radius more efficiently. Moreover, IoT devices are fully connected via device controller, called smartphone. As an IoT device controller, smartphone gives various applications to users through widely developed applicable software and hardware products. Consequently, users feel increasingly drawn to use it in their daily lives in IoT environment via smartphone. The major parts of future IoT network infrastructure will be based on high-speed cellular networks that will be available everywhere, even in hostile places. Subsequently, the number of persons connecting to the Internet wirelessly will surpass the number connecting via wired infrastructure. Concomitant with the growth in mobile-related techniques, security and privacy issues will also increase. Likewise, the parts of human everyday life associated with IoT infrastructure will increase over time, and, thus, IoT devices will change from simple information-processing terminals to assisting and guiding their owners' whole lives as private secretaries. As a result, the information stored on IoT devices is more closely related to personal privacy. Hence, security and privacy issues are more important compared to non-IoT infrastructure-based society. In addition, today's device controllers include various

sensors to serve a variety of applications that deal with human biometric information, because some of these sensors collect and manage fingerprint, voice, iris, signature, and even behavior patterns. These types of information are unique information that can be used to verify the legitimacy of a user. Moreover, in accordance with development trends, developers are increasingly focusing on human biometric information for user identification and healthcare services [1, 2]. In accordance with the changes in the types of information handled by device controllers, attack patterns have also changed. Compared to the traditional attack patterns, attacks today tend to focus on human error. Thus, traditional and existing attack countermeasure schemes may not be suitable for emerging attack issues. Therefore, it is worth noting that security techniques must search wider and deeper than before. The remainder of this paper is organized as follows. In Section 2, we introduce related work, from traditional security threats to existing countermeasures. In Section 3, we show the drawbacks and downsides associated with existing countermeasures. In Section 4, we present countermeasure concepts and proposals that ensure resilience against emerging security threats. In Section 5, we present comparative results. Finally, we conclude the paper in Section 6 with a brief discussion.

2. Related Work

Much work has been carried out analyzing security threats. In this section, we categorize security threats and countermeasures. First, we divide security threats into two groups, traditional and emerging models, and then associate countermeasures with each security threat.

2.1. Traditional Threat Models

2.1.1. Guessing. Typically, users can access their systems with their own ID/password. In password guessing attacks, users' access rights are compromised by the two types of attack models discussed below.

Brute Force. In brute force attacks, the attacker continuously and repeatedly tries every possible passcode combination until the correct passcode is found. In scenarios where the passcode is short, only a short time is needed for the attacker to succeed, whereas a long passcode requires more time [3, 4].

Dictionary. In contrast to the brute force attack type, dictionary attacks try the most probable passcodes. Typically, many people have a tendency to choose short/meaningful words with no concern of being exposed. Thus, dictionary attacks try to find passcodes comprising word/phrases appearing in a dictionary [5, 6].

2.1.2. Replay. In replay attacks, successfully transferred valid data packets are delayed or repeated in order for attackers to get inside and pretend to be a legitimate user. Replay attacks are well known and thus countermeasures to avoid such attacks have been determined. However, in mobile authentication environments, replay attacks are being applied as a new type of attack [7, 8].

2.1.3. Spyware. Spyware is predominantly used for malicious purposes, with the aim of hiding from users, such as gathering information, tracking the behavior of users, and monitoring systems without the users' consent. From desktop PCs to mobile devices, spyware is a typical method for fulfilling the malicious purpose of attackers [9, 10].

2.2. Traditional Countermeasures

2.2.1. Text-Based Password. Passwords based on text are commonly used, even though the vulnerabilities are well known.

One of the factors influencing the security level of such passwords is their length. Long passwords take a long time for attackers to crack. However, users tend to use short passwords that are easier to remember [11].

To ensure adequate security, the following rules should be adhered to when using text-based passwords [12]:

(i) Do not use less than eight characters (more is better).

(ii) Do not use words that have meaning (meaningless is better).

(iii) Do not store them externally (keeping them only in your mind is better).

(iv) Do not unify (a unique password for each system is better).

(v) Do not maintain them over a long period (changing them periodically is better).

2.2.2. Personal Identification Number. Personal Identification Numbers (PINs) are commonly used for banking services, credit card authentication, mobile phone unlock systems, door lock systems, and so forth. A PIN comprises numeric keys only and typically ranges from four to eight digits [13, 14].

2.2.3. One-Time Password. A One-Time Password (OTP) is valid for one login session only and is not storable. An OTP is algorithmically generated based on time or mathematics for timely use. There are two approaches: static and dynamic. The static approach uses a document with a list of codes, whereas the dynamic approach may utilize methods such as SMS, hardware, and software tokens [15, 16].

2.3. Emerging Threat Models. As the number of mobile users continues to increase, the number of persons who want to connect to the Internet or use various online-based application services is also increasing. As is well known, mobile devices in public places are not as safe as wired network-connected devices that are located in secured spaces. However, unlike previous types of attacks, emerging attacks occur everywhere, even in secured places. This is because emerging threats are focused on the structural defects of mobile devices and the vulnerabilities of their owners. Therefore, we divide the threats into two types, owner and device, where owner means human errors and device means screen size and touch function.

Owner vulnerability, in other words, human errors, means the weakness induced by human mistakes. In most

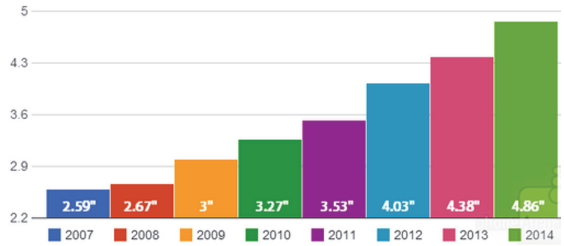


Figure 1: Trend in smartphone screen size.

cases, persons who use a mobile device do not take into account their surroundings, which may cause serious problems. One example of a device's structural defect is screen size greater than four or five inches. With a larger screen size for displaying information, a touch mechanism is also present on the screen. Typically, owners input secure information via touchable screen keypad to communicate with the device. Most structural defects, including owner vulnerabilities, result from this factor. Figure 1 shows the trend in smartphone screen size over time [17].

As screen sizes are bigger than before, owners should try their best to protect the information displayed onscreen from attackers.

2.3.1. Shoulder Surfing. For getting information, shoulder surfing uses direct observation techniques via the naked eyes. Nowadays, shoulder surfing is most effective, because it is easy to look over someone's shoulder when that mobile user is looking at the device screen without worrying about his/her surroundings.

There are two types of shoulder surfing attacks: single-attacker-based shoulder surfing with naked eyes and multiple-attacker-based shoulder surfing with naked eyes. Figure 2 illustrates the differences between these two attack types.

(i) Single Attacker. Single naked-eye-based shoulder surfing is commonly used by a single attacker. Preparing the attack incurs no cost for the attacker, but it is powerful enough to obtain the user's secret information [18, 19].

(ii) Multiple Attackers. In the case of single attack, the success rate is low. In contrast, when an attacker cooperates with other attackers to get users' secret information, even when the attacker only obtains some part of the secret information, they can combine the parts with each other. In this manner, they obtain all the information with a higher success rate compared to the single attacker.

2.3.2. Recording. The basic concept underlying recording is shoulder surfing. Shoulder surfing is based on naked human eyes only. As shown in Figure 3, recording is an extended peeking concept using all kinds of recording devices for the attack. It can also be divided into two types: single and multiple recording devices.

(i) Single Device. Single device recording is commonly used by a single attacker. Preparing for the attack incurs minimal cost for the attacker, but it is more powerful than single shoulder surfing because of the replayable video data.

(ii) Multiple Devices. As with the multiple shoulder surfing attack, single attacker or multiple attackers cooperate with each other with their video recording devices to obtain users' secret information. Thus, even though one attacker or device obtains only a part of the secret information, they can combine the parts with each other. Then, the entire information is obtained with a higher success rate than in the single case [20].

2.3.3. Hybrid. As shown in Figure 4, a hybrid attack is a combination of shoulder surfing and recording.

In this scenario, the naked eyes and multiple recording devices are used to obtain the password.

2.3.4. Smudge. The smudge attack is focused on the oily residue on the smartphone. Typically, a person who touches the screen uses their finger to touch the screen for usability. However, in the case of key arrays where the virtual keypad displayed on the screen is unchangeable, it is harmful. This is because finger touch positions can be restored by tracing the oily residue. In addition, in the case of pattern lock, it can be easily restored to the original onscreen pattern shape and direction, as shown in Figure 5 [21, 22].

2.3.5. Password Guessing with Sensors. Password guessing attacks use mobile device embedded sensors for guessing and obtaining secret information. Assume that a user inputs his/her own password or pattern using touch screen; if the attacker captures the touch sensors, he/she might get the information from the sensor data. Through the information, he/she can guess the key position or pattern in reverse [23, 24].

2.4. Existing Attack Countermeasures. Recently, in order to ensure safety against the various attacks occurring against mobile devices, researchers have proposed various concepts and mechanisms according to cost and security level [25]:

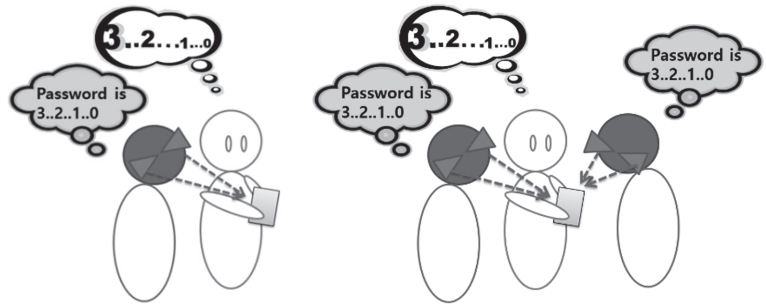
(i) High security with low cost—keystroke dynamics

(ii) High security with high cost—physical biometrics, token, and smart cards

(iii) Low security with low cost—password and PIN

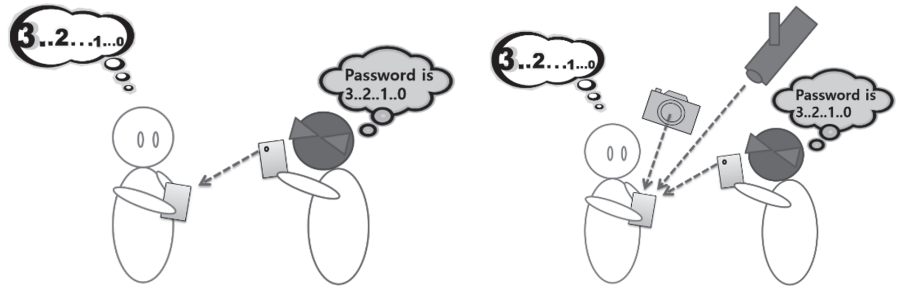
However, compared to the existing security mechanisms, several mechanisms are still not suitable owing to the higher cost. Thus, in this section, we omit several high-cost mechanisms such as token, smart card, and some physical biometrics.

2.4.1. Graphical Password. Many types of graphical password authentication mechanisms have been proposed by researchers. One such type is pattern-based mechanism [26]. Nowadays, many modified versions of pattern-based authentication mechanisms have been proposed with appealing advantages. Google has developed and imported a pattern-lock algorithm into their Android OS-based smartphones, and it is typically used worldwide.



(a) Single (b) Multiple

Figure 2: Comparison of shoulder surfing types.



(a) Single (b) Multiple

Figure 3: Comparison of single and multiple recording attack types.

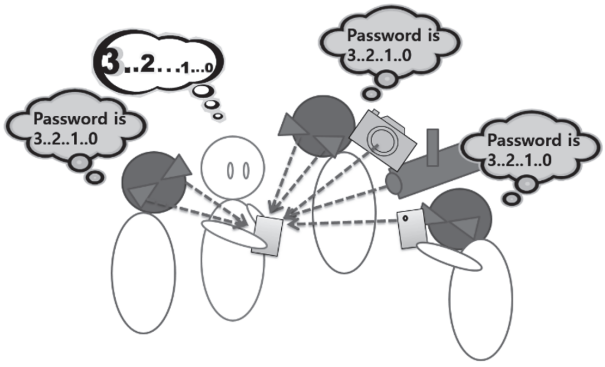


Figure 4: Hybrid attack scenario.

2.4.2. Fingerprint. Starting a few years ago, fingerprint modules have been embedded in mobile devices and developed with higher recognition rate and higher processing speed. Thus, fingerprint authentication is more suitable for mobile devices [27].

2.4.3. Voice. Based on differences in the voice signature of humans, researchers have proposed various person-identifying mechanisms. Das et al. even proposed a related algorithm for cellular phones [28].

2.4.4. Signature. Signature recognition is divided into two types: 2D-based and 3D-based.

Two-dimensional signature recognition is the traditional technique used to authenticate users. Nowadays, mobile-based 2D signature recognition mechanism is based on touchscreen. In the 3D-based mechanism, as shown in Figure 6, users take a magnetic object in hand, and then a compass sensor embedded in the mobile device checks the variances in the magnetic field. The mobile device verifies the user in accordance with the information of the changing log of the magnetic field [29, 30].

2.4.5. Behavior. With the network connection, network service providers can verify users by comparing their behavior profile. The profile is made by the service provider, and typically it has the user's interaction pattern with the service. Without the network connection, the mobile device checks the user's interaction with applications to identify whether the user is legitimate [31].

2.4.6. Keystroke Dynamics. On the basis of the concept of the keypad and keyboard typing pattern being different, the authentication mechanism verifies user. In contrast to typical authentication mechanisms, keystroke dynamics continuously check and verify users in the background. For one-time authentication, users register their password pattern for algorithmic learning. Figure 7 shows an example of keystroke pattern recognition [32].

3. Drawbacks and Countermeasures

3.1. Text Password

Drawbacks: various kinds of emerging attacks

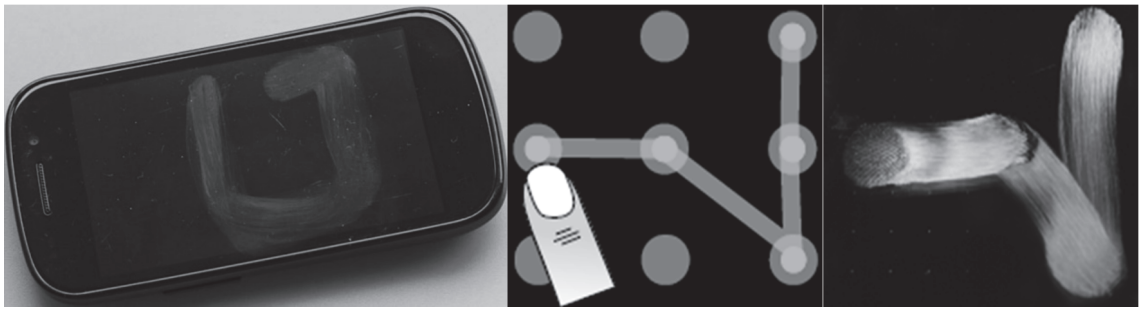
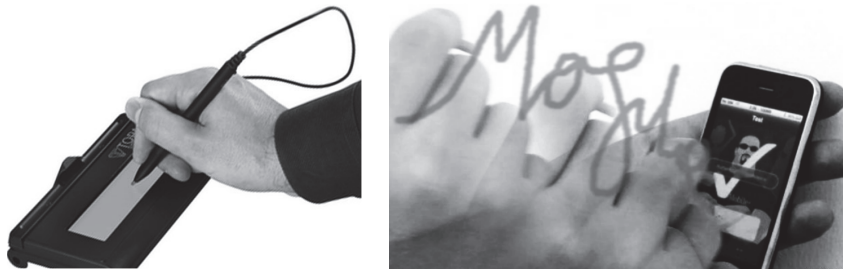


Figure 5: Imprinted finger direction on smartphone touch screen along with pattern direction.



(a) 2D touchpad signature (b) 3D magnetic signature

Figure 6: Signature comparison.

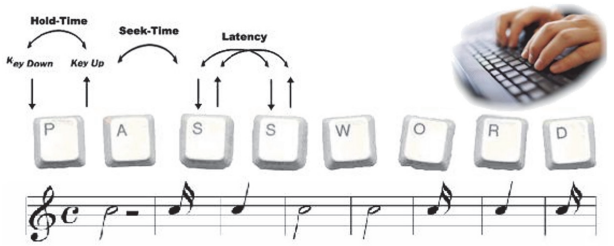


Figure 7: Example of keystroke pattern recognition.

Text passwords are vulnerable to traditional password guessing, shoulder surfing, recording, smudge, and password guessing with sensors. Once the secret information is exposed, it is over.

Countermeasures: information hiding, algorithm or mechanism combination, and indirect input method.

Wiping all information in the password mechanism which could give a clue for guessing the password is mandatory. Thus, the screen size or display window should display less information with a small area. Typically, the password input method is a virtual keypad or keyboard. Thus, a small keypad is required for higher resiliency against shoulder surfing, recording, and smudge.

3.2. Graphical Password

Drawbacks: shoulder surfing, recording, smudge, and password guessing with sensors

In this case, there is little or no information for guiding users to input password displayed on the screen of the mobile device. However, even though the password is not composed of textual information, it is still vulnerable to shoulder surfing, smudge, and password guessing with sensors. This is because graphical components are also easy for the attacker to remember. Thus, the attacker can easily obtain the information using electronic replayable recording devices.

Countermeasures: information hiding, algorithm, or mechanism combination

The development of nonvisible user graphical password authentication scheme can be a countermeasure. Typically, graphical components are visible. Thus, compared to text-based password, graphical password may be more vulnerable in specified conditions. Assuming that the graphical password-using user predefined a picture selection mechanism, whereas in the case of text-based password a shoulder surfing attacker may experience difficulty obtaining the original password from a long distance, a picture is larger than text. Thus, a shoulder surfing attacker may get the passcode from a long distance. Therefore, the use of one or more authentication mechanisms in concert with a graphical passcode is required.

3.3. PIN

Drawbacks: various kinds of emerging attacks

As with text password mechanisms, PINs are vulnerable to traditional password guessing, shoulder surfing, recording,

smudge, and password guessing with sensors. Moreover, a PIN utilizes only four to eight digits for user authentication. Thus, it is harmful when exposed. Even when only one or two digits are exposed, attackers can easily guess the whole secret information, because the number of cases is small compared the text password mechanisms.

Countermeasures: information hiding, algorithm or mechanism combination, and indirect input method

The direct password input method is not mandatory. In the case of social engineering attack scenarios, the attacker could see the user's behavior, while the user is operating his/her mobile device, including password input actions. If the mechanisms involve direct password input procedure, all kinds of display information, including user guide information, should be hidden against the emerging attack types. The best way to do this is to develop an indirect password input method. In the case of indirect input, guessing all the information through the displayed information only is difficult. Therefore, traditional PIN codes can support personal identification and authentication combined with an indirect input mechanism.

3.4. Fingerprint

Drawbacks: fake fingerprint and smudge

This method is still expensive and vulnerable to fake fingerprints [33]. Even though mobile device makers have produced flagship models with this technology, the higher price is not appropriate for mainstream users. Further, fingerprint is still vulnerable to fake fingerprint. Therefore, it is not very attractive at this time [32].

Countermeasures: algorithm or mechanism combination

To develop fake fingerprint resilient module, a combination of one or two more authentication mechanisms for identifying the correct owner is required. The development of high-resolution fingerprint detection device for detecting whole fingers is also required. However, these are only temporary solutions; therefore, it is mandatory to use two-way authentication.

3.5. Voice

Drawbacks: recording and environmental problems

Voice is vulnerable and may not be able to identify the owner if the owner's voice has changed owing to environmental reasons, such as fatigue, cold, and flu. Further, recording attacks may be able to circumvent voice recognition as attackers could record the user's voice when the user is logging into his/her device with his/her voice.

Countermeasures: algorithm or mechanism combination

It is not mandatory to use voice recognition mechanism for user verification only. In addition, it is recommended that it not be used for user verification with high proportion when the authentication mechanism uses two or more authentication mechanisms for user verification.

3.6. Signature

Drawbacks: recording, password guessing with sensors, and need for extra devices

A signature has been a typical way of verifying a legitimate user for a long time. However, with plain 2D, it is easy to imitate original signature characteristics. Thus, signature-based user authentication schemes in which information is written in 3D space, in short 3D signature, are being proposed. However, successful 3D signature schemes require extra devices to collect magnetic variation information.

Countermeasures: combination of authentication scheme and development of new types of signature authentication schemes

The security vulnerability of traditional 2D-based signature is well known. Thus, in the case of 2D signatures, one or more user verification mechanisms should be combined. In the case of 3D-based signatures, researchers still do not have sufficient results from their proposals, and they are still in the development stage. However, most 3D-based signature mechanisms use device embedded sensors, and they are thus reasonable.

3.7. Behaviors

Drawbacks: password guessing with sensors

User behavior patterns can be sensed and stored via device-embedded sensors as a kind of pattern data. Let us assume the case of mobile device application operation. Most applications typically are activated by the user's finger touch. Even if the user does not use the application via the touch screen, they can use other types of sensors such as gyro, gesture, accelerometer, magnetic, and light. Thus, at least one sensor can be used for user input in the mobile environment. However, sensors also enable vulnerability to password guessing. User behavior patterns that are recorded in real time are difficult to protect by encryption techniques because of hardware and power source limitations. Hence, before the sensor data profiling process, the raw sensor data can easily be captured by attackers using password guessing. Even in cases after the process of data profiling, exposure to common attacks using malicious codes is easy.

Countermeasures: no encryption in mobile device, selective use of raw data profiling by server, and combination of multisensor data

In order to reduce meaningless energy consumption, data encrypting and profiling tasks should be carried out on the server. Mobile devices should conserve their energy and be used only for detecting or sensing owner's behaviors. Then, some selected part of the data extracted from each sensor's raw data should be transferred to the server via algorithmic selection. Using the selected data, the server could profile and encrypt the data for authentication usage and then transfer these data to the mobile device. On receiving the data, the mobile device could use them to compare the current user's behavior pattern.

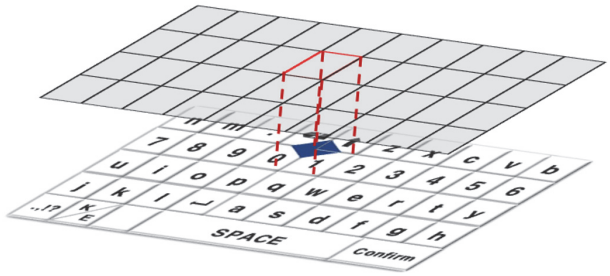


Figure 8: Grid mesh, pointer, and circular keyboard layout.

3.8. Keystroke Dynamics

Drawbacks: shoulder surfing, recording, and password guessing with sensors

Keystroke dynamics technique is visible and recordable. Thus, it is vulnerable when the legitimate user is inputting his/her passcode with a unique keystroke pattern in public places. Even in secret places, the typing sound also can be rhythmically generated. Thus, keystroke dynamics are also vulnerable.

Countermeasures: hiding visible information and applying sound shadowing technique

All kinds of visible information can give a clue to attackers as to how to circumvent the secure scheme. Thus, without the core contents appearing on the screen to guide the user, other information must disappear. Moreover, there should be a solution that enables hiding of the original keystroke pattern. The typical passcode input scheme does not consider hiding the original keystroke pattern. Thus, it is vulnerable when the attacker uses sound-based password guessing.

4. Proposals Related to Countermeasure Concepts

Among the existing attack countermeasures, we chose the five existing security mechanisms most commonly used worldwide. We briefly explain our proposals for ensuring that systems are robust to emerging attacks according to each proposal.

4.1. Text Password Robust to Emerging Attacks

(i) Combination of Circular Keypad and Grid Mesh Pointer. Our proposal is based on circular keypad and grid mesh. A circular keypad layout has no boundary; all edges of the keypad are connected with each other. Moreover, it can move every direction on the mobile screen as it is edgeless. Thus, it would appear to be circulating in every direction.

For secure input, a grid mesh that includes a secret pointer overlaps the circular keypad. The secret pointer location does not appear onscreen, and it is selected by the user in the registration stage. Figure 8 shows the proposed circular keypad-based grid mesh with secret pointer mechanism.

With this mechanism, the user should know the secret pointer location and password and then choose his/her password using the secret pointer. Figure 9 shows how the password is chosen via secret pointer in our proposed mechanism.

Figure 9(a) illustrates how a user inputs the letter "k" with the secret pointer. The grid mesh is fixed, and the keypad is freely moved in any direction. Thus, to choose the letter "k," the user slides the circular keypad to the right three grid spaces, as shown in Figure 9(b). Finally, the user slides the keypad up two grid spaces and then touches the edge of the screen to choose the letter.

(ii) Combination of Floating Keypad and Stick Pointers. The second proposal uses a combination of floating keypad and pointer array. As with the combination of circular keypad and grid mesh pointer, the floating keypad and pointer array can duplicate each other, as shown in Figure 10. At the registration stage, users define the size of the pointer array, choose a real pointer for password input, which makes the others fake pointers, and then register their secret information, password. At the user verification stage, users input their password using the real pointer. As shown in Figures 10(a)–10(c), the robustness of the proposal depends on the size of the pointer array.

4.2. Graphical Password Robust to Emerging Attacks

(i) Layered Pattern-Based Pattern Recognition Scheme. To deal with the problem of imprinted oily residue, our proposed scheme introduces an infinite layered concept called "pattern layer."

As shown in Figure 11, our concept is based on layer level. When users view the screen, they will see nine dots, same as the Google pattern lock. However, when the user is drawing his/her pattern, the pattern can be divided into many layers, which makes it difficult for an attacker to exploit, even if the pattern drawing motion and oily residue are being exposed. Figure 12 shows some example of layered pattern structures.

During pattern registration, putting a part of the pattern on the first layer is not required. Moreover, between two layers on which a pattern is drawn, space is available to put a number of blank (nonpattern) layers. Thus, the amount of combination is decided by user selection, which makes the proposed system more robust than the existing pattern drawing scheme.

(ii) Pattern with 3D Touch Scheme. Another way to make the pattern-based scheme more robust is to introduce 3D touch sensor combination.

The 3D touch sensor embedded in the newest iPhone series was developed in 2015.

According to the 3D touch function sensitive test application, it can recognize various pressure levels. Hence, instead of a number of pattern layers, we could apply 3D touch sensor data to verify the user. Figure 13 shows a pattern drawing with pressure level sensing.

As shown in Figure 13, let us assume that the whole pattern is a "Z" shape, and the first and third strokes are

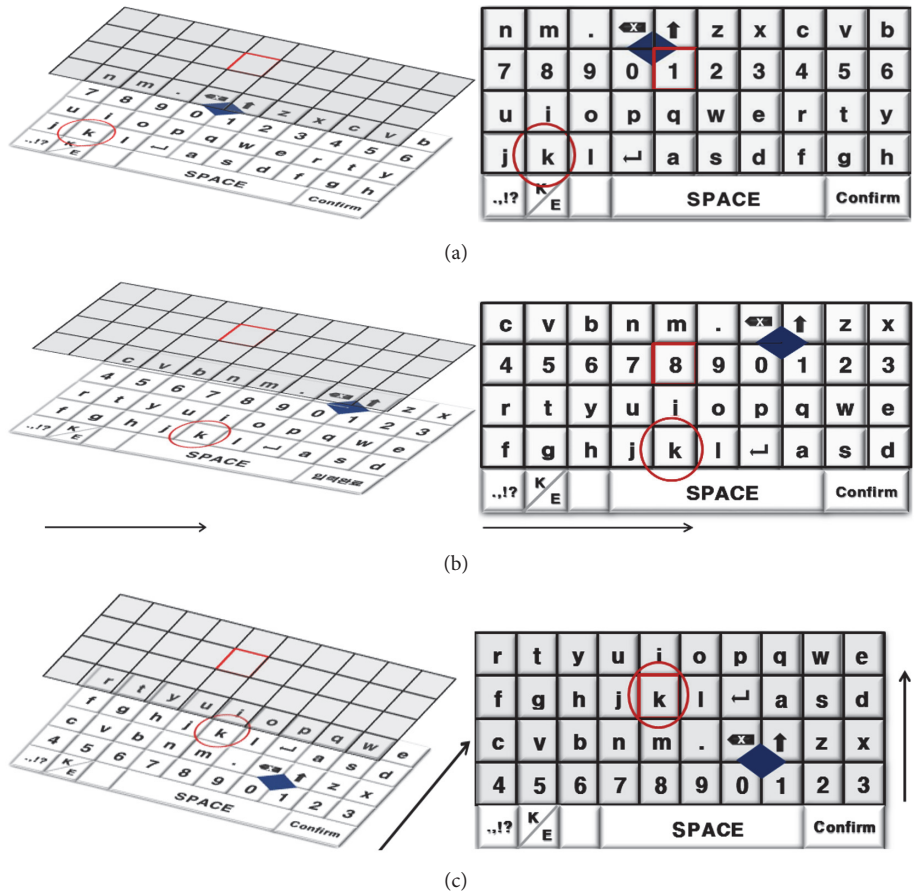


Figure 9: How to use grid pointer for input letter "k."

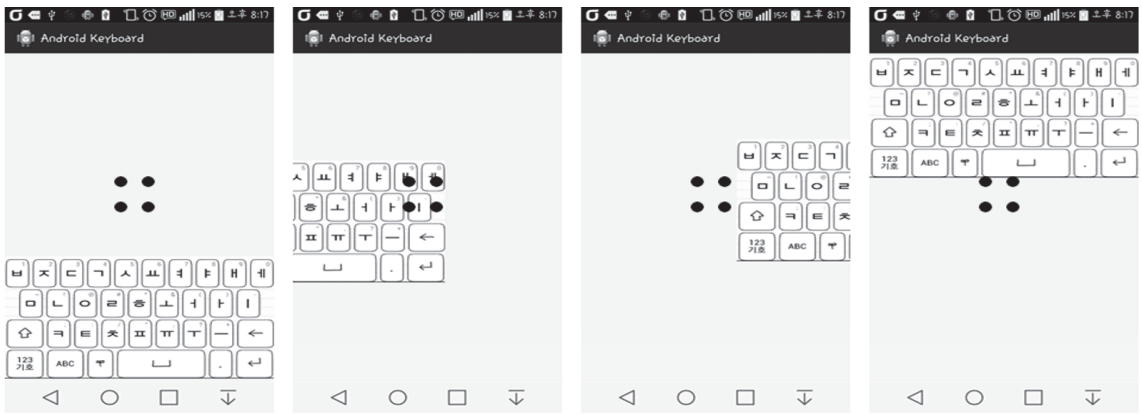
registered with high pressure. In other words, at the registration stage, users registered this pattern with the pressure information that is divided into three sections. The first section is drawn with high pressure, the second section is with low pressure, and the third section is with high pressure. Before the registration, the value of high and low pressure was defined by the user. Then, the user verifies it as shown in Figure 13(b). The pattern in section one was drawn with high finger pressure that is sensed by the pressure sensor in the device screen. During the second pass through with the finger, users do not draw with high pressure. Finally, in the third section, the user draws with high pressure. Therefore, even in the case of shoulder surfing, attackers cannot guess the exact pressure information. Moreover, with the complicated pattern and pressure pattern, it is more difficult to be exposed.

4.3. PIN Code Robust to Emerging Attacks

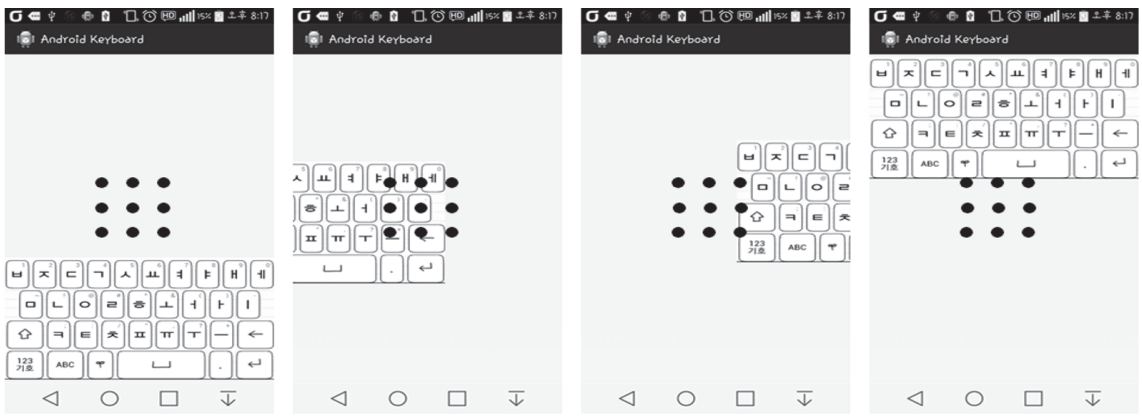
(i) Combination of Image Array and Circular PIN-Based Scheme. This scheme is similar to our proposal for robust text password. Figure 14 shows the proposed scheme robust to emerging attacks. The position of the PIN numbers appearing onscreen is randomly changed at every touch. In addition, the PIN code needs to be input with the user-selected image pointer or pointer sequence. Thus, there may be a need to remember PIN and image pointer or pointer sequence. Each PIN number digit must be connected with its own image pointer, and reusing of image pointer is allowed for every PIN digit. Therefore, the minimum length of our scheme is PIN length + one image pointer. The maximum length is allowed up to PIN length + maximum image pointer length equal to the length of PIN. Like this, our proposal gives various ways to choose between robustness and usability.

4.4. Signature Recognition Robust to Emerging Attacks

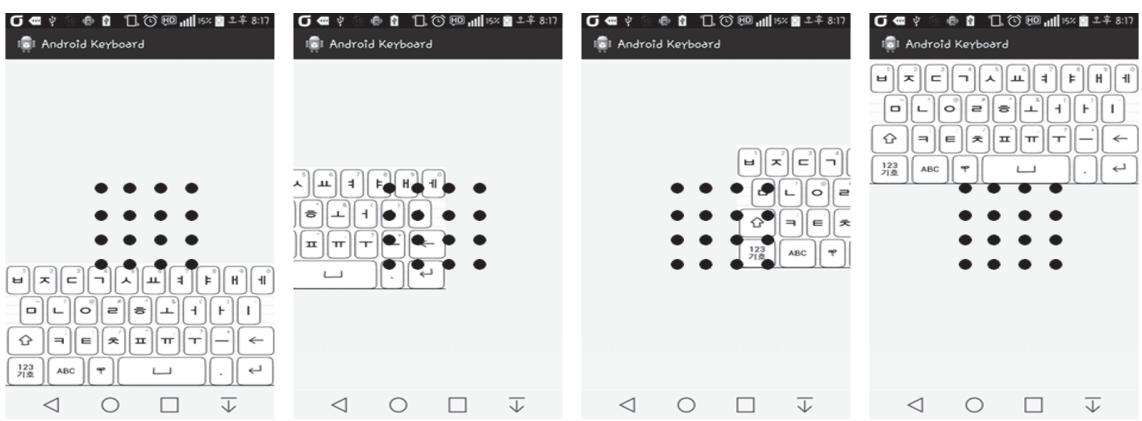
(i) Three-Dimensional Trace with Multisensor-Combination-Based Scheme. In contrast to the 2D-based handwritten signature recognition scheme, the 3D-based scheme traces the user's signature drawing pattern in three-dimensional space. Hence, 3D signature drawing information includes more information that is not detected in 2D-based schemes. Therefore, it is harder to counterfeit than 2D-based schemes. In addition, as shown in Figure 15(a), users may see their signature while writing on the tablet or touchscreen in 2D-based schemes, and this causes vulnerability to emerging attacks. However, as shown in Figure 15(b), no visible tracing action is needed in 3D-based scheme while the users are writing their signature in the air.



(a) 2 × 2 stick pointers



(b) 3 × 3 stick pointers



(c) 4 × 4 stick pointers

Figure 10: Floating keypad and stick pointers layout.

4.5. Fingerprint Recognition Robust to Fake Attacks and Emerging Attacks

(i) Combination of Fingerprint and Heart-Rate Mechanism. Typically, fingerprint authentication techniques are based on a single fingerprint module embedded on the backside or bottom of the device. However, it is vulnerable to fake attacks. Thus, we applied a heart-rate measurement technique for smartphones in order to identify whether a person is real. As shown in Figure 16, the smartphone-based heart-rate measurement technique utilizes a camera and flash. After combining the light and image, the blood flow rate is detected and can then be used for heart-rate calculation. In other words, the heart-rate detection technique can also detect a real person. Thus, combining heart-rate detection and fingerprint module is mandatory to make a robust scheme against fake attacks. Consequently, defending against fake attacks is the best way to defend against emerging attacks

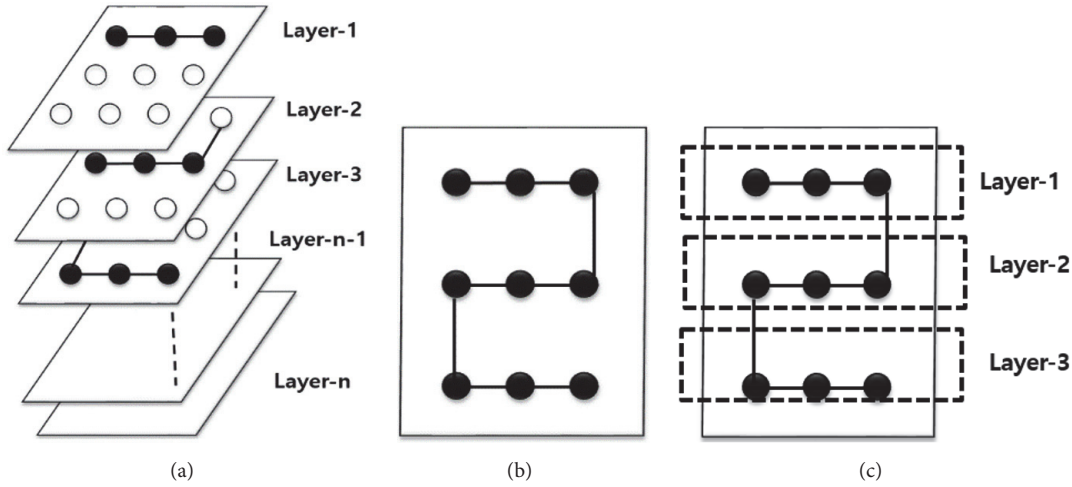


Figure 11: Layered pattern structure: (a) layer level in depth, (b) top view, and (c) top view with layer information.

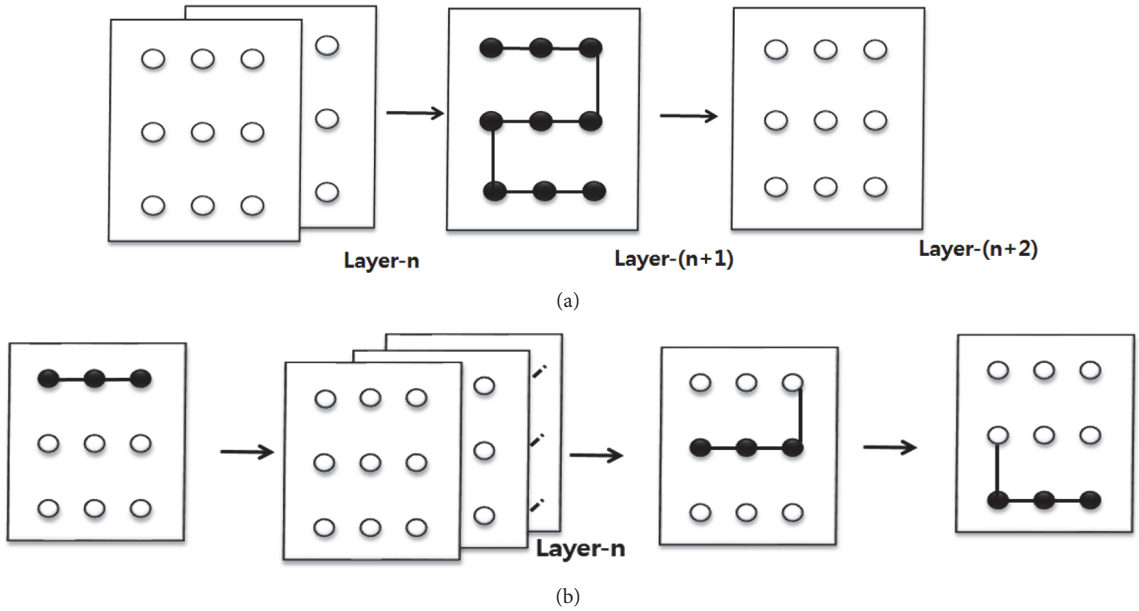


Figure 12: Other layer pattern registration methods: (a) n-layer shift before pattern drawing and (b) n-layer shift during pattern drawing.

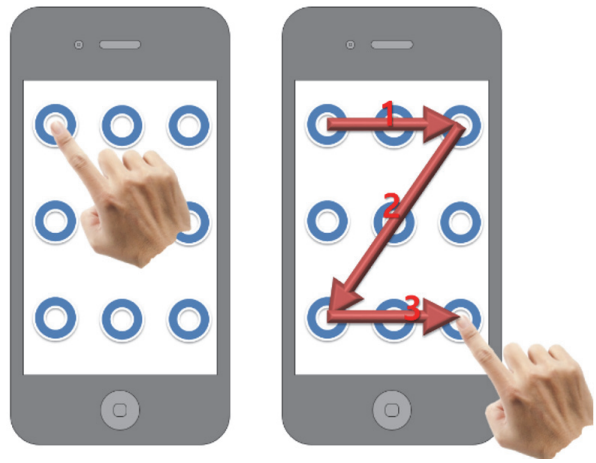
because emerging attacks consider oily residue from the owner's fingerprint only.

5. Comparative Analysis

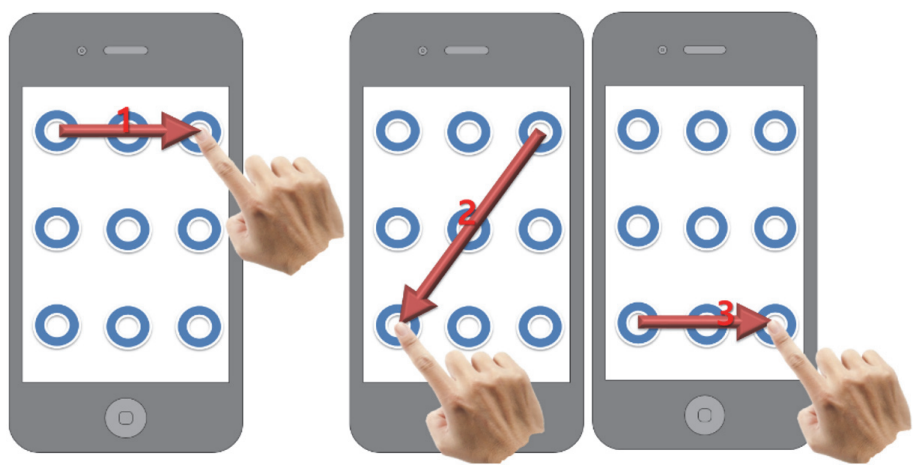
In this section, we analyze each proposed scheme in accordance with privacy and usability compared to the existing schemes. For the evaluation of usability, we simulated and tested the existing and proposed schemes layout structure using MIT App Inventor 2. Then, we checked the schemes against a checklist. In the security comparison, we compared related existing and proposed schemes according to the emerging attack categorization addressed in Section 2.

Table 1 compares the robustness of the existing and proposed schemes against possible emerging attacks. We assigned a number to each of the existing and proposed schemes for convenience: (1) existing text password, (2) proposed combination of circular keypad and grid mesh pointer, (3) proposed combination of floating keypad and stick pointers, (4) existing graphical password, (5) proposed layered pattern-based pattern recognition scheme, (6) proposed pattern with 3D touch scheme, (7) existing PIN code, (8) proposed combination of image array and circular PIN-based scheme, (9) existing signature recognition, (10) proposed 3D trace with multisensor-combination-based scheme, (11) existing fingerprint recognition, and (12) proposed combination of fingerprint and heart-rate mechanism. For the comparison, we used the initials "G," "M," and "B" to denote good, moderate, and bad for each item, respectively.

As shown in Table 1, the proposed schemes have higher security levels than existing schemes except for the sensor item. This is because the proposed scheme did not assume

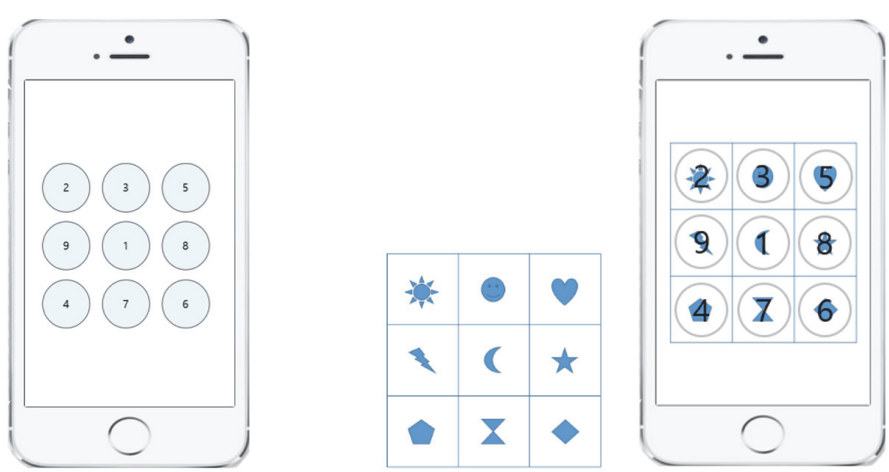


(a) Whole pattern drawing



(b) Example of pattern drawing sequence and pressure differences

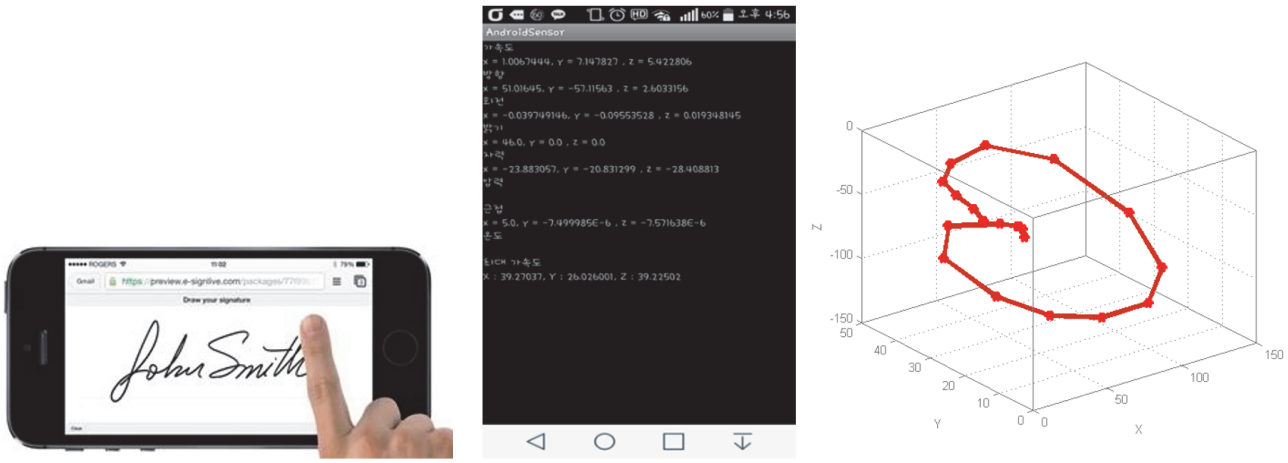
Figure 13: Pattern drawing with pressure.



(a) Normal PIN and image pointer grid set

(b) PIN overlay on image pointer

Figure 14: Proposed PIN scheme.



(a) Example of 2D signature

(b) Example of 3D signature

Figure 15: Proposed 3D signature scheme.

Table 1: Security comparison.

Existing and Proposed Schemes		Possible Emerging Attacks						
		Shoulder surfing		Recording		Hybrid	Smudge	Password guessing with sensors
		single	multiple	single	multiple			
Text	1	B	B	B	B	B	B	B
	2	G	G	G	G	G	G	G
	3	G	M	G	M	M	G	G
Graphical	4	B	B	B	B	B	B	B
	5	G	G	G	M	M	G	M
	6	G	G	G	G	M	G	B
PIN	7	B	B	B	B	B	B	B
	8	G	G	G	M	M	G	G
Signature	9	M	M	B	B	B	M	B
	10	G	G	G	G	G	G	B
Fingerprint	11	G	G	G	G	G	B	G
	12	G	G	G	G	G	G	G

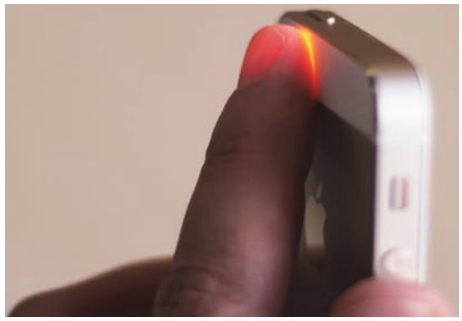


Figure 16: Heart-rate detection with camera and flash.

data protection. Thus, it would give a better result if it considered data encryption. Table 2 shows the usability comparison.

As shown in Table 2, most of the existing schemes have user-friendly characteristics. Because of this point, a number of users do not want to change from their traditional scheme to a more secure scheme. During the comparison, we found that user-behavior-based schemes have many possibilities to expand their application. User behavior is a kind of human biometrics, and behavior is based on accumulated human behavior pattern, which is naturally optimized for many years for each person. This is the reason why the user feels that this kind of scheme is user-friendly.

Finally, Table 3 shows the proposed schemes' advantages and disadvantages from users' experience.

As shown in Table 3, most of the proposed schemes have a low input speed problem and a memory problem. Apart from these problems, participants gave positive reactions. Thus, we expect that the proposed schemes will get more positive results following input speed and memory improvements.

6. Conclusions

In recent times, the mobile device-based market has been increasing continuously. However, opportunities for creating damage have also increased along with the size of the market.

Table 2: Usability comparison.

Existing and proposed schemes		Possible metrics classification					
		Easy registration	Typing speed	Typing error	Easy to understand	Easy to remember	Login speed
Text	1	G	G	M	G	G	G
	2	G	M	G	M	M	M
	3	G	M	G	M	M	M
Graphical	4	G	G	G	G	G	G
	5	M	B	M	M	B	B
	6	G	G	G	G	G	G
PIN	7	G	G	G	G	G	G
	8	M	M	G	M	M	M
Signature	9	G	G	G	G	G	G
	10	G	G	M	G	G	G
Fingerprint	11	G	G	G	G	G	G
	12	G	G	G	G	G	G

Table 3: Comparison table of proposed schemes.

Category	Proposed	Advantages	Disadvantages
Text password	Circular keypad with grid mesh pointer	Display no password on screen Very hard to estimate or trace	Low input speed Remember longer password information
	Floating keypad with stick pointers	Display no password on screen Hard to estimate or trace	Low input speed Remember longer password information
Graphical password	Layered pattern	Hard to estimate or trace	Low drawing speed Remember more information about layers Need more functions for moving between the layers
	Pattern with 3D touch	Very hard to estimate or trace Higher speed compared to layered pattern	Remember more information about section pressures Need 3D sensor-embedded device
PIN	Image array and circular PIN combination	Randomly repositioned PIN code Hard to estimate or trace	Low input speed Remember more information about password/image combination
Signature	Sensor-based 3D signature	High speed No need for optional device	Need higher algorithmic sensible detection process
Fingerprint	Fingerprint with heart-rate detection	Robust to fake attack No need for optional device	Depending on mobile device structure

This paper focused on emerging security threats targeting mobile device structure defects and human errors and highlighted the vulnerabilities in existing schemes. Subsequently, schemes were proposed in accordance with each category and were shown to exhibit robust results against emerging attacks compared to existing schemes. However, although our schemes are very secure, their usability is still insufficient. Therefore, in future work, we plan to modify our schemes to make them more acceptable to users. Our objective is to improve the schemes up to at least the existing user experience level of the well-known schemes. In addition, we determined that human biometric-based schemes are more reasonable, user-friendly, and robust to verify users. Thus, we also plan to develop a novel human biometric-based scheme.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the Natural Science Foundation of Jiangsu Province (Grant BK20160287) and in part by the Key Research and Development Program of Jiangsu (Grants BE2017071 and BE2017647). It was also supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (no. 2017R1E1A1A01077913) and by the MIST (Ministry of Science & ICT), Korea, under the National Program for Excellence in SW supervised by the IITP (Institute for Information & Communications Technology Promotion) (2017-0-00137).

References

[1] T. Wang, Y. Chen, M. Zhang, J. Chen, and H. Snoussi, "Internal transfer learning for improving performance in human action

recognition for small datasets," *IEEE Access*, vol. 5, pp. 17627–17633, 2017.

[2] T. Wang and H. Snoussi, "Detection of abnormal visual events via global optical flow orientation histogram," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 6, pp. 988–998, 2014.

[3] https://en.wikipedia.org/wiki/Brute-force_attack.

[4] I. Kim, "Keypad against brute force attacks on smartphones," *IET Information Security*, vol. 6, no. 2, pp. 71–76, 2012.

[5] https://en.wikipedia.org/wiki/Dictionary_attack.

[6] A. K. Kyaw, F. Sioquim, and J. Joseph, "Dictionary attack on wordpress: security and forensic analysis," in *Proceedings of the 2nd International Conference on Information Security and Cyber Forensics (InfoSec '15)*, pp. 158–164, November 2015.

[7] H. Shin, D. Kim, and J. Hur, "Secure pattern-based authentication against shoulder surfing attack in smart devices," in *Proceedings of the 7th International Conference on Ubiquitous and Future Networks (ICUFN '15)*, pp. 13–18, July 2015.

[8] https://en.wikipedia.org/wiki/Replay_attack.

[9] https://www.ftc.gov/reports/spyware-workshop-monitoring-software-your-personal-computer-spyware-adware-other-software.

[10] E. Darbanian and G. D. Fard, "A graphical password against spyware and shoulder-surfing attacks," in *Proceedings of the 20th International Symposium on Computer Science and Software Engineering (CSSE '15)*, pp. 1–6, August 2015.

[11] A. Adams and M. A. Sasse, "Users are not the enemy: why users comprise computer security mechanisms and how to take remedial measures," *Communications of the ACM*, vol. 42, no. 12, pp. 41–46, 1999.

[12] T.-S. Wu, M.-L. Lee, H.-Y. Lin, and C.-Y. Wang, "Shoulder-surfing-proof graphical password authentication scheme," *International Journal of Information Security*, vol. 13, no. 3, pp. 245–254, 2014.

[13] J. Bonneau, S. Preibusch, and R. Anderson, "A birthday present every eleven wallets? The security of customer-chosen banking PINs," in *Financial Cryptography (LNCS)*, pp. 25–40, Springer, New York, NY, USA, 2012.

[14] https://en.wikipedia.org/wiki/Personal_identification_number.

[15] https://en.wikipedia.org/wiki/One-time_password.

[16] J. A. Vila, J. Serna-Olvera, L. Fernández, M. Medina, and A. Sfakianakis, "A professional view on ebanking authentication: challenges and recommendations," in *Proceedings of the 9th International Conference on Information Assurance and Security (IAS '13)*, pp. 43–48, December 2013.

[17] G. Perna, "How does screen size effect viewer's response to various types of media?" screenMediaUCSD, March 2015, http://screenmediaucsd.wikispaces.com/-/Term%20Wiki%20WI15/Team%2016/How+does+screen+size+effect+viewer%27+response+to+various+types+of+media%3F.

[18] H. Kim, H. Seo, Y. Lee, and T. Park, "Implementation of secure virtual financial keypad for shoulder surfing attack," *Korea Institute of Information Security and Cryptography*, vol. 23, no. 6, pp. 21–29, 2013.

[19] H.-M. Sun, S.-T. Chen, J.-H. Yeh, and C.-Y. Cheng, "A shoulder surfing resistant graphical authentication system," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 2, pp. 180–193, 2018.

[20] T. Takada, "FakePointer: an authentication scheme for improving security against peeping attacks using video cameras," in *Proceedings of the 2nd International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM '08)*, pp. 395–400, Valencia, Spain, September 2008.

[21] A. J. Aviv, K. Gibson, E. Mossop, M. Blaze, and J. M. Smith, "Smudge attacks on smartphone touch screens," in *Proceedings of the USENIX Conference on Offensive Technologies*, pp. 1–7, 2010.

[22] S. Shankland, Reverse smudge engineering foils android unlock security, http://www.cnet.com/news/reverse-smudge-engineering-foils-android-unlock-security/.

[23] L. Cai and H. Chen, "TouchLogger: inferring keystrokes on touch screen from smartphone motion," in *Proceedings of the 6th USENIX Conference on Hot Topics in Security*, 2011.

[24] E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. R. Choudhury, "Tapprints: your finger taps have fingerprints," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys '12)*, pp. 323–336, Ambleside, UK, June 2012.

[25] P. S. Teh, A. B. J. Teoh, C. Tee, and T. S. Ong, "Keystroke dynamics in password authentication enhancement," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8618–8627, 2010.

[26] C. E. Larsen, R. Trip, and C. R. Johnson, "Direct, Gesture-based Actions from Device's Lock Screen," US 8136053 B1, http://www.google.com/patents/US8136053.

[27] N. L. Clarke, S. M. Furnell, P. M. Rodwell, and P. L. Reynolds, "Acceptance of subscriber authentication methods for mobile telephony devices," *Computers & Security*, vol. 21, no. 3, pp. 220–228, 2002.

[28] A. Das, O. K. Manyam, M. Tapaswi, and V. Taranalli, "Multilingual spoken-password based user authentication in emerging economies using cellular phone networks," in *Proceedings of the IEEE Workshop on Spoken Language Technology (SLT '08)*, pp. 5–8, IEEE, December 2008.

[29] K.-H. Kamer, A. Yuksel, A. Jahnbekam, M. Roshan-del, and D. Skirpo, "MagiSign: user identification/authentication based on 3D around device magnetic signatures," in *Proceedings of Ubicomm*, pp. 31–34, 2010.

[30] H. Ketabdar, P. Moghadam, B. Naderi, and M. Roshandel, "Magnetic signatures in air for mobile devices," in *Proceedings of the 14th ACM International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '12)*, pp. 185–188, September 2012.

[31] F. Li, N. Clarke, M. Papadaki, and P. Dowland, "Behaviour profiling for transparent authentication for mobile devices," in *Proceedings of the 10th European Conference on Information Warfare and Security (ECIW '11)*, pp. 307–314, July 2011.

[32] Deepnet Security, "Keystroke Recognition," http://www.deepnetsecurity.com/authenticators/biometrics/typesense/.

[33] J. Kil, "Iphone and galaxy phone penetrated by fake fingerprint," http://www.etnews.com/20160217000327.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 7670939, 11 pages
https://doi.org/10.1155/2018/7670939

Research Article

Jammer Localization in Multihop Wireless Networks Based on Gravitational Search

Tongxiang Wang[1], **Xianglin Wei**[2], **Jianhua Fan**[2] **and Tao Liang**[2]

[1]*Graduate School, PLA Army Engineering University, Nanjing 210007, China*
[2]*Nanjing Telecommunication Technology Research Institute, Nanjing 210007, China*

Correspondence should be addressed to Jianhua Fan; 237008273@qq.com

Received 15 January 2018; Accepted 20 March 2018; Published 6 May 2018

Academic Editor: Ilsun You

Multihop Wireless Networks (MHWNs) can be easily attacked by the jammer for their shared nature and open access to the wireless medium. The jamming attack may prevent the normal communication through occupying the same wireless channel of legal nodes. It is critical to locate the jammer accurately, which may provide necessary message for the implementation of antijamming mechanisms. However, current range-free methods are sensitive to the distribution of nodes and parameters of the jammer. In order to improve the localization accuracy, this article proposes a jammer localization method based on Gravitational Search Algorithm (GSA), which is a heuristic optimization evolutionary algorithm based on Newton's law of universal gravitation and mass interactions. At first, the initial particles are selected randomly from the jammed area. Then, the fitness function is designed based on range-free method. At each iteration, the mass and position of the particles are updated. Finally, the position of particle with the maximum mass is considered as the estimated jammer's position. A series of simulations are conducted to evaluate our proposed algorithms and the simulation results show that the GSA-based localization algorithm outperforms many state-of-the-art algorithms.

1. Introduction

Multihop Wireless Networks (MHWNs) face various security problems due to their shared nature and open access to wireless mediums. Among all the security threats to the MHWNs, one typical case of attacks is jamming attack, which usually emits useless radio signal to disrupt normal communications between wireless devices by occupying the wireless channel or destroying the coupling of protocols with one or many low-end simple off-the-shelf wireless devices [1, 2]. For instance, different from interferences among wireless nodes [3], jamming attack can break down the MAC protocols by sending fabricated ACK or CTS packets to the wireless channel. Generally speaking, jamming attack can be initiated from different protocol layers and decreases the network performance significantly through limited resource consumption, which makes it be widely employed by adversaries.

In order to mitigate the impact of jamming attack and restore the normal communications, a series of anti-jamming countermeasures have been proposed from multiple network layers, such as channel-hopping, secure routing, and spatial retreat [4–7]. However, these anti-jamming strategies mainly provide useful approaches to avoid or evade an attack in order to maintain the normal operation of wireless networks. Although the negative impact of jamming attack can be mitigated, the networks can only adjust themselves passively without utilizing the information of jamming. Moreover, when conducting the anti-jamming measures, the constraints of wireless devices including limited memory and energy supply and low computation capabilities must be considered.

Besides these passive anti-jamming measures, another way is to locate the jammer and obtain the position information of jammers, which makes it possible to eliminate the jammer from the networks by physical methods or manual ways. Actually, the position information of jammers may allow better deployment of wireless devices and provide useful information when designing MAC or routing protocols.

Up to now, jammer localization has been widely investigated and a number of localization algorithms have been

proposed. In conclusion, exiting jammer localization algorithms can be divided into range-based methods and range-free ones. Range-based algorithms need to estimate the parameters of wireless channel in advance and calculate the relative distance between nodes and the jammer. Although some typical models of wireless channel have been proposed, the parameters of wireless channel can be hardly estimated accurately in real scenario. Besides, the performance of range-free algorithms can be easily affected by the distribution of nodes and the jammer's parameters.

In order to reduce the sensitivity of range-free algorithms and improve the localization accuracy, a robust jammer localization algorithm based on Gravitational Search Algorithm (GSA) is proposed in this paper. At first, several related models, that is, network model, jamming model, and communication model, are illustrated. Then, the GSA-based jammer localization is presented, which mainly consists of selection of initial particles, determination of fitness function, resultant force calculation, and parameters update. At last, a series of simulations are conducted to evaluate the performance of our proposed algorithm. Compared with many state-of-the-art jammer localization algorithms, our algorithm performs better in many different scenarios with different parameter settings.

The architecture of this article is organized as follows. Section 2 summarizes related work. Several related models are introduced in Section 3. Section 4 presents our jammer localization strategy based on GSA in detail. Simulation experiments and results are described in Section 5. The main work is concluded in the last section and some discussions on the future work are highlighted.

2. Related Work

Over the past few years, Xu et al. conducted a series of researches on the jamming attack and four basic approaches of jamming attack were proposed [8], which were defined as constant jamming, random jamming, proactive jamming, and reactive jamming. Wei et al. provided a comprehensive survey of the major works done in the field of jammer localization for MHWN [9].

Range-free localization algorithms utilize the geometric knowledge of the jammed area to locate the jammer. Wang and Zheng took the weighted factor determined by the relative position between jammer and node into consideration when modifying the Centroid Localization (CL) [10] and presented Weighted Centroid Localization (WCL). Liu et al. put forward the Virtual Force Iterative Localization (VFIL) to locate the jammer [11]. At first, the jammed area and jamming range were estimated by VFIL. Then, the estimated position of the jammer was amended iteratively in order to cover the most jammed nodes. Sun et al. computed the convex hull that was determined by the boundary nodes to locate the jammer [12]. The minimum circumscribed circle was achieved based on the convex hull and the center of it is the estimated jammer's location. Similarly, α-hull was adopted by Zhang et al. to obtain circumcircle of the jammed area and then the least square circle was formulated to estimate jammer's location [13]. In addition, Wei et al. also made the research on the collaborative mobile jammer tracking in MHWN and the jammer is located based on multilateral localization method [14]. For multi-jammers scenario, Cheng et al. put forward the M-clusters and X-ray to estimate jammers' positions, respectively [15]. Wang et al. proposed the k-mean cluster algorithm based on the neighbor nodes' information to estimate the positions of the jammers [16].

The relationship between jammer and node is established based on wireless channel model to locate the jammer for range-based localization algorithms. Pelechrinis et al. pointed out that Packet Delivery Rate (PDR) decreased with increasing distance between node and the jammer [17]. So value of PDR could be used to indicate the influence that the jammer had on the node. They proposed a light distributed jammer localization algorithm based on PDR and the node would select the node with minimum PDR from its neighbor nodes as the next hop. Liu et al. proposed the jammer localization algorithm based on the nodes' hearing range [18], which is defined as the maximum distance for the node that can successfully decode the signal generated from other nodes. Wang et al. put forward the scheme to locate the jammer based on the combination of PDR gradient descent and power adaptive technique [19]. The power would increase at the termination node of PDR gradient descent and the localization accuracy was improved a lot compared to that of PDR gradient descent.

3. System Models and Problem Formulation

This section analyzes the impact of jamming on the legal communication link and introduces several related models. The nodes in the jammed network can be divided into three categories based on the impact of jamming, that is, unaffected nodes, boundary nodes, and jammed nodes.

3.1. Impact of Jamming. According to the characteristic of wireless communication, the signals cannot be decoded correctly if the received SNR is lower than a certain threshold. Assume that the interference among nodes has been avoided through specific MAC or network protocols, such as TDMA and 802.11 DCF. Thus, the overall interference mainly includes the background noise for nonjamming scenes and the background noise and jamming signal for jamming scenes. For the transmitter i and receiver j, the received SNR_{ij} of node j is

$$\mathrm{SNR}_{ij} = \frac{P_{ij}}{P_N + P_{jR}}, \tag{1}$$

where R represents the jammer and P_{jR} is the received jamming power at node j. P_{ij} and P_N represent the received power of node j and the power of background noise, respectively. The state of link between node i and node j is defined as l_{ij}:

$$l_{ij} = \begin{cases} 0, & \mathrm{SNR}_{ij} \le \gamma_0 \\ 1, & \mathrm{SNR}_{ij} > \gamma_0, \end{cases} \tag{2}$$

where γ_0 is the received SNR threshold for all the nodes. $l_{ij} = 1$ denotes the normal communication between node

i and node j. The communication links among nodes are bidirectional and the link between node i and node j is considered to be connected when both the conditions $l_{ij} = 1$ and $l_{ji} = 1$ are satisfied.

3.2. Related Models

3.2.1. Network Model. The characteristics of MHWN model considered in this article mainly include the following.

Multihop and Stationary. Once deployed, the position of MHWN node remains unchanged and the nodes communicate with each other through multihop fashion. The nodes are assumed to be time-synchronous, which can be achieved by the clock calibration after initial deployment.

Location-Aware. The MHWN nodes can be aware of their own locations and their neighbors' locations through GPS or specific location-aware algorithms and many applications also require the location of nodes in order to provide specific services. Assume that the locations of nodes have been obtained after initial deployment.

Neighbor-Aware. Each node can store its neighbors' information and update a neighbor list at regular intervals. The list can be achieved by several routing protocols, such as AODV and DSR.

Besides, each node is equipped with omnidirectional antenna and transmits signals with the same power. In other words, the nodes are homogeneous in MHWN.

3.2.2. Jamming Model. The jammer considered in this article remains static and the jamming power remains unchanged. Besides, the constant jammer equipped with omnidirectional antenna is adopted in this article, which transmits RF signals consistently.

3.2.3. Node Model. The nodes deployed in the MHWN randomly can be divided into jammed ones, boundary nodes, and unaffected ones according to different degree of jamming produced by the jammer:

(i) Unaffected node: a node is determined to be unaffected if it can receive packets from all of its neighbors after the appearance of the jammer

(ii) Boundary node: the node is considered as a boundary node if it loses some of its neighbors, while it can still communicate with part of the unaffected nodes

(iii) Jammed node: the jammed node is defined as the node that cannot receive any message from all the unaffected nodes and boundary nodes

3.2.4. Wireless Channel Model. Typical wireless channel models mainly include free-space propagation model, shadow-fading model, and exponential-fading model [20–22]. The shadow-fading model is adopted here to model the small-scale fading circumstance. If the receiver locates at the distance d from the transmitter, the received power ($PL(d)$) is

$$PL(d) = PL(d_0) - 10\eta \log\left(\frac{d}{d_0}\right) + X_\sigma, \quad (3)$$

where $PL(d_0)$ represents the received power at specific distance d_0 and η is the fading index. X_σ is the Gauss random variable with zero mean and variance σ^2.

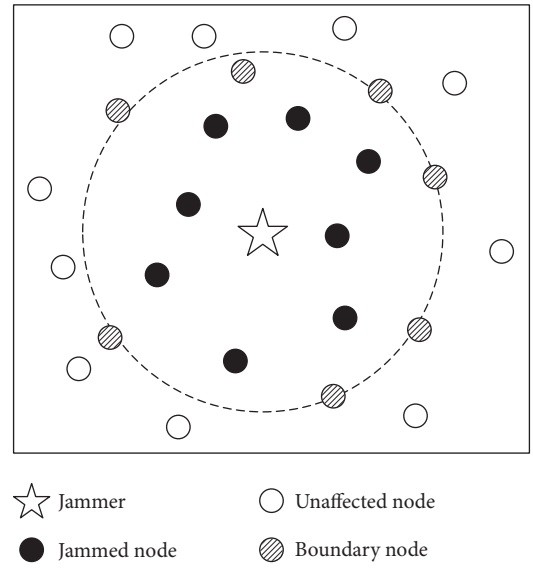


Figure 1: Typical scenario of jammed network.

3.3. Problem Formulation. A typical jammed network scenario is illustrated in Figure 1. We aim at locating the jammer under the above settings by using the jamming information. To achieve this goal, several challenges need be solved and we present our basic ideas here. At first, each node should determine its state based on the neighbors number, received SNR, and so forth. Then, we need to decide the jamming information that could be collected by wireless nodes, such as sensor node. Besides, the information can be used to detect the jammer's existence. At last, an efficient localization algorithm needs to be carefully designed considering both the complexity and accuracy.

4. Algorithm Description

4.1. GSA Principle. The Gravitational Search Algorithm was proposed by Rashedi et al. in 2009 [23] and the searching progress can be carried out by interaction among particles. In GSA, the particle's position represents the solution of the problem. At first, the initial solution is obtained through the feasible region and the particles' mass is calculated by the fitness function. Then, the interaction among particles is used to update their mass and positions. The particles will move to the particles with larger mass, which represents the better solutions. At last, the particle with the largest mass is considered as the best solution.

GSA has been widely employed in data mining, parameter identification, and multi-objectives decision and achieved good performance. In the problem of jammer localization, the estimated jammer's position is the solution of the problem, which is consistent with the solution of GSA. Therefore, it is feasible to employ the GSA in the jammer localization, which is the basis of our paper.

In order to achieve this target, the challenges and main work in our paper mainly include the following:

(i) Mapping of particles' positions and jammer's estimated location: in our proposed algorithm, the particle's position represents the jammer's estimated location. With the progress of iteration process, the particles' positions are updated. At the end of the iteration, the position of the particle with the maximum mass is considered as the final estimated jammer's location

(ii) Selection of initial particles: according to the characteristics of jamming area, the jammer is supposed to be located in its inside. In the initial step of GSA, a specific number of particles are distributed in the jammed area randomly

(iii) Calculation of fitness function: the jammer is equipped with omnidirectional antenna and the jammed area is about a circle. Thus, the distances between the jammer and boundary nodes that located in the boundary of the jammed area are similar. The fitness function can be determined by the variance of distances between the boundary nodes and the jammer

4.2. Preliminary. Generally speaking, the affected nodes (including boundary nodes and jammed nodes) can be used to reflect the existence of jamming attack. Several jamming detection methods have been proposed based on affected nodes' collected jamming information, such as received signal strength and carrier sensing time [8]. However, jamming detection is not our main work and we pay our attention to the localization process, which would be conducted after detecting jamming attack. Assume that each wireless node in the network can work normally and detect the existence of jamming attack correctly.

Before conducting the localization process, each node should determine its state based on the jamming information. If every node tries to communicate with all of its neighbor nodes, the network load would increase a lot. In order to determine its state efficiently, each node maintains its neighbors number and records each neighbor's SNR. Then, it will determine its own state according to Algorithm 1, where parameters b, c, and d are determined by specific network condition.

4.3. Information Collection. Assume that there is a locating node, which is in charge of jammer localization, in the wireless network chosen from the unaffected nodes through some kind of voting algorithm. To realize the localization process, it has to firstly collect information from wireless nodes and then execute the localization algorithm.

```
(1) if a node does not detect jamming attack then
(2)     This node is an unaffected node;
(3) else
(4)     if this node does not lose any neighbor and its
        received SNRs from most of its neighbors' do not
        decrease more than a percent then
(5)         This node is an unaffected node;
(6)     else
(7)         if this node loses more than b percent of its
            neighbors then
(8)             This node is a jammed node;
(9)         else
(10)            if more than c percent of its neighbors' SNRs
                decrease more than d percent then
(11)                This node is a jammed node;
(12)            else
(13)                This node is a boundary node;
(14)            end if
(15)        end if
(16)    end if
(17) end if
```

Algorithm 1: Node state determination algorithm.

Node_ID	*Destination_node_ID*	*RJSS*	*Node_position*

Figure 2: Frame structure of information collection.

Based on the analysis in the above section, we have to design an information-collecting protocol to gather necessary RJSS values and other related information of nodes while reducing the transmission overhead introduced by this process, since normal communication among wireless nodes may be damaged or even disrupted by jammer. To achieve this goal, a node is required to report its RJSS values to the locating node if and only if it is determined to be a boundary node.

Each boundary node will send its collected RJSS values and position to the locating node. The basic structure of the reporting packet is shown in Figure 2, where *Node_ID* and *Destination_node_ID* represent the IDs of the boundary node and locating node, respectively. *RJSS* and *Node_position* are the reported RJSS values and the coordinate values of the boundary node, respectively.

4.4. Jammer Localization Based on GSA. The GSA-based jammer localization can be divided into four steps according to GSA principle: selection of initial particles, determination of fitness function, resultant force calculation, and parameters update.

4.4.1. Selection of the Initial Particles. The position of particle represents the solution of the problem based on GSA principle. According to the jamming characteristics, the jammer must be within the jammed area. Therefore, the initial particles, which are represented by triangle in Figure 3,

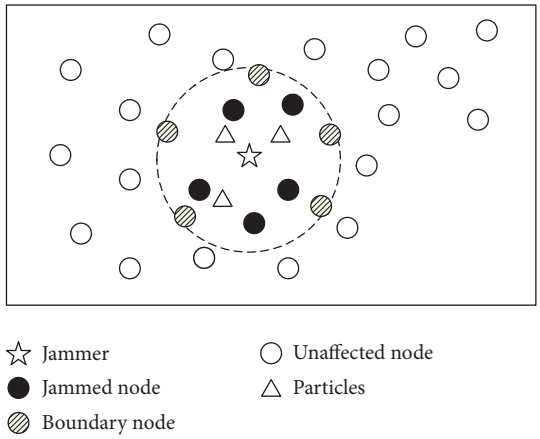


Figure 3: Distribution of initial particles.

should be selected within the jammed area. Besides, the velocity and acceleration of the particles are set as zero at the first iteration.

In order to improve the uniformity of the initial particles' distribution, the backward learning method is adopted to promote the performance of the algorithm. Assume that N particles are deployed in a D-dimensional space and the position of particle i is represented as $X_i = \{x_i^1, x_i^2, \ldots, x_i^k, \ldots, x_i^D\}$, where $x_i^k \in [a_k, b_k]$ and a_k and b_k are the lower bound and upper bound in the kth dimension. Then, the opposite particle of x_i^k is $\tilde{x}_i^k = a_k + b_k - x_i^k$. At last, the initial particles and opposite particles are all added to the final particle set. The particles (including initial particles and opposite particles) are deployed in the jammed area formulated by the jammed nodes and the acceleration and velocity of the particles are all set to be zero at the first iteration.

4.4.2. Determination of the Fitness Function. The fitness function is designed to evaluate the performance of estimated jammer's position. In the single-jammer scenario, the jammed area is similar to a circle and the distance between boundary nodes and the jammer is approximately equal to the jamming radius. Therefore, we obtain the boundary nodes based on convex hull at first and then the variance of distances between the boundary nodes and the jammer is calculated, which is defined as the fitness function. Suppose that there are N particles in the MHWN; the position of particle i is defined as $X_i = (X_{ix}, X_{iy})$, $i = 1, 2, \ldots, N$. Thus, the fitness function at the t iteration for particle i is

$$\begin{aligned} \mathrm{fit}_i(t) &= \frac{1}{M}\sum_{j=1}^{M}\left(d_{ij}(t) - \overline{d}_i(t)\right)^2, \quad i = 1, 2, \ldots, N, \\ d_{ij}(t) &= \sqrt{\left(X_{ix}(t) - B_{jx}\right)^2 + \left(X_{iy}(t) - B_{jy}\right)^2}, \\ \overline{d}_i(t) &= \frac{1}{M}\sum_{j=1}^{M}\left(d_{ij}(t)\right), \end{aligned} \tag{4}$$

where M is the number of boundary nodes. B_{jx} and B_{jy} represent the x and y coordinate values for the boundary node j, respectively. $d_{ij}(t)$ is defined as the Euclidean distance between particle i and boundary node j. $\overline{d}_i(t)$ is the average Euclidean distance.

4.4.3. Resultant Force Calculation. The mass of each particle is determined by its fitness, which can be calculated by

$$\begin{aligned} m_i(t) &= \frac{\mathrm{fit}_i(t) - \mathrm{worst}(t)}{\mathrm{best}(t) - \mathrm{worst}(t)}, \\ M_i(t) &= \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}, \end{aligned} \tag{5}$$

where $\mathrm{fit}_i(t)$ is the fitness value for particle i at the t iteration. $\mathrm{best}(t)$ and $\mathrm{worst}(t)$ represent the best value and worst value of the particles at the t iteration. $m_i(t)$ and $M_i(t)$ are the mass and normalized mass for particle i at the t iteration, respectively.

The gravitation for particle i and particle j in d dimension can be calculated by

$$F_{ij}^d = G(t)\frac{M_i(t) \times M_j(t)}{d_{ij}^2(t) + \varepsilon}\left(x_j^d(t) - x_i^d(t)\right), \tag{6}$$

where $x_i^d(t)$ and $x_j^d(t)$ represent the positions of particles i and j at the t iteration, respectively. $d_{ij}(t)$ represents the Euclidean distance between particle i and particle j. ε is a small constant and $G(t)$ indicates the gravity coefficient. The resultant force $F_i^d(t)$ for particle i in d dimension is the resultant force generated from all the other particles:

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} \mathrm{rand}_j F_{ij}^d(t), \tag{7}$$

where rand_j is a random number in $[0, 1]$.

4.4.4. Parameters Update

Update the Gravity Coefficient. According to the GSA principle, the gravity coefficient that can be calculated by formulation (8) decreases with the time.

$$G(t) = G_0 \times e^{-\alpha(t/T)}, \tag{8}$$

where α is the time constant, T is the maximum number of iterations, and G_0 is the initial value of gravity coefficient.

Update the Acceleration, Velocity, and Position. The acceleration of particle i in d dimension at the t iteration is

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)}, \tag{9}$$

where $M_i(t)$ is the normalized mass for particle i. The velocity and position for particle i are updated by

$$\begin{aligned} v_i^d(t+1) &= \mathrm{rand}_i \times v_i^d(t) + a_i^d(t), \\ x_i^d(t+1) &= x_i^d(t) + v_i^d(t+1), \end{aligned} \tag{10}$$

where $v_i^d(t)$ and $x_i^d(t)$ are the velocity and position, respectively, for particle i in d dimension at the t iteration.

At last, the GSA will be terminated until the time of iterations reaches the threshold and, otherwise, the mass, velocity, and position for all the particles would be updated at the next iteration.

4.5. Discussions. In our proposed GSA-based jammer localization algorithm, the fading index of wireless channel and jamming power should be estimated exactly in order to calculate the estimated RJSS values. Due to the fact that the received signal's power and nodes' locations have already been obtained, the fading index of wireless channel can be estimated by the communication among nodes. In this paper, the fading index is assumed to be estimated accurately. However, the jamming power cannot be derived or estimated directly. In order to solve this problem, the jamming power is assumed to be chosen from a series of discrete values, which is also the assumption made in [24]. The proposed localization algorithm will be conducted under these discrete jamming powers and the best position and optimal value under each jamming power will be recorded. In the end, the global optimal value chosen from all the optimal values of different power levels will be chosen and its corresponding jamming power is considered as the real jamming power.

4.6. Pseudocode. The pseudocode of GSA-based localization algorithm is illustrated in Algorithm 2. Step (1) is the initialization of related parameters, that is, number of iterations, number of particles, the acceleration, velocity and position of the particles, the gravity coefficient, and the time constant used to update the gravity coefficient. N particles that present the initial solutions are randomly selected in the jammed area in Step (3). Step (4) calculates the fitness value for each particle and the best value is updated according to the fitness values. The mass and the resultant force will be calculated according to Newton's second law in Steps (6) and (7). The gravity coefficient and acceleration, velocity, and position of the particle will be updated for each iteration in Step (8). After the iterations, the position of particle with maximum mass is considered as the estimated jammer's position.

Complexity Analysis. The time complexity of our proposed algorithm is $O(TN^2)$. T is the number of iterations and N is the number of particles.

Input: the state and position of MHWN nodes
Output: the estimated jammer's position
(1) **Initialize** number of iterations (T), number of particles (N), acceleration, velocity and position of the particle, gravity coefficient and time constant α.;
(2) **for** $t = 1 : T$ **do**
(3) Random select N particles in the jammed area;
(4) Calculate the fitness function for each particle;
(5) Save the best value at this iteration and update the global best value;
(6) Calculate the mass and normalized mass of the particles;
(7) Calculate the resultant force in each dimension for all the particles;
(8) Update the gravity coefficient and acceleration, velocity, position of the particle.
(9) **end for**

Algorithm 2: GSA-based localization algorithm.

Table 1: Simulation parameters.

Parameter	Meaning	Value
M	Simulation times	200
Q	Number of MHWN nodes	400
L	Radius of MHWN	400 m
P_J	Transmitting power of jammer	10 mW
P_T	Transmitting power of node	10 mW
P_N	Power of noise	−60 dBm
n	Fading exponent	2
G_r	Gain of the transmitting antenna	1
G_t	Gain of the receiving antenna	1
N	Number of particles	50
G_0	Initial value of gravity coefficient	100
α	Time constant	20
T	Number of iterations	50

5. Simulation Experiments

5.1. Parameters Setting and Benchmark

5.1.1. Parameters Setting. The basic MHWN is established in an area of $L * L$ square meters and the number of nodes is Q. The jammer equipped with omnidirectional antenna is deployed in the center of the network with coordinate (200, 200). Other related parameters are represented in Table 1.

5.1.2. Benchmark. The algorithms used for comparison with our proposed algorithm are CL, WCL, VFIL, and DCL. These four algorithms are the common localization algorithms usually used for comparison in most of the papers and the comparison results are convincing.

The average error (e) is adopted to measure the performance of our proposed algorithm, which can be calculated by

$$e = \frac{1}{M}\sum_{i=1}^{M}\left(\left\|z_i - \widehat{z}_i\right\|\right), \tag{11}$$

where z_i and $\widehat{z}_i$ are the real position and estimated position for ith simulation. Besides, the cumulative distribution functions (CDF) are also shown below.

5.2. Performance Comparison and Results Analysis. We conduct 200 simulations independently and nodes are deployed in the network randomly for each time. Besides, the position of the jammer and the transmitting powers of the jammer and nodes remain unchanged. In this section, the CDF of localization errors and average localization errors are shown.

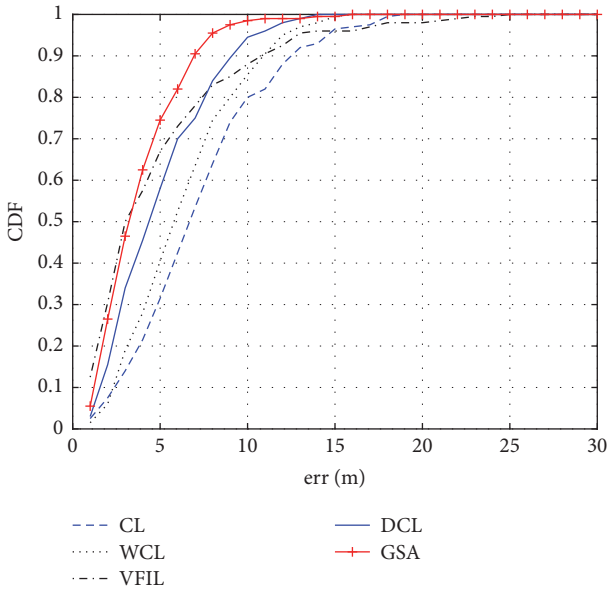


Figure 4: CDF of localization errors.

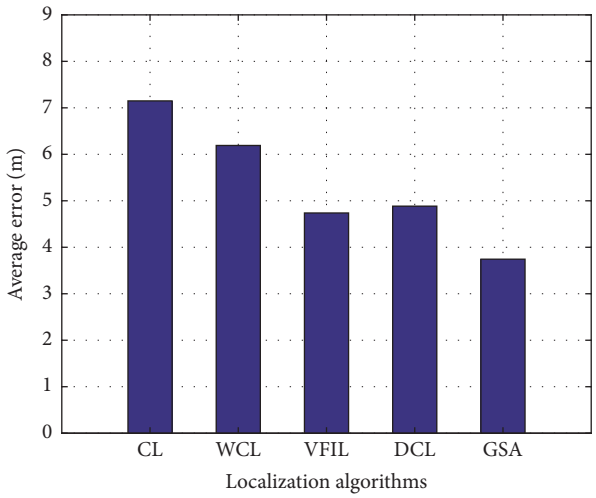


Figure 5: Average localization errors.

Besides, the localization errors influenced by the number of particles are also discussed.

5.2.1. Comparison Results. We compare the performance of GSA with the exiting localization algorithms, CL, WCL, VFIL, and DCL. Figure 4 presents the CDF of different localization algorithms for 200 simulations and Figure 5 presents the average error of localization. From the figures, we can conclude that the localization error of our proposed algorithm is lower than those of CL, WCL, VFIL, and DCL. Besides, we can conclude that VFIL may be more sensitive to the distribution of nodes for some larger values of localization error. The average localization error of GSA can reach 3.7 m, which is smaller than that of the other algorithms.

5.2.2. Impact of Node Density. We compare the localization errors for different node density and the numbers of nodes

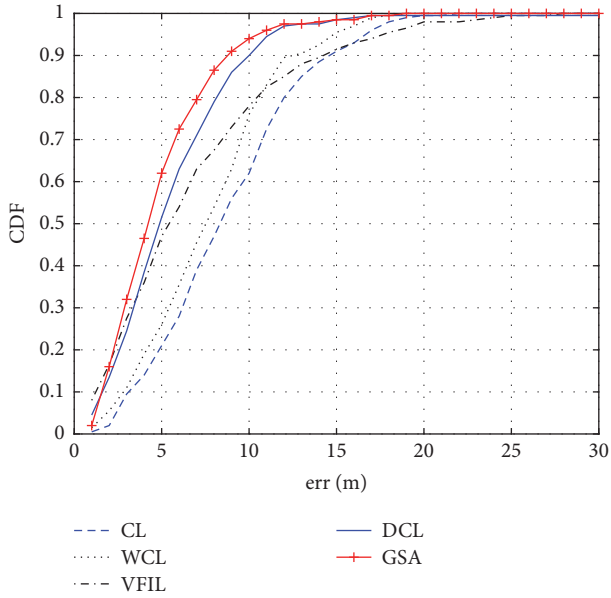


Figure 6: CDF of localization errors when the number of nodes is 300.

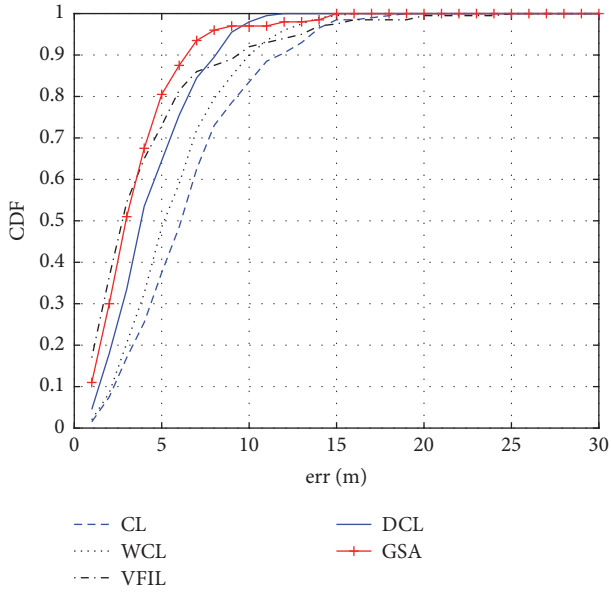


Figure 7: CDF of localization errors when the number of nodes is 500.

are set as 300, 400, and 500, respectively. When the area is assumed to be constant, the nodes' number can be utilized to reflect the node density of the network. After conducting 200 simulations independently, the CDFs of localization errors for different nodes' number are illustrated in Figures 6 and 7. The average localization errors of these algorithms decrease with the increasing number of nodes, which is shown in Figure 8. Besides, GSA achieves better performance than the other four algorithms for different nodes' number.

5.2.3. Impact of Jamming Power. In order to compare the localization performance for different jamming powers, the

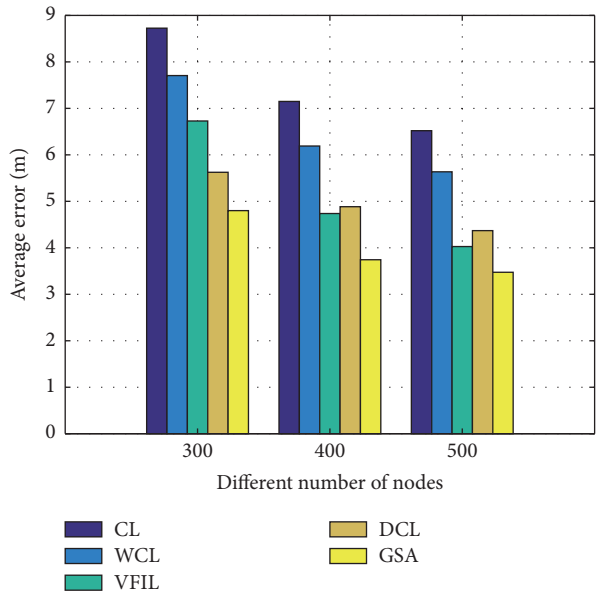


Figure 8: Impact of node density.

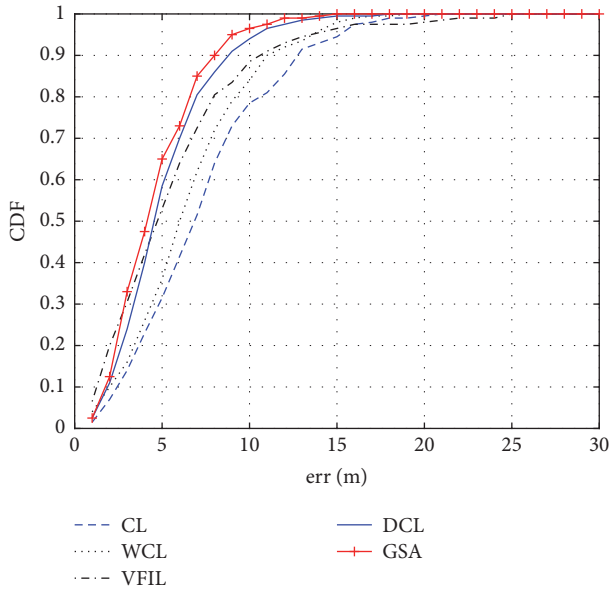


Figure 9: CDF of localization errors when the jamming power is 6 mW.

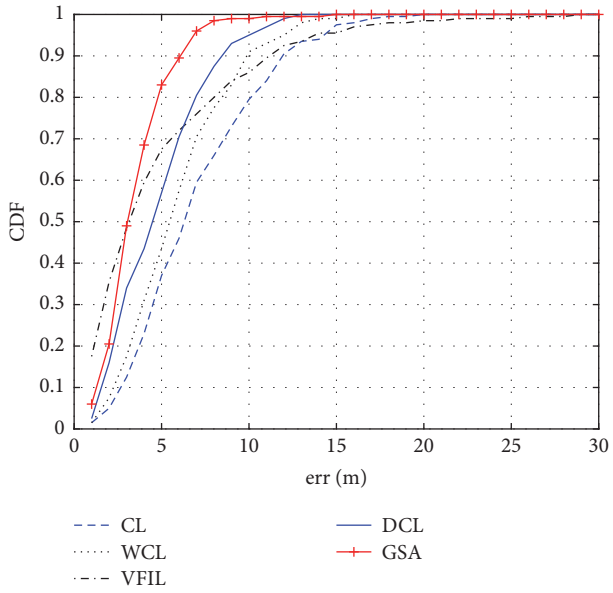


Figure 10: CDF of localization errors when the jamming power is 15 mW.

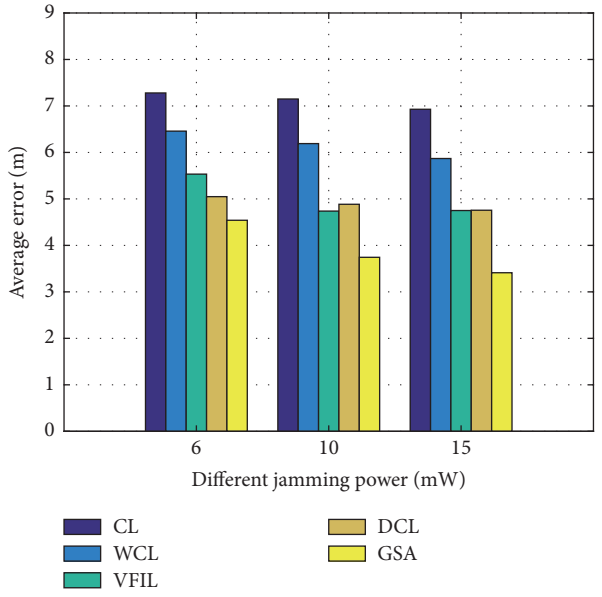


Figure 11: Impact of jamming power.

jamming powers are set as 6 mW, 10 mW, and 15 mW. The CDFs of localization errors for different jamming powers are illustrated in Figures 9 and 10. It can be concluded that as the jamming power increases, the localization errors of these algorithms decrease, which is shown in Figure 11. Besides, we can also conclude that the GSA obtains the best performance for the localization obviously.

5.2.4. Impact of Jammer's Position. In order to analyze the impact of jammer's position on the localization errors, the jammer is located in the positions of (60, 60), (70, 70), and (200, 200), respectively. When the jammer is located in (60, 60) or (70, 70), the assumption that the jammed area is a circle is biased and the real jammed area formulated by the jammed nodes is irregular. From Figure 12, it can be seen that when the jammer locates close to the edge of the network, the localization errors increase for the five algorithms. GSA achieves the best localization performance for different scenarios.

Moreover, a comparison table for outcomes of different applied algorithms is illustrated in Table 2. The average localization errors (m) for CL, WCL, VFIL, DCL, and GSA under different conditions (including number of nodes and jamming power) are illustrated in the table.

Table 2: Average error under different settings.

Number of nodes	Jamming power	*CL*	*WCL*	*VFIL*	*DCL*	*GSA*
300	10 mW	8.73	7.71	6.73	5.63	4.80
400	6 mW	7.28	6.46	5.53	5.05	4.54
400	10 mW	7.15	6.19	4.74	4.88	3.74
400	15 mW	6.93	5.87	4.75	4.76	3.41
500	10 mW	6.52	5.64	4.03	4.37	3.47

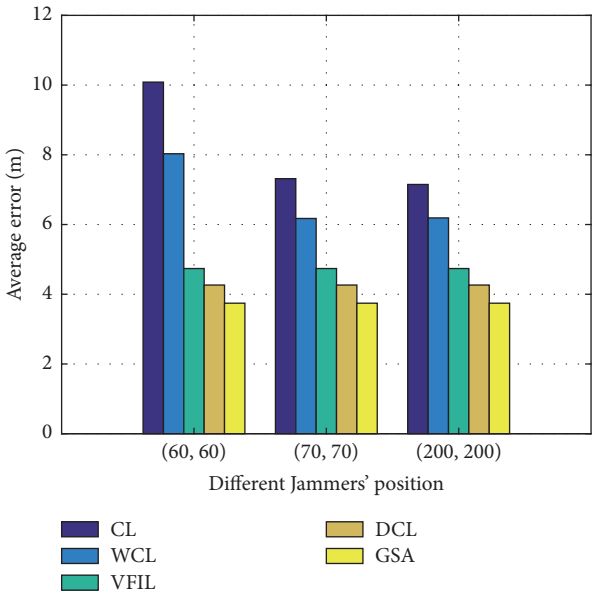


Figure 12: Impact of jammer's position.

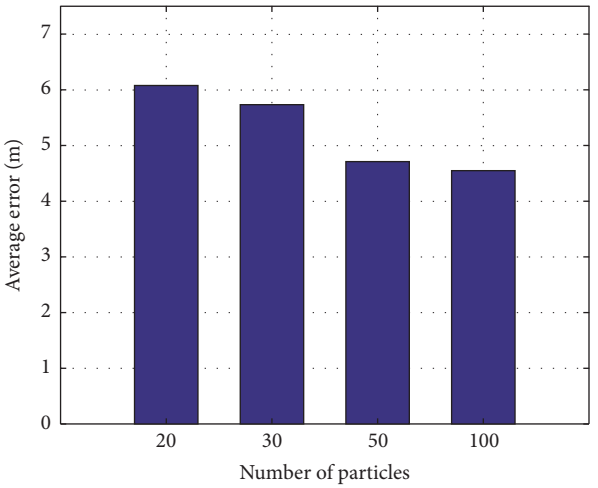


Figure 13: Impact of the number of particles.

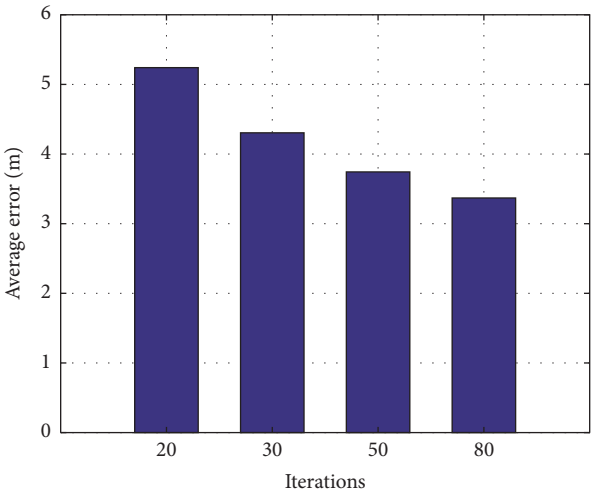


Figure 14: Impact of GSA's iterations.

5.3. Impact of Parameters

5.3.1. Impact of Particles' Number. The localization performance of our proposed GSA algorithm is discussed under different number of particles. The number of particles is assumed to be 20, 30, 50, and 100, respectively, and the localization results are shown in Figure 13. As the number of particles increases, the localization error decreases a little. However, the complexity of GSA is closely related to the number of particles and the tradeoff between the localization accuracy and complexity should be considered when determining the number of particles.

5.3.2. Impact of GSA's Iterations. The localization performance of the GSA-based algorithm is analyzed under different GSA's iterations. The iterations are assumed to be 20, 30, 50, and 80, respectively. The average localization errors for different iterations are illustrated in Figure 14. From the figure, we can conclude that as the GSA's iterations increase, the localization errors decrease. This is due to the fact that GSA searching results will converge to a stable value as the iterations increases. Besides, the complexity of GSA is determined by particles' number and iterations. Although the localization results are more accurate, the execution time of GSA will increase.

6. Conclusion and Future Work

In order to reduce the sensitivity of the existing algorithms to the MHWN nodes deployment and parameters of the jammer for the jammer localization, we have presented a novel localization strategy based on Gravitational Search Algorithm (GSA), which is an evolutionary algorithm based on Newton's law of universal gravitation and mass interactions. The initial particles are assumed to be deployed in the jammed area randomly with acceleration and velocity set to

be zero. The convex hull is adopted to obtain the boundary nodes and these boundary nodes are used to calculate the fitness function for the particles. After the iterations, position of the particle with the maximum mass is considered as the estimated position of the jammer. A series of simulations are conducted and the localization performance of our proposed algorithm is validated. Compared with CL, WCL, VFIL, and DCL, the simulation results show that GSA-based localization algorithm can locate the jammer more accurately.

In the future, we will take the multijammers localization into consideration and put forward efficient localization algorithms.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported in part by the National Natural Science Foundation of China (Grants nos. 61402521 and 61471392) and Natural Science Foundation of Jiangsu Province, China (Grants nos. BK20140068 and BK20150201).

References

[1] S. D. Babar, N. R. Prasad, and R. Prasad, "Jamming attack: Behavioral modelling and analysis," in *Proceedings of the 2013 3rd International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace & Electronic Systems (VITAE)*, pp. 1–5, IEEE, 2013.

[2] K. Pelechrinis, M. Iliofotou, and S. V. Krishnamurthy, "Denial of service attacks in wireless networks: the case of jammers," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 2, pp. 245–257, 2011.

[3] L. Fan, X. Lei, N. Yang, T. Q. Duong, and G. K. Karagiannidis, "Secure Multiple Amplify-and-Forward Relaying with Cochannel Interference," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 8, pp. 1494–1505, 2016.

[4] Y. Liu, F. Jiang, H. Liu, J. Wu, C. Hu, and M. Zhang, "Interference robust channel hopping strategies for wireless sensor networks," *China Communications*, vol. 13, no. 3, Article ID 7445505, pp. 96–104, 2016.

[5] W. Xu, T. Wood, W. Trappe, and Y. Zhang, "Channel surfing and spatial retreats: Defenses against wireless denial of service," in *Proceedings of the 2004 ACM Workshop on Wireless Security, WiSe*, pp. 80–89, October 2004.

[6] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure attribute-based data sharing for resource-limited users in cloud computing," *Computers & Security*, vol. 72, Article ID S0167404817301621, pp. 1–12, 2018.

[7] Z. Huang, S. Liu, X. Mao, K. Chen, and J. Li, "Insight of the protection for data security under selective opening attacks," *Information Sciences*, vol. 412-413, pp. 223–241, 2017.

[8] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'05)*, pp. 46–57, ACM, Urbana-Champaign, Ill, USA, May 2005.

[9] X. Wei, Q. Wang, T. Wang, and J. Fan, "Jammer Localization in Multi-Hop Wireless Network: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 765–799, 2017.

[10] Z.-M. Wang and Y. Zheng, "The study of the weighted centroid localization algorithm based on RSSI," in *Proceedings of the 2014 International Conference on Wireless Communication and Sensor Network, WCSN 2014*, pp. 276–279, Wuhan, China, December 2014.

[11] H. Liu, X. Wenyuan, Y. Chen, and Z. Liu, "Localizing jammers in wireless networks," in *Proceedings of the 2009 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, vol. 25, pp. 1–6, Galveston, TX, USA, March 2009.

[12] Y.-Q. Sun, X.-D. Wang, and X.-M. Zhou, "Geometry-covering based localization for jamming attack in wireless sensor networks," *Journal of China Institute of Communications*, vol. 31, no. 11, pp. 10–16, 2010.

[13] J. Zhang, L. Xu, Q. Shen, and X. Ji, "Localization for Jamming Attack in Wireless Sensor Networks," in *Intelligent Data Analysis and Applications*, vol. 370 of *Advances in Intelligent Systems and Computing*, pp. 361–369, Springer International Publishing, 2015.

[14] X. Wei, T. Wang, C. Tang, and J. Fan, "Collaborative mobile jammer tracking in Multi-Hop Wireless Network," *Future Generation Computer Systems*, vol. 78, pp. 1027–1039, 2018.

[15] T. Cheng, P. Li, S. Zhu, and D. Torrieri, "M-cluster and X-ray: Two methods for multi-jammer localization in wireless sensor networks," *Integrated Computer-Aided Engineering*, vol. 21, no. 1, pp. 19–34, 2014.

[16] Q. Wang, J. Fan, X. Wei, and T. Wang, "Multi-jammers localization for multi-hop wireless network," *Journal of Communication*, vol. 37, no. 13, pp. 176–186, 2016.

[17] K. Pelechrinis, I. Koutsopoulos, I. Broustis, and S. V. Krishnamurthy, "Lightweight jammer localization in wireless networks: System design and implementation," in *Proceedings of the 2009 IEEE Global Telecommunications Conference, GLOBECOM 2009*, pp. 1–6, December 2009.

[18] Z. Liu, H. Liu, W. Xu, and Y. Chen, "Exploiting Jamming-Caused Neighbor Changes for Jammer Localization," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 3, pp. 547–555, 2012.

[19] Q. Wang, X. Wei, J. Fan et al., "A step further of PDR-based jammer localization through dynamic power adaptation," in *Proceedings of the 11th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2015)*, pp. 1–6, Shanghai, China, 2015.

[20] P. Bahl and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM '00)*, vol. 2, pp. 775–784, Tel Aviv, Israel, March 2000.

[21] L. Fan, X. Lei, N. Yang, T. Q. Duong, and G. K. Karagiannidis, "Secrecy Cooperative Networks with Outdated Relay Selection over Correlated Fading Channels," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7599–7603, 2017.

[22] R. Zhao, Y. Yuan, L. Fan, and Y.-C. He, "Secrecy Performance Analysis of Cognitive Decode-and-Forward Relay Networks in Nakagami-m Fading Channels," *IEEE Transactions on Communications*, vol. 65, no. 2, pp. 549–563, 2017.

[23] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "Gsa: a gravitational search algorithm," *Information sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[24] Z. Liu, H. Liu, W. Xu, and Y. Chen, "Error minimizing jammer localization through smart estimation of ambient noise," in *Proceedings of the 9th IEEE International Conference on Mobile Ad-Hoc and Sensor Systems, MASS 2012*, pp. 308–316, October 2012.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 6063456, 10 pages
https://doi.org/10.1155/2018/6063456

Research Article

A Smart Trust Management Method to Detect On-Off Attacks in the Internet of Things

Jean Caminha,[1,2] **Angelo Perkusich**,[2] **and Mirko Perkusich**[2]

[1]*Computing Institute, Federal University of Mato Grosso, Cuiabá, MT, Brazil*
[2]*Embedded Systems and Pervasive Computing, Federal University of Campina Grande, Campina Grande, PB, Brazil*

Correspondence should be addressed to Jean Caminha; jean@ic.ufmt.br

Received 24 November 2017; Revised 7 February 2018; Accepted 25 February 2018; Published 15 April 2018

Academic Editor: Ilsun You

Internet of Things (IoT) resources cooperate with themselves for requesting and providing services. In heterogeneous and complex environments, those resources must trust each other. On-Off attacks threaten the IoT trust security through nodes performing good and bad behaviors randomly, to avoid being rated as a menace. Some countermeasures demand prior levels of trust knowledge and time to classify a node behavior. In some cases, a malfunctioning node can be mismatched as an attacker. In this paper, we introduce a smart trust management method, based on machine learning and an elastic slide window technique that automatically assesses the IoT resource trust, evaluating service provider attributes. In simulated and real-world data, this method was able to identify On-Off attackers and fault nodes with a precision up to 96% and low time consumption.

1. Introduction

The Internet of Things (IoT) connects people and things to information systems via smart devices based on Internet-oriented devices and knowledge paradigms. In the near future, over 50 billion devices will be connected to the Internet, supporting several applications like smart cities, smart houses, supply chain, and precision agriculture [1]. Given this heterogeneous and complex environment, security is a key concern in the IoT, and devices must be smart to protect systems from threats [2].

The swarm concept applied to the IoT leverages the independent cooperation of devices to execute tasks and demands special infrastructure to support heterogeneous device interactions, such as those in smart cities environments (Figure 1). The components of a swarm system must connect seamlessly and trust each other.

Security and privacy problems challenge the IoT vision. Large amount of data will be produced from billions of interactions between devices and people in new and existing paradigms like cloud computing, machine to machine, Internet of Vehicles, Internet of Energy, and Internet of Sensors [3].

IoT resources interact with themselves by requesting and providing services, sometimes opportunistically as they come into contact with each other. In this context, misbehaving devices may perform discriminatory attacks based on trust abuse. To maximize system security, it is important to evaluate the trustworthiness of service providers in IoT environments [4].

An IoT device can act as a service provider or service requester. A service requester wants to select the best service provider and trust it. A malicious provider resource can offer bad services and information, thereby compromising systems. Trust attacks and their countermeasures are an open issue being addressed by researchers [5].

On-Off attacks are considered a selective attack type. Multiservice IoT architectures may suffer attacks from malicious nodes that perform actions based on type of service they provide to other nodes in the network. A malicious device can provide good and bad services randomly to avoid being rated as a low trust node. An On-Off (OA) attacker can also behave differently with different neighbors to achieve inconsistent trust opinions of the same node. This kind of attack is hard to detect using traditional trust management schemes [6].

Figure 1: The swarm concept applied to IoT enabled smart city.

In addition, not all misbehaving devices are attackers. Some may be devices in malfunction status. Separating attackers nodes from broken nodes is useful for systems administrators to recover the IoT systems.

To mitigate threats like OA, artificial intelligence and machine learning boost the performance of security solutions in IoT architectures. Researches in security using artificial intelligence range from simple modeling attack patterns to sophisticated anomaly detection schemes, which can detect unknown attacks. Machine learning applications are developing security solutions like antivirus, network intrusion systems, and fraud-detection systems [7].

The solution described in this study aims to provide a real-time method to automatically assess OA resources, evaluating service attributes such as data provision or quality-of-service data. For this purpose, we developed an elastic slide window and machine learning based trust management method to aid systems and users to protect themselves against OA.

The main contributions of this study are the following:

(i) Introduction of a method to improve security in IoT by identification of On-Off trust attacks, using less data

(ii) An efficient method to differentiate attackers nodes from broken nodes

(iii) A flexible implementation, which combines machine learning and elastic sliding window to correctly understand variations in trusted devices outputs

(iv) Design and execution of a proof of concept using simulated and real data from a smart city project, demonstrating the efficiency of the method.

This article is structured as follows. Section 2 presents related works on OA. Section 3 presents our smart trust management method and methodologies to validate the solution. Section 4 presents the results achieved in simulated and real-world scenarios. Finally, Section 5 presents our final conclusions and future works.

2. Related Works

Trust evaluation is an essential part of a trust management scheme. There are many different methods to compute the degree of trust in distributed networks. They can be divided into direct and indirect trust. Direct trust refers to methods that infer a trust score owing to direct data observations. Indirect trust uses reputation and recommendations by other peers. Trust scores can persist in a known central node or an authorized third party. In a decentralized model of trust evaluation, a node computes a trust value for every node interaction.

The distributed management approach in [8] computes the trust value locally by nodes. The trust value is based on direct observations, through the service availability of related node. This scheme is time and resource consuming. It required 120 min to fill the local table with suspicious and trusted nodes. Another drawback of this approach is that it did not consider the initial trust level of a peer. A recent study [6] found this introduced reward and punishment scheme as only method to protect against OA menace in IoT environments.

The adaptive security model in [9] is based on a trust evaluation method composed of three complementary components: experiences, observations, and recommendations. It focuses on reducing resource consumption in mobile ad hoc network. The clustering architecture in [10] addresses trust management in the IoT based on the similarity of interest in each cluster. Its prediction mechanism uses the Kalman filter to estimate the trust value in advance.

The trust-based offloading method for mobile M2M communications in [11] uses reinforcement learning and builds a feedback system. The trust level of an initiator node toward other nodes is updated after each communication to enable the node to more precisely evaluate new interactions. It focuses on trust which can improve the energy consumption and computation speed of devices and improve the availability of the system. However, this scheme does not consider the different services provided by a peer, and nor does it consider the trustworthiness of the collected trust data of each node.

RealAlert is a policy-based secure and trustworthy sensing scheme proposed in [12]. In this scheme, the data trustworthiness and IoT node attributes are assessed using anomalous IoT data and contextual information that represents the environment from which anomalous IoT data were obtained. Policy rules are defined to specify how to evaluate the trustworthiness in different situations. New devices or new normal observations may be considered attacker by an outdated policy.

The quantitative model of trust value based on multi-dimensional decision attributes in [13] uses the monitored direct trust value measured from network communication. Packet forwarding capacity, repetition rate, consistency of the packet content, delay, and integrity are evaluated. It adopts the D-S theory to computes the trust. Its drawback is related to a large amount of data collected from various devices in the IoT environment. The amount of data increases exponentially, and continuously streaming data is difficult to manage with traditional network communication data analysis methods.

The study [14] uses a trust relationship scheme based on clustered wireless sensor networks (M2M). A cloud model implements the conversion between qualitative and quantitative data of sensor nodes (trust metrics). To calculate the trustworthiness of sensor nodes, the proposed method considers communication, message, and energy factors, a

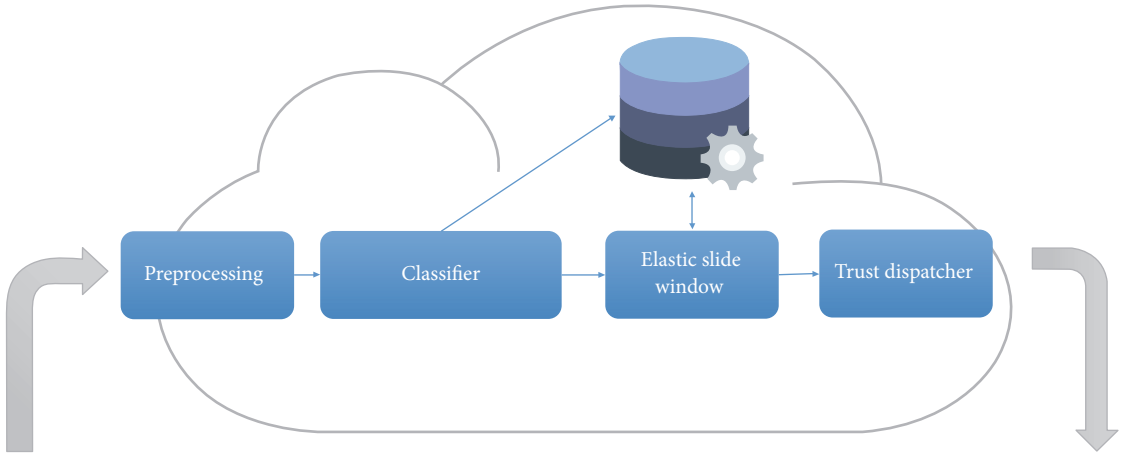


Figure 2: The smart trust management method data flow on cloud.

Table 1: Related works comparison.

Study	FDN	NPTS	DAM
[8]	x		
[9]	x		
[10]	x	x	
[11]	x	x	
[12]			
[13]	x		
[14]			
[15]	x		
Our method	X	X	X

mathematical assignment of a dynamic weight for each trust factor to detect attacks. Cloud solutions are threatened by vendor related problems (proprietary protocols or end-to-end designs), low end devices computer power, and service discovery incompatible methods.

The trust management and redemption scheme in [15] discriminates between temporary errors and malicious behaviors to detect and defend against OA. It uses predictability trust, computed as the ratio of good behavior to the total behavior in the system, and a static sliding window that records previous behavior history. This scheme needs time to compute the entire system behavior. Maintaining a static behavior record, it cannot accommodate new trusted acts.

All studies evaluated were also compared (Table 1) in terms of information (metadata entries) needed to compute the trust score (FDN), no need to know previous trust score from a neighbor node (NPTS), or the differentiation between attacker nodes and malfunctioning (broken) ones (DAM).

The method presented in this work fills the FDN, PTS, and DAM gaps present in other works and aggregates more efficiency in terms of detection precision, recall, and implementation flexibility.

3. Methodology

In this section we demonstrate our proposed method to detect OA in IoT. The two validation scenarios are described. To find the best machine learning classifier, we also conduct a comparison between supervision methods.

3.1. A Smart Trust Management Method to Detect On-Off Attacks in the Internet of Things. The goal of the proposed method is to collaborate with IoT systems to identify OA attacks and broken nodes. Interactions among IoT devices are evaluated using available metadata attributes. This smart trust management method is designed to be accessed through a representational state transfer (REST) application programming interface (API). Figure 2 illustrates the metadata flow of the cloud.

An IoT object metadata can be sent to the method to be evaluated. In the preprocessing phase, data are submitted to related feature type extraction process. Text data are processed by the HashingVectorizer [16]. This process converts text (n-grams only from text inside word boundaries) to a matrix of token occurrences and finds a token string name for the feature integer index mapping. This approach was selected because it does not store a vocabulary dictionary in memory and can also be used in streaming (partial fit).

Each kind of preprocessed data is submitted to a specific machine learning classifier to identify its class. Whenever the data, such as the annual temperature range for a city, is on the range of accepted values it is assumed to be trusted (Figure 3). Otherwise, out-of-range values are assigned as outliers.

The classifier confirms whether it identified a class and returns a decision function value. The decision function is used by our method to determine the elastic slide window size. It is calculated by observing the evaluated data sample distance hyperplane of the model decision function. The degree of separation achieved by the hyperplane has the largest distance to the nearest training data points of any class (the functional margin). A high (or positive) decision function value corresponds to the prediction assurance.

Figure 3: Expected range of trusted values.

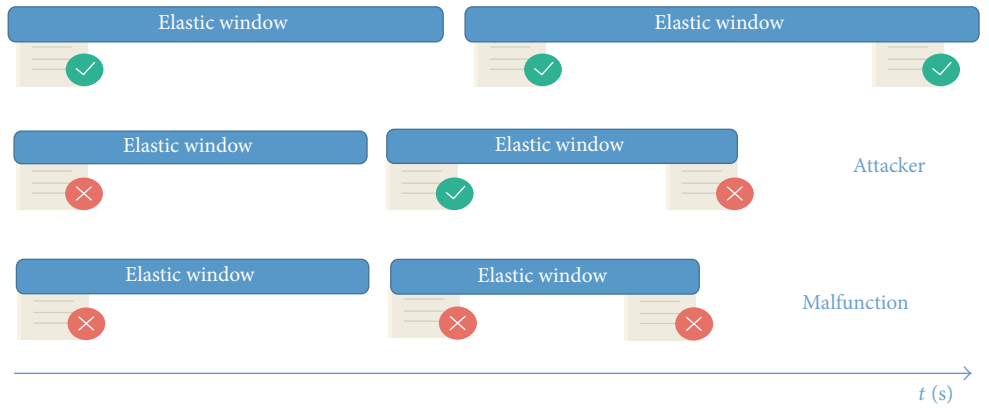


Figure 4: The elastic slide window.

Table 2: Output decisions for two analyses in the same elastic slide window.

Elastic slide window						Output	
	Read 1			Read 2			
Class	High	Low	Class	High	Low	Thing	ESW size
x	x		x	x		Trusted	Decrease
x	x		x		x	Trusted	Increase
x		x	x	x		Trusted	Decrease
x		x	x		x	Trusted	Increase
x	x			x		Attacker	Decrease
x	x				x	Attacker	Increase
x		x		x		Attacker	Decrease
x		x			x	Attacker	Increase
	x		x	x		Attacker	Decrease
	x		x		x	Attacker	Increase
		x	x	x		Attacker	Decrease
		x	x		x	Attacker	Increase
	x			x		Broken	Decrease
	x				x	Broken	Increase
		x		x		Broken	Decrease
		x			x	Broken	Increase

The elastic slide window (ESW) presented in Figure 4 is an important phase of the data flow. It enhances the trust using time frame analysis. An OA sends good and bad read values in a discretionary manner. Healthy systems expect only good values over time. When the classifier sends an identified class and low decision function value, our method assumes there are doubts about trust, and the evaluated resource needs to be tested again in a larger time frame. With each interaction, the elastic slide window is aggregated by decision function values. Low decision function values lead to higher elastic slide windows and permit the method to analyze the variance in significant node behaviors (good or bad).

The trust dispatcher is responsible for saving the elastic slide window size value in a database or memory for future reference. It also determines the trust resource type: Good, On-Off attacker, or Broken. The trust dispatcher implements the decisions mapped in Table 2.

Some implementation details must be considered in our smart trust management method (Algorithm 1). Two variables need to be initialized to the elastic slide window:

```
    input: A metadata m (ID, read)
    output: A predicted type: (Tusted, On-OffAttacker, Broken)
(1) eswAlpha ← alpha;
(2) eswInit ← beta;
(3) NewPrediction ← Classifier.Predict(m);
(4) NewDecisionFunction ← Classifier.DecisionFunction(m);
(5) if m in Database then
(6)     m.SlideWindow ←
          (eswInit + time()) − NewDecisionFunction;
(7)     m.prediction ← NewPrediction;
(8)     if m.SlideWindow ≥ Time() then
(9)         if NewPrediction == −1 and m.prediction == −1 and
              NewDecisionFunction ≤ eswAlpha then
(10)            m.prediction ← 0;
(11)        end
(12)        if NewPrediction != m.prediction and
              NewDecisionFunction ≥ eswAlpha then
(13)            m.prediction ← −1;
(14)        end
(15)        if NewPrediction != m.prediction and
              NewDecisionFunction ≤ eswAlpha then
(16)            m.prediction ← m.prediction;
(17)        end
(18)    end
(19) end
(20) m.SlideWindow ← m.SlideWindow − NewDecisionFunction;
```

Algorithm 1: The smart trust management method algorithm.

eswAlpha(line 1) and *swInit*(line 2). *eswAlpha*(1) allows system administrator to define the value to be considered as trusted in a decision function score (lines 11, 14, and 37). It is also utilized to calculate the growth degree of an ESW (line 30). The variable *swInit* records the initial ESW time (in seconds) for a new resource.

The verification in line 9 checks if a ESW for a resource is greater than the actual computer time and must be considered in this analysis. New ESW sizes are calculated using previous ESW values minus the new decision function value. Negative decision function values aggregate the ESW size (line 30). The output decisions for two analyses in the same elastic slide window are implemented in lines 7, 11, 14, and 17.

The smart trust management method returns results by a REST/API query. Figure 5 shows an example of an output analysis from our method. The smart trust management server is consulted via the Constrained Application Protocol (CoAP) about an object and returns the results in JSON formatted data. For example, the object *ID*: 14 with a metadata payload 46 (degree Celsius) is marked as an *On-Offattacker*. A trust score (the decision function value) is also presented.

Other possible results are *Good* for predictable devices or *Broken* for two or more fault values in the same elastic slide window.

The classifier model used by the smart trust management method was created by the OneClassSVM (RBF kernel, upper and lower bound on fraction of training errors = 0.1, and kernel coefficient = 0.1) method, trained with 2000 samples, implemented with Python 3.6 scikit-learn and LIBSVM

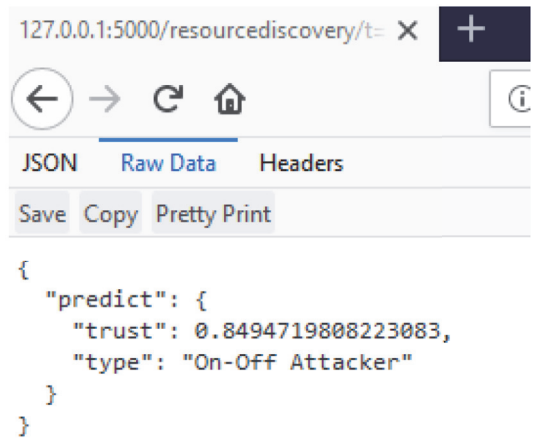


Figure 5: Example of an output result by the smart trust management server.

library [16], version 0.18.0, on a desktop machine with a Core i7 3.60 GHz processor and running Windows 10.

To validate our solution, we prepared two experimental setups: computer simulation and a real-world scenario. The setups have annotated data for OA nodes and broken nodes.

3.2. Simulation Validation. Our simulation setup consists of 51 nodes' set (Figure 6). A node is the destination object representing the data consumer and uses our method to identify trusted and misbehaved nodes. Forty nodes act as

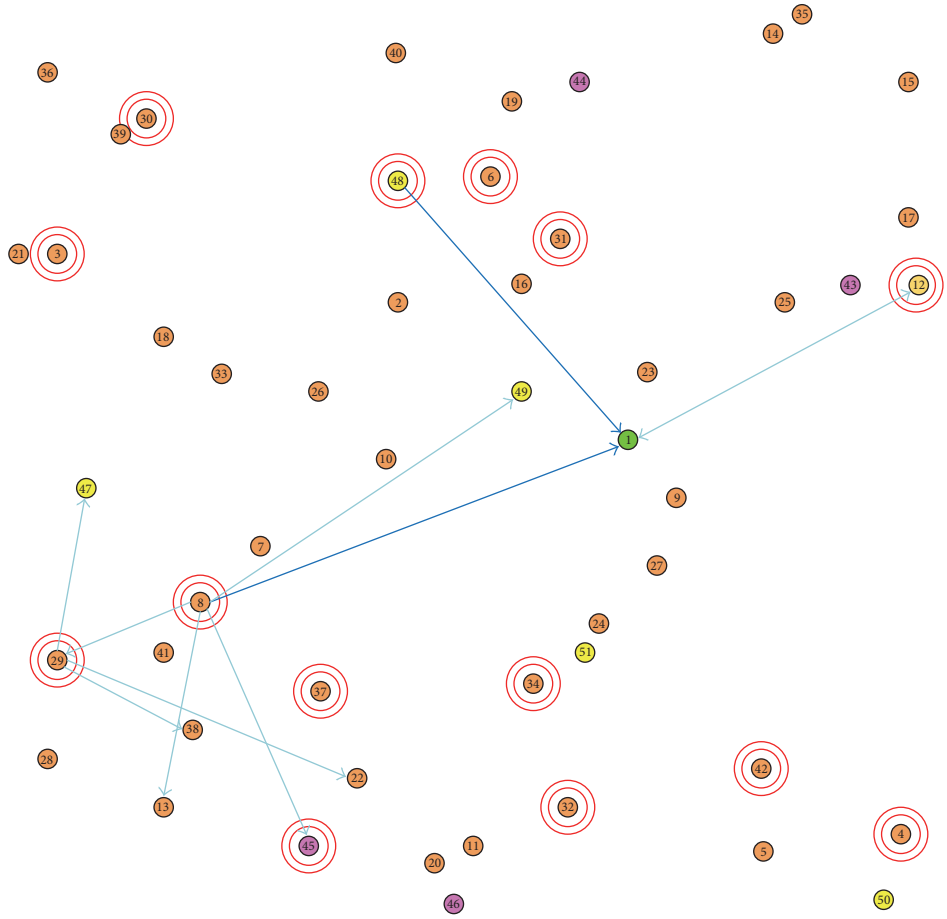


Figure 6: Simulation experiment scenario.

good nodes (orange) and only provide trusted values. Five nodes (pink) are OA and they provide randomly trusted and untrusted data. We also simulated five malfunction nodes (yellow), which always send untrustable data.

Table 3 gives the simulation configuration parameters. We used the Cooja simulator for Contiki 3.0 OS. The simulation ran for two hours on a desktop machine with a Core i7 3.60 GHz processor running Windows 10. The simulation generated 4844 samples of data. Cooja is popular within the WSN and IoT research community and results can be benchmarked with other studies.

3.3. Real Data Validation. To evaluate our method in a real-world scenario, we used 4111 samples of temperature data from February to March 2015 from the city of Aarhus, located in Denmark [17]. This city has a regular temperature range during this time of the year ranging from −3 to 16 degrees Celsius (Figure 7). A total of 500 misbehavior attack samples were simulated using random out-of-range temperature observations (from −23 to 36 degrees Celsius) and injected in the test dataset.

3.4. Classifiers Comparison. To verify the best classifier method to solve this problem we conducted tests with

Table 3: Summary of simulation parameters.

Parameter	Value
Simulator	Cooja under Contiki 3.0 OS
Radio environment	Unit disk graph medium (UDGM): dist. loss
Deployment area	400 m × 400 m
Type & number of nodes	Sky mote, 50 senders & 1 sink
Range of nodes	Trans. range: 50 m, interference range: 50 m
Physical layer	IEEE 802.15.4
MAC layer	IPv6
Network layer	RPL
Transport layer	UDP
Simulation duration	2 h
Sending rate	1 packet in every 20–60 sec

two kinds of classifiers: one-class and multiclass support supervised classifiers. One-class classifiers studied were OneClassSVM, robust covariance (EllipticEnvelope), and isolation forest. Nearest Neighbors (KNeighborsClassifier),

Table 4: Classifiers configuration.

Classifier	Configuration
OneClassSVM	OneClassSVM (cache_size = 200, coef0 = 0.0, degree = 3, gamma = 0.01, kernel = "rbf",, max_iter = −1, nu = 0.01, random_state = None, shrinking = True, tol = 0.001,, verbose = False)
Elliptic Envelope	EllipticEnvelope (assume_centered = False, contamination = 0.1, random_state = None,, store_precision = True, support_fraction = None)
Isolation Forest	IsolationForest (bootstrap = False, contamination = 0.1, max_features = 1.0,, max_samples = "auto", n_estimators = 100, n_jobs = 1, random_state = None,, verbose = 0)
Nearest Neighbors	KNeighborsClassifier (algorithm = "auto", leaf_size = 30, metric = "minkowski",, metric_params = None, n_jobs = 1, n_neighbors = 5, p = 2,, weights = "uniform")
Linear SVM	(C = 0.025, cache_size = 200, class_weight = None, coef0 = 0.0,, decision_function_shape = None, degree = 3, gamma = "auto", kernel = "linear",, max_iter = −1, probability = False, random_state = None, shrinking = True,, tol = 0.001, verbose = False)
Neural Net	MLPClassifier (activation = "relu", alpha = 1, batch_size = "auto", beta_1 = 0.9,, beta_2 = 0.999, early_stopping = False, epsilon = $1e-08$,, hidden_layer_sizes = (100,), learning_rate = "constant",, learning_rate_init = 0.001, max_iter = 200, momentum = 0.9, nesterovs_momentum = True, power_t = 0.5, random_state = None,, shuffle = True, solver = "adam", tol = 0.0001, validation_fraction = 0.1,, verbose = False, warm_start = False)
Naive Bayes	GaussianNB (priors = None)

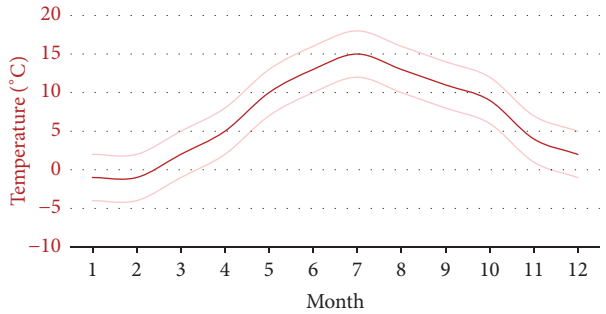


Figure 7: Temperature range observed from the city of Aarhus (extracted from https://en.climate-data.org/location/302/).

linear SVM, Naive Bayes (GaussianNB), and Neural Net (multilayer perceptron) were tested as multiclass classifiers. Table 4 shows the configuration used in each classifier tested.

All classifiers were trained with the same train simulated dataset, with 2000 reads, and tested with a 4844-sample test dataset. A zero-value array was used as a secondary class for multiclass classifiers.

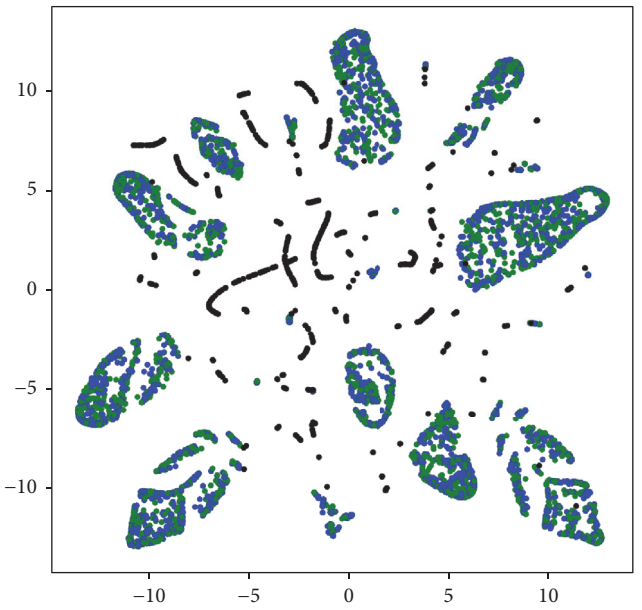


Figure 8: Decision function hyperplane.

4. Results

Our proposed security method was able to detect OA in the IoT with 97% of precision in a real-world dataset and 96% of precision in simulated environment. Compared to other studies, our method was 95% faster and 5% more precise in OAs identification. In a novel way, the elastic window feature helped differentiate broken or malfunctioning nodes among misbehaving devices.

The hyperplane related to the model created by the OneClassSVM classifier is shown in Figure 8. This visualization was generated using the t-SNE technique. The trained dataset (blue points) was grouped to create the decision function boundaries. Meanwhile new normal observations (green points) are likely close to the data used for training. OA (black points) or broken devices are located far from decision function boundaries.

The OneClassSVM classifier efficiently grouped the test data near to the trained dataset, without overfitting. Abnormal samples were presented far from the decision function.

The decision function returns a distance value of an evaluated sample from the decision hyperplane. If a class is identified, normal (trusted) observations have values near 0, while attacks (misbehaving and out-of-range reads) have high distance values (up to −200). In Figure 9 we showed the trust score (distance from decision function) for a set of observed reads.

The two experimental setups were trained with a set of 2000 reads related to the temperature range observed in the

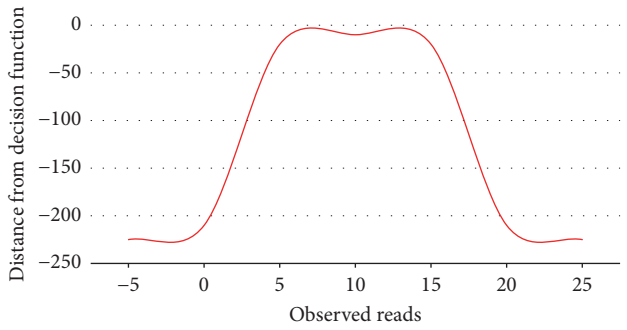


Figure 9: Distance from decision function for observed reads.

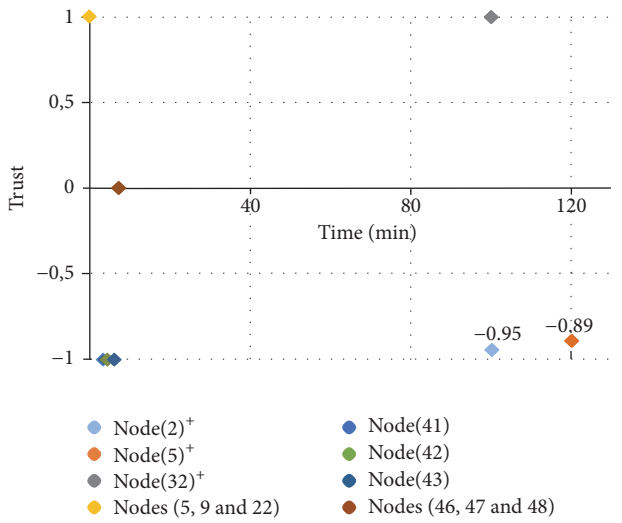


Figure 10: Time spent to identify OA and broken nodes.

city of Aarhus (−3 to 16 degrees Celsius). Test reads in this range obtain decision function distance values near to zero, while other values received low values.

4.1. Simulation Validation. Study [8] utilizing the number, position, and traffic volume of malicious nodes demands approximately 120 min to determine OA. Our method identified OAs in 5 min average time (95% faster) and 96% precision. Figure 10 shows the time spent to identify OA. Nodes 31^* (good), 8^*, and 32^* (attackers) are from compared study [8] and Nodes 5, 9, 22 (goods), 41, 42, and 43 (OA) are from our simulated scenario. Nodes 46, 47, and 48 are broken nodes and also were identified in 7 min. Positive trust scores are related to good nodes and negative trust scores to attackers respectively.

In various situations, not all misbehaving devices are attackers. Some of them may be devices in malfunction status. We did not identify relevant researches in the differentiation IoT OA from errors sent by broken nodes. In this paper, we compare our solution to density-based spatial clustering of applications with noise (DBSCAN) algorithm. DBSCAN identified only two classes. The results in Figure 11 show the true positives (good, attackers, and broken nodes) predicted

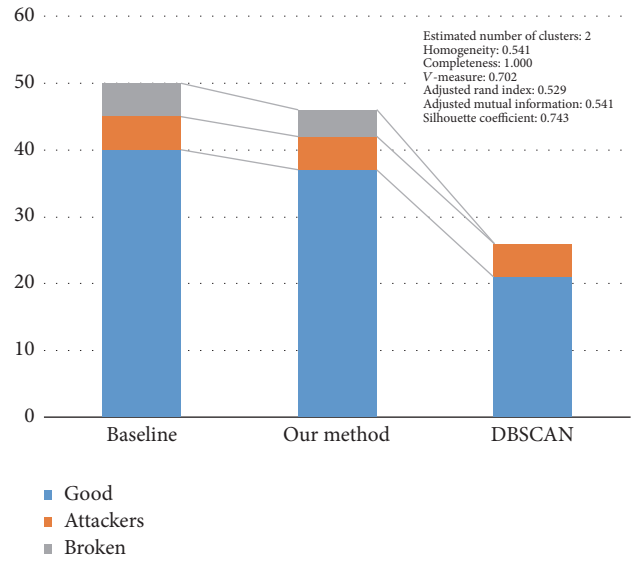


Figure 11: Comparison with DBSCAN clustering algorithm.

Table 5: Comparison with supervised classifiers.

Classifier	Precision	Recall	*F*1-score
Linear SVM	0.88	0.71	0.74
Naive Bayes	0.92	0.82	0.85
Neural Net	0.92	0.81	0.84
Nearest Neighbors	0.91	0.84	0.87
Our method	**0.96**	**0.85**	**0.87**

from the presented method and the DBSCAN, compared to expected nodes (baseline).

The annotated simulated dataset was useful to conduct another validation. We train, test, and compare our method to Nearest Neighbors (KNeighborsClassifier), linear SVM, Naive Bayes (GaussianNB), and Neural Net (multilayer perceptron). The results in Table 5 show our method, with the elastic slide window approach, reaches superior precision, recall, and $f1$ scores.

Our method was able to find three good nodes, two attackers, and two broken nodes more than the supervision methods.

Figure 12 shows how nodes were identified by our method in simulation experiment. The filled points are the actual nodes. The yellow circles are the good nodes, the cyan squares are the attackers, and the red diamonds are the malfunctioning nodes. The mark around the nodes represents the predicted class.

4.2. Real Data Validation. To develop and evaluate the smart middleware, we used 4111 samples of temperature data collected by 115 sensors from February to March 2015 from the city of Aarhus, in Denmark [17]. This city has a regular temperature range during this time of the year ranging from −3 to 16 degrees Celsius. A total of 500 misbehavior samples were simulated using random out-of-range temperature observations (from −10 to 30 degrees Celsius).

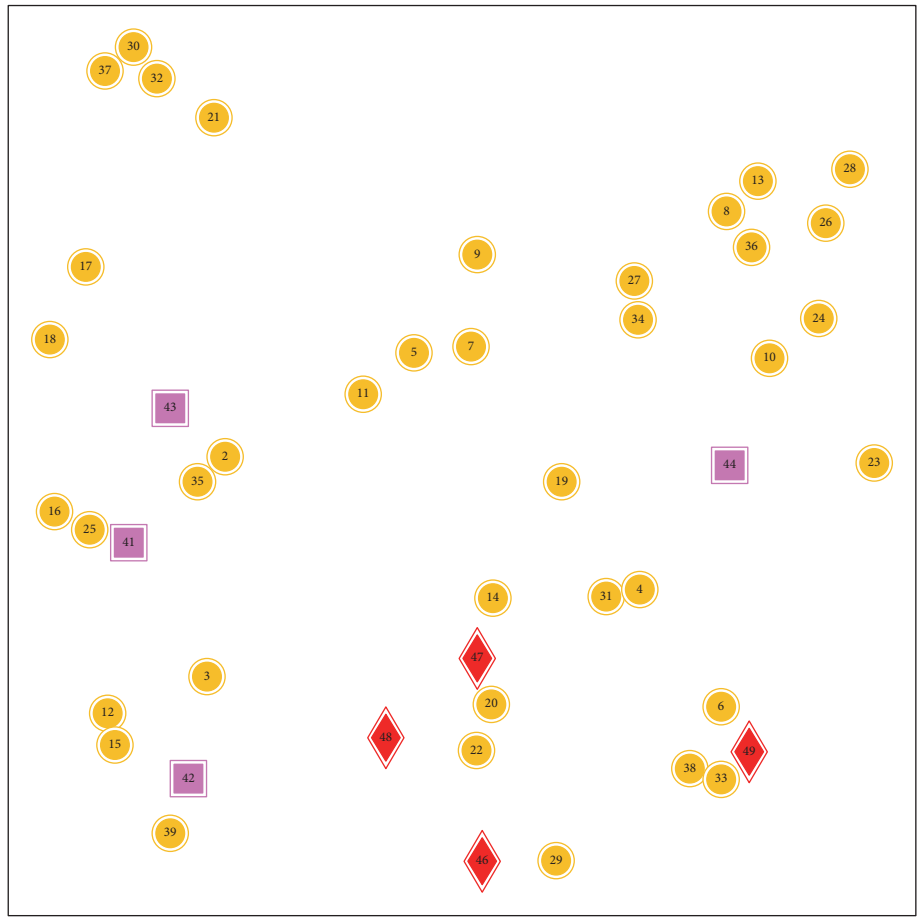


Figure 12: Resources identified by our method.

Similar to the simulation experiment, our method reaches 97% of precision and 95% of recall in classifier goods and misbehaving nodes.

4.3. Classifiers Comparison. The proposed method uses the output results from one-class classifier types. Classifiers like the OneClassSVM classifier are proven to be efficient in outliers detection [18]. We use one-class classifiers because our method only needs to differentiate good (expected) metadata reads from abnormal ones. In real-world implementation, it is hard to train a classifier with each type of attack data. In addition, the system analysis capacity can be enhanced with new one-class classifier for new evaluated metadata.

To identify the best one-class classifier, we compare OneClassSVM, robust covariance, and isolation forest classifiers with the same train and test data (Table 6). The OneClassSVM reaches 99% of precision and 98% of recall in trusted resource prediction.

Table 6: One-class classifiers comparison for outlier detection in our method.

Classifier	Precision	Recall	*F*1-score
OneClassSVM	**0.99**	**0.98**	**0.99**
Robust covariance	**0.99**	0.90	0.94
Isolation forest	**0.99**	0.90	0.94

5. Threats to Validity

This method may have been affected by some validation threats, such as related random simulation outputs, classifiers seeds utilized, and datasets.

Simulations outputs may vary from each run period. To minimize this threat, we ran each simulation three times and use the average results. The scikit-learn library documentation alerts users that it uses random seed values, parameter initialization variable, in the training tasks of the models, which slightly corroborates to different precision results. As in simulations, we annotate average values of three fitting rounds. The real-world dataset used was sanitized removing NaN values and nonrelated data.

The related works session may have been threatened by failing to consider any relevant study. To minimize this risk, we consulted the main indexing databases to verify references and citations to/from selected studies. However, some studies may not have been considered owing to the technical limitation of search engines, the possibility of

electronic databases not representing the complete list of all available studies, and the nonavailability from the Brazilian academic network or access to the printed only version.

6. Conclusion

In this article, we introduced a smart trust management method based on machine learning and an elastic slide window technique that automatically assesses the IoT resource trust by evaluating service provider attributes. Our proposed security method was able to detect OAs in the IoT with 97% of precision in a real-world dataset and 96% of precision in simulated environment. Compared to other studies, our method was 95% faster in OA identification.

The smart trust management method presented here contributes to IoT trust management, protecting systems against OA using less data (starting from two entries of one feature). In a novel way, the elastic slide window feature also helps to differentiate broken or malfunctioning nodes among misbehaving devices.

For future works, we plan to increase the overall precision of our method by using other datasets for training and adjusting the classifier configuration. We also intend to use the elastic slide window to identify other IoT trust related attacks, like opportunistic service attacks, ballot-stuffing attacks, self-promotion attacks, and bad-mouthing attacks.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was partially supported by Fundação de Amparo a Pesquisa de Mato Grosso (FAPEMAT), Brazil.

References

[1] A. Nordrum, "Popular Internet of Things Forecast of 50 Billion Devices by 2020 Is Outdated," 2016, Accessed in January 20th 2018, https://spectrum.ieee.org/tech-talk/telecom/internet/popular-internet-of-things-forecast-of-50-billion-devices-by-2020-is-outdated.

[2] I. Yaqoob, E. Ahmed, I. A. T. Hashem et al., "Internet of things architecture: recent advances, taxonomy, requirements, and open challenges," *IEEE Wireless Communications Magazine*, vol. 24, no. 3, pp. 10–16, 2017.

[3] M. A. Ferrag, L. A. Maglaras, H. Janicke, J. Jiang, and L. Shu, "Authentication protocols for internet of things: a comprehensive survey," *Security and Communication Networks*, vol. 2017, Article ID 6562953, 41 pages, 2017.

[4] C. V. L. Mendoza, J. H. Kleinschmidt, R. Chen et al., "Trust management for SOA-based IoT and its application to service composition," *Journal of Machine Learning Research*, vol. 39, pp. 1–6, 2016.

[5] M. Nitti, R. Girau, and L. Atzori, "Trustworthiness management in the social internet of things," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1253–1266, 2014.

[6] J. Guo, I.-R. Chen, and J. J. P. Tsai, "A survey of trust computation models for service management in internet of things systems," *Computer Communications*, vol. 97, pp. 1–14, 2017.

[7] J.-H. Lee and H. Kim, "Security and privacy challenges in the internet of things [security and privacy matters]," *IEEE Consumer Electronics Magazine*, vol. 6, no. 3, pp. 134–136, 2017.

[8] C. V. L. Mendoza and J. H. Kleinschmidt, "Mitigating on-off attacks in the internet of things using a distributed trust management scheme," *International Journal of Distributed Sensor Networks*, vol. 2015, Article ID 859731, 2015.

[9] H. Hellaoui, A. Bouabdallah, and M. Koudil, "TAS-IoT: Trust-Based Adaptive Security in the IoT," in *Proceedings of the 41st IEEE Conference on Local Computer Networks (LCN '16)*, pp. 599–602, November 2016.

[10] O. Ben Abderrahim, M. H. Elhdhili, and L. Saidane, "TMCoI-SIOT: A trust management system based on communities of interest for the social internet of things," in *Proceedings of the 13th IEEE International Wireless Communications and Mobile Computing Conference (IWCMC '17)*, pp. 747–752, June 2017.

[11] F. Boustanifar and Z. Movahedi, "A trust-based offloading for mobile M2M communications," in *Proceedings of the Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld '16)*, pp. 1139–1143, IEEE, 2016.

[12] W. Li, H. Song, and F. Zeng, "Policy-based secure and trustworthy sensing for internet of things in smart cities," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1-1, 2017.

[13] Y. Yu, Z. Jia, W. Tao, B. Xue, and C. Lee, "An efficient trust evaluation scheme for node behavior detection in the internet of things," *Wireless Personal Communications*, vol. 93, no. 2, pp. 571–587, 2017.

[14] T. Zhang, L. Yan, and Y. Yang, "Trust evaluation method for clustered wireless sensor networks based on cloud model," *Wireless Networks*, pp. 1–21, 2016.

[15] S. M. Sony and S. B. Sasi, "On-Off attack management based on trust," in *Proceedings of the 2016 Online International Conference on Green Engineering and Technologies (IC-GET '16)*, pp. 1–4, 2016.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort et al., "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[17] "C. of Aarhus, Weather Data for the City of Aarhus in Denmark," 2016, Accessed in January 20th 2018, http://iot.ee.surrey.ac.uk:8080/datasets.html.

[18] W. Khreich, B. Khosravifar, A. Hamou-Lhadj, and C. Talhi, "An anomaly detection system based on variable N-gram features and one-class SVM," *Information and Software Technology*, vol. 91, pp. 186–197, 2017.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 4957045, 9 pages
https://doi.org/10.1155/2018/4957045

Research Article

Cryptanalysis of Compact-LWE and Related Lightweight Public Key Encryption

Dianyan Xiao [1] **and Yang Yu** [2]

[1] *Institute for Advanced Study, Tsinghua University, Beijing 100084, China*
[2] *Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China*

Correspondence should be addressed to Yang Yu; y-y13@mails.tsinghua.edu.cn

Received 15 December 2017; Revised 14 January 2018; Accepted 28 January 2018; Published 11 March 2018

Academic Editor: Ilsun You

In the emerging Internet of Things (IoT), lightweight public key cryptography plays an essential role in security and privacy protection. With the approach of quantum computing era, it is important to design and evaluate lightweight quantum-resistant cryptographic algorithms applicable to IoT. LWE-based cryptography is a widely used and well-studied family of postquantum cryptographic constructions whose hardness is based on worst-case lattice problems. To make LWE friendly to resource-constrained IoT devices, a variant of LWE, named Compact-LWE, was proposed and used to design lightweight cryptographic schemes. In this paper, we study the so-called Compact-LWE problem and clarify that under certain parameter settings it can be solved in polynomial time. As a consequence, our result leads to a practical attack against an instantiated scheme based on Compact-LWE proposed by Liu et al. in 2017.

1. Introduction

The Internet is changing from a network of conventional computers to a network of smart objects, that is, "things," including vehicles, electronics, implantable medical devices, and sensors. The trend of Internet of Things (IoT) makes the Internet more ubiquitous, but it simultaneously brings a series of challenges, such as monitoring [1], communication [2], and management [3]. Among all these challenges, security [4–6] is currently listed as a top concern. As the theoretical basis, cryptographic algorithms play a key role in achieving data confidentiality and integrity, authentication, and other security needs in IoT.

Currently, RSA and ECC cryptosystems have been implemented efficiently on resource-constrained devices [7, 8], which provides desirable security for IoT applications. However, these public key schemes are based on integer factorization or discrete logarithms, which are fragile under quantum cryptanalysis. To defense quantum attacks, NIST has launched the postquantum cryptography standardization. Lattice-based cryptography is viewed as a very promising postquantum alternative to classical cryptography due to its strong security guarantee, great performance and powerful functionality. It is becoming increasingly important to design and evaluate practical schemes based on well-studied lattice problems.

The *Learning With Errors* (LWE) problem, introduced by Regev [9], is one of the most popular lattice problems for cryptographic applications [10–13]. An LWE instance consists of a random matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ and a vector $\mathbf{b} = \mathbf{As} + \mathbf{e} \bmod q$, where the secret $\mathbf{s} \in \mathbb{Z}_q^n$ and the error $\mathbf{e} \in \mathbb{Z}^m$ are sampled from a certain distribution. The decision LWE problem is to distinguish the distribution of LWE instances from the uniform distribution over $\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m$, while the search version is to recover the secret $\mathbf{s}$ from LWE instances. In [9], the average-case LWE is proved as hard as certain worst-case lattice problems, which provides a solid theoretical grounding for LWE-based schemes.

However, LWE-based schemes are usually not efficient in practice. It seems infeasible to apply regular LWE-based cryptographic constructions to IoT directly, due to the constrained computing environments of smart devices. Thus it is critical to refine existing algorithms or develop new LWE-based cryptographic schemes for security protection using limited resources. So far, there are mainly two optimization

strategies: (1) introducing extra algebraic structures and (2) reducing the sizes of matrix or vector elements. Following the first one, some LWE variants, such as Ring-LWE [14] and Module-LWE [15], were developed and led to many practical schemes [16–18] and efficient implementations [19, 20]. Following the second strategy, some variants were proposed as well, including LWE with short secret or error [21–23] and LWE with compact matrix [24, 25]. Then, related cryptanalyses [26–29] provided concrete security estimations for the schemes based on these variants.

A recent instantiation of LWE-based encryption scheme with particularly aggressive parameter was proposed by Liu et al. [25] and presented as an invited talk at ACISP 2017 conference. The scheme is based on the so-called Compact-LWE and designed especially for resource-constrained IoT devices. As shown by experimental results, the scheme indeed achieves an excellent performance on small IoT devices. Subsequently, Bootle and Tibouchi gave a cryptanalysis of this scheme [29] by recovering the nonce in the encryption process with the help of lattice embedding technique. They pointed out that the security level was much lower than [25] claimed.

We took an insight into the Compact-LWE problem, an LWE variant with the random $\mathbf{A}$ selected from a small range, and discovered that two q-ary lattices defined by $\mathbf{A}$ have reduced bases of special patterns. We proved that the Compact-LWE problem can be solved in polynomial time under certain parameters, which is applied to analyze two concrete lightweight public key schemes proposed in [24, 25], respectively. We failed to attack the scheme of [24] due to its moderate parameters and successfully recovered plaintexts with 100% probability and within a very short time for the encryption scheme in [25]. Compared with the attack against the scheme of [25] in [29], our attack follows a different method and can be used to analyze general cryptographic constructions based on this kind of LWE variant.

The article is organized as follows. In Section 2, we recall some notations and basic facts used in our discussion. In Section 3, we introduce Compact-LWE and present our analysis. We describe a concrete attack against related Compact-LWE-based schemes in Section 4 and conclude in Section 5.

2. Preliminaries

2.1. Notations. For any positive integer q, we identify $\mathbb{Z}_q$ with the set $\{0, \ldots, q-1\}$. We denote by $[x]_q$ the remainder of x divided by q in $\mathbb{Z}_q$ and by $[x]_q$ the remainder in $\{-\lceil q/2\rceil, \ldots, q-1-\lceil q/2\rceil\}$. Let $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ be the Euclidean inner product and norm, respectively. The elements of $\mathbb{R}^m$ are viewed as column vectors. For any point $\mathbf{t} \in \mathbb{R}^m$ and $r > 0$, we denote by $\mathcal{B}_m(\mathbf{t}, r)$ the m-dimensional ball of radius r centered at $\mathbf{t}$.

2.2. Probability and Statistics. Let χ be a distribution over a discrete domain E. We write $X \hookleftarrow \chi$ to represent the random variable X that is sampled from the distribution χ. For a finite domain E, we denote by $U(E)$ the uniform distribution over E.

A function $f(n)$ is *negligible*, if $f(n) = o(n^{-c})$ for every fixed constant c. We generally denote by $\mathrm{negl}(n)$ as a negligible function with respect to n. We say that a probability is *overwhelming* if it is $1 - \mathrm{negl}(n)$, and a probability is *nonnegligible* if it is $\omega(n^{-c})$ for some constant c.

Definition 1. Given a distribution χ over $\mathbb{Q}^m$, we say that χ is (α, β)-confidence with respect to λ, if $\Pr[\|X\| \geq \alpha] \leq \mathrm{negl}(\lambda)$ and $\Pr[\|X\| \leq \beta] \geq 1/\mathrm{poly}(\lambda)$ for $X \hookleftarrow \chi$.

The parameter α describes an overwhelming confidence interval for χ with respect to λ, while β describes a nonnegligible confidence interval.

2.3. Lattices. A *lattice* $\mathcal{L}$ is a discrete additive subgroup of $\mathbb{R}^m$ and generated by a set of linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$, that is, $\mathcal{L} = \{\sum_{i=1}^n x_i \mathbf{b}_i \mid x_i \in \mathbb{Z} \text{ for any } i\}$. We call $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n) \in \mathbb{R}^{m\times n}$ a *basis* of $\mathcal{L}$ and write $\mathcal{L}$ as $\mathcal{L}(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ or $\mathcal{L}(\mathbf{B})$. The integer n is called the *rank* of $\mathcal{L}$. For any unimodular matrix $\mathbf{U} \in \mathbb{Z}^{n\times n}$, $\mathbf{BU}$ is also a basis of $\mathcal{L}$. The *span* of $\mathcal{L}$, denoted by $\mathrm{span}(\mathcal{L})$, is the linear space spanned by its basis. The first minimum of a lattice $\mathcal{L}$ is defined as $\lambda_1(\mathcal{L}) := \min_{\mathbf{v}\in\mathcal{L}\setminus\{\mathbf{0}\}}\|\mathbf{v}\|$.

We denote by $\mathbf{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*)$ the *Gram-Schmidt orthogonalization* of $\mathbf{B}$ where $\mathbf{b}_i^* = \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j}\mathbf{b}_j^*$ and $\mu_{i,j} = \langle \mathbf{b}_i, \mathbf{b}_j^*\rangle / \langle \mathbf{b}_j^*, \mathbf{b}_j^*\rangle$. The *volume* of $\mathcal{L}$ is defined as $\mathrm{vol}(\mathcal{L}) = \prod_{i=1}^n \|\mathbf{b}_i^*\|$ that is an invariant of $\mathcal{L}$ and independent of the choice of the basis.

The *dual* lattice of $\mathcal{L}$ is $\mathcal{L}^* := \{\mathbf{y} \in \mathrm{span}(\mathcal{L}) \mid \langle \mathbf{x}, \mathbf{y}\rangle \in \mathbb{Z}, \forall \mathbf{x} \in \mathcal{L}\}$. If $\mathbf{B}$ is a basis of $\mathcal{L}$, it is known that $\mathbf{D} = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-T}$ is a basis of $\mathcal{L}^*$. Furthermore, we have the following relation between the Gram-Schmidt orthogonalization of a basis and its dual.

Lemma 2. *Let $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ be an ordered basis of lattice $\mathcal{L}$ and $(\mathbf{d}_1, \ldots, \mathbf{d}_n)$ be its dual basis in reverse order (i.e., $\langle \mathbf{d}_i, \mathbf{b}_j\rangle = \delta_{i,n+1-j}$ where $\delta_{i,j}$ denotes Kronecker delta). Then $\mathbf{d}_i^* = \mathbf{b}_{n+1-i}^*/\|\mathbf{b}_{n+1-i}^*\|^2$ for $i = 1, \ldots, n$.*

Given a lattice $\mathcal{L}$ and a "reasonable" subset K of $\mathrm{span}(\mathcal{L})$, *Gaussian heuristic* says that the number of points in $K \cap \mathcal{L}$ is approximately $\mathrm{vol}(K)/\mathrm{vol}(\mathcal{L})$. From Gaussian heuristic, we would expect that $\lambda_1(\mathcal{L}) \approx \mathrm{vol}(\mathcal{L})^{1/n}\cdot \mathrm{GH}(n)$ where $\mathrm{GH}(n) = \mathrm{vol}(\mathcal{B}_n(\mathbf{0}, 1))^{-1/n} \approx \sqrt{n/2\pi e}$.

Lattice reduction is a powerful tool for cryptanalysis. LLL, invented by Lenstra et al. [30], is the first polynomial time lattice reduction algorithm. We now recall this classical reduction. For a detailed introduction, we refer to [31].

Definition 3 (LLL reduced basis). A basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$ is a δ-LLL reduced basis with $\delta \in (1/2, 1)$ if the following conditions hold:

(1) Size Reduced: $|\mu_{i,j}| \leq 1/2$ for $1 \leq j < i \leq n$.

(2) Lovász Condition: $\delta\|\mathbf{b}_i^*\| \leq \|\mathbf{b}_{i+1}^* + \mu_{i+1,i}\mathbf{b}_i^*\|$ for $1 \leq i < n$.

Then we immediately get the following property of LLL reduced bases.

Lemma 4. *Let* $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ *be a* δ*-LLL reduced basis. For any* $1 \le j < i \le n$*, then*

$$\left\|\mathbf{b}_i^*\right\| \ge \gamma^{j-i} \left\|\mathbf{b}_j^*\right\|, \tag{1}$$

where $\gamma = 1/\sqrt{\delta^2 - 1/4}$.

3. Compact-LWE and Its Weak Instances

In this section, we will introduce an LWE variant named Compact-LWE and report on an attack against certain Compact-LWE instances. A formal definition of Compact-LWE is given as follows.

Definition 5. Let q, m, n, b be positive integers and χ be a distribution over $\mathbb{Z}^m$. Given $\mathbf{s} \hookleftarrow U(\mathbb{Z}_q^n)$, the Compact-LWE$_{q,m,n,b,\chi}$ problem is to recover $\mathbf{s}$ from $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \bmod q)$ where $\mathbf{A} \hookleftarrow U(\mathbb{Z}_b^{m\times n})$ and $\mathbf{e} \hookleftarrow \chi$.

Compared with classical LWE, the sizes of elements of $\mathbf{A}$, namely b, can be less than the modulus q. Thanks to this modification, Compact-LWE-based schemes are of smaller public key sizes and better efficiency than original LWE-based schemes. Thus Compact-LWE seems friendly to lightweight cryptography and constrained devices.

3.1. Structures of q-Ary Lattices in Compact-LWE. We introduce two m-dimensional q-ary lattices which are widely used in the cryptanalysis of LWE. The first lattice, denoted by $\mathscr{L}_q(\mathbf{A})$, is generated by the columns of $\mathbf{A}$ and $q\cdot\mathbf{I}_m$ and defined as

$$\begin{aligned}&\mathscr{L}_q(\mathbf{A})\\ &\quad\coloneqq \left\{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{x} = \mathbf{A}\mathbf{y} \bmod q \text{ for some } \mathbf{y} \in \mathbb{Z}^n\right\},\end{aligned} \tag{2}$$

The second lattice $\mathscr{L}_q^\perp(\mathbf{A})$ is formed by all integer vectors "orthogonal" (modulo q) to the columns of $\mathbf{A}$, which is

$$\mathscr{L}_q^\perp(\mathbf{A}) \coloneqq \left\{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}^T\mathbf{x} = \mathbf{0} \bmod q\right\}. \tag{3}$$

As shown in [10], these two lattices are duals scaled by a factor:

$$\mathscr{L}_q^\perp(\mathbf{A}) = q \cdot \mathscr{L}_q(\mathbf{A})^*. \tag{4}$$

By running LLL algorithm with input $(\mathbf{A} \mid q\cdot\mathbf{I}_m)$, one can obtain a basis of $\mathscr{L}_q(\mathbf{A})$. For $\mathbf{A}$ in the compact setting, the LLL reduced basis is of a special structure.

Lemma 6. *Let* $\mathbf{A} \in \mathbb{Z}_b^{m\times n}$ *where* $m \ge n + 2\pi e$ *and* $b \le q^{(m-n)/m}/\sqrt{m}$*. Let* $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m)$ *be the basis of* $\mathscr{L}_q(\mathbf{A})$ *obtained by running LLL with parameter* δ *on* $(\mathbf{A} \mid q\cdot\mathbf{I}_m)$*. Under Gaussian heuristic, then, for* $\gamma = 1/\sqrt{\delta^2 - 1/4}$*,*

(1) $\mathscr{L}(\mathbf{b}_1, \ldots, \mathbf{b}_n) = \mathscr{L}(\mathbf{A})$ *and* $\|\mathbf{b}_i^*\| \le b\sqrt{m}$ *for* $1 \le i \le n$*;*

(2) $\|\mathbf{b}_i^*\| > q\gamma^{n-i+1}(b\sqrt{m})^{-n/(m-n)}$ *for* $n+1 \le i \le m$.

Proof. Let $\phi : \mathbb{Z}^m/\mathscr{L}_q^\perp(\mathbf{A}) \to \mathbb{Z}_q^n$ be the homomorphism mapping $\mathbf{v} + \mathscr{L}_q^\perp(\mathbf{A})$ to $\mathbf{A}^T\mathbf{v} \bmod q$. It can be verified that ϕ is injective, then we have

$$\mathrm{vol}\left(\mathscr{L}_q^\perp(\mathbf{A})\right) \le q^n. \tag{5}$$

Together with (4), it follows that

$$\mathrm{vol}\left(\mathscr{L}_q(\mathbf{A})\right) \ge q^{m-n}. \tag{6}$$

Let $\boldsymbol{\pi}_{\mathbf{A}}(\cdot)$ denote the projection to the orthogonal complement of span$(\mathbf{A})$. Considering the projected lattice $\mathscr{L}'$ generated by $\boldsymbol{\pi}_{\mathbf{A}}(q\cdot\mathbf{I}_m)$, the dimension of $\mathscr{L}'$ is $(m-n)$. Combined with (6), we have

$$\mathrm{vol}\left(\mathscr{L}'\right) \ge \frac{\mathrm{vol}\left(\mathscr{L}_q(\mathbf{A})\right)}{\mathrm{vol}\left(\mathscr{L}(\mathbf{A})\right)} \ge \frac{q^{m-n}}{\mathrm{vol}\left(\mathscr{L}(\mathbf{A})\right)}. \tag{7}$$

Since $\mathrm{vol}(\mathscr{L}(\mathbf{A})) = \prod_{i=1}^n \|\mathbf{a}_i^*\| \le \prod_{i=1}^n \|\mathbf{a}_i\| \le (b\sqrt{m})^n$, it follows that

$$\mathrm{vol}\left(\mathscr{L}'\right) \ge \frac{q^{m-n}}{(b\sqrt{m})^n}. \tag{8}$$

By Gaussian heuristic, we have that

$$\begin{aligned}\lambda_1\left(\mathscr{L}'\right) &\approx \sqrt{\frac{m-n}{2\pi e}} \cdot \left(\mathrm{vol}\left(\mathscr{L}'\right)\right)^{1/(m-n)}\\ &\ge q \cdot (b\sqrt{m})^{-n/(m-n)}.\end{aligned} \tag{9}$$

A straightforward computation leads to that $\lambda_1(\mathscr{L}') \ge b\sqrt{m} \ge \max_{i=1}^n \|\mathbf{a}_i^*\|$. It is known that the maximum of the Gram-Schmidt norms would never increase in LLL algorithm. Thus, Lovász condition always holds for the nth and $(n+1)$th vectors during LLL, which means that these two vectors would never be swapped. In other words, running LLL on $(\mathbf{A} \mid q\cdot\mathbf{I}_m)$ is equivalent to running LLL on $\mathbf{A}$ and $\boldsymbol{\pi}_{\mathbf{A}}(q\cdot\mathbf{I}_m)$, respectively. Consequently, we have $\mathscr{L}(\mathbf{b}_1, \ldots, \mathbf{b}_n) = \mathscr{L}(\mathbf{A})$ and $\|\mathbf{b}_i^*\| \le \max_{i=1}^n \|\mathbf{a}_i^*\| \le b\sqrt{m}$ for $1 \le i \le n$.

For the second inequality, Lemma 4 yields that

$$\left\|\mathbf{b}_{n+i}^*\right\| \ge \left\|\mathbf{b}_{n+1}^*\right\| \gamma^{1-i} > q\gamma^{1-i}(b\sqrt{m})^{-n/(m-n)}, \tag{10}$$

because $\|\mathbf{b}_{n+1}^*\| \ge \lambda_1(\mathscr{L}')$. We now complete the proof. □

Remark 7. Experimental results coincide with Lemma 6. Under parameter settings $(q, m, n) = (2^{20}, 300, 120)$, we generated 20 instances for each b ranging from 2 to 2^{18}. Figure 1 illustrates the average profile of $\mathbf{B}$, where the first n $\mathbf{b}_i^*$'s are relatively short when b is small. We notice that the slope of $\{\log_2\|\mathbf{b}_i^*\|\}_{i=n+1}^m$ is less than the theoretical bound $\log_2\gamma \approx 0.2172$, which can be explained by the better performance of LLL in practice than the theoretical prediction. Figure 2 shows the gap between $\|\mathbf{b}_{n+1}^*\|$ and $\|\mathbf{b}_n^*\|$, which is narrowing as b increases. It is worth noting that when $b < q^{m/(m-n)}/\sqrt{m}$ (the bound in Lemma 6 marked by the dashed line), the gap is quite significant.

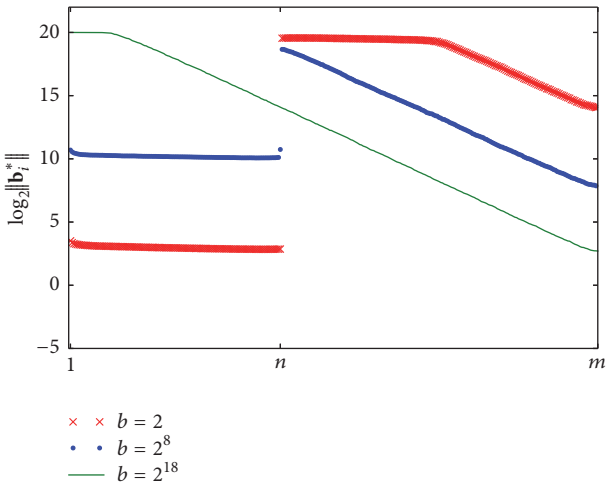


Figure 1: Experimental measure of $\{\log_2\|\mathbf{b}_i^*\|\}_{i=1}^m$ for different b.

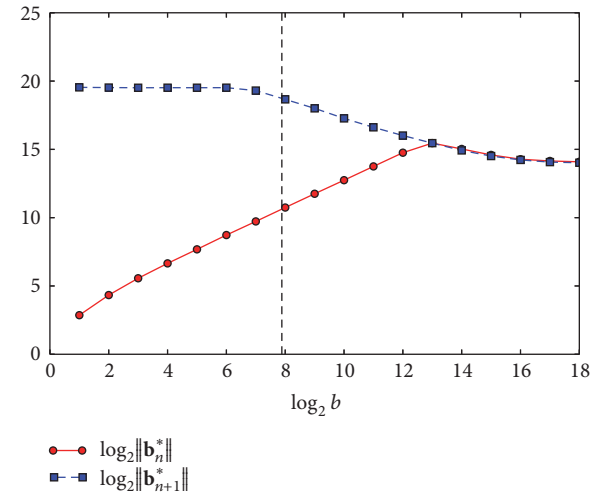


Figure 2: Experimental measure of the gap between $\log_2\|\mathbf{b}_n^*\|$ and $\log_2\|\mathbf{b}_{n+1}^*\|$.

Lemma 8. *Let $m \geq n + 2\pi e$ and $b \leq q^{(m-n)/m}/\sqrt{m}$. Let $\mathbf{A} \in \mathbb{Z}_b^{m\times n}$ and $\delta \in (1/2, 1)$. There exists a basis of $\mathcal{L}_q^\perp(\mathbf{A})$, denoted by $\mathbf{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_m)$, satisfying the following conditions under Gaussian heuristic:*

(1) $\mathcal{L}(\mathbf{d}_1, \ldots, \mathbf{d}_{m-n}) = \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{x}^T\mathbf{A} = \mathbf{0}\}$,

(2) $\|\mathbf{d}_i\| \leq \sqrt{(i+3)/4}\cdot\gamma^{m-n}(b\sqrt{m})^{n/(m-n)}$ *for* $1 \leq i \leq m-n$,

(3) $\|\mathbf{d}_i\| \geq \sqrt{n/2\pi e}(q/b\sqrt{m})$ *for* $m - n + 1 \leq i \leq m$,

where $\gamma = 1/\sqrt{\delta^2 - 1/4}$. This basis can be obtained in polynomial time.

Proof. Let $\mathbf{B} = (\mathbf{B}_1 \mid \mathbf{B}_2)$ be the LLL reduced basis of $\mathcal{L}_q(\mathbf{A})$ defined in Lemma 6 where $\mathcal{L}(\mathbf{B}_1) = \mathcal{L}(\mathbf{A})$. Let $\mathbf{U}$ be a matrix such that $\mathbf{U}^T\mathbf{B} = q\mathbf{I}_m$. Then, from (4), $\mathbf{U}$ is a basis of $\mathcal{L}_q^\perp(\mathbf{A})$. Let $\mathbf{U} = (\mathbf{U}_1 \mid \mathbf{U}_2) = (\mathbf{u}_m, \ldots, \mathbf{u}_1)$ where $\mathbf{U}_1 \in \mathbb{Z}^{m\times n}$.

Let $\mathcal{L}^\perp(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{x}^T\mathbf{A} = \mathbf{0}\}$. We claim that $\mathcal{L}(\mathbf{U}_2) = \mathcal{L}^\perp(\mathbf{A})$. It is easy to observe that $\mathcal{L}^\perp(\mathbf{A}) = \mathcal{L}^\perp(\mathbf{B}_1)$ where $\mathcal{L}^\perp(\mathbf{B}_1) = \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{x}^T\mathbf{B}_1 = \mathbf{0}\}$. On one hand, we have $\mathcal{L}(\mathbf{U}_2) \subseteq \mathcal{L}^\perp(\mathbf{B}_1)$ since $\mathbf{U}_2^T\mathbf{B}_1 = \mathbf{0}$. On the other hand, for arbitrary $\mathbf{v} \in \mathcal{L}^\perp(\mathbf{B}_1) \subseteq \mathcal{L}_q^\perp(\mathbf{A})$, there exists a unique vector pair $(\mathbf{z}_1, \mathbf{z}_2)$ such that $\mathbf{v} = \mathbf{U}_1\mathbf{z}_1 + \mathbf{U}_2\mathbf{z}_2$. Since $\mathbf{U}_1^T\mathbf{B}_1 = q\mathbf{I}_n$, we have $\mathbf{v}^T\mathbf{B}_1 = q\mathbf{z}_1^T = \mathbf{0}$ and then $\mathbf{v} \in \mathcal{L}(\mathbf{U}_2)$. Therefore, it holds that $\mathcal{L}(\mathbf{U}_2) = \mathcal{L}^\perp(\mathbf{A})$.

We run size reduction algorithm on $(\mathbf{u}_1, \ldots, \mathbf{u}_m)$ (vectors of $\mathbf{U}$ in reverse order) and obtain a new basis of $\mathcal{L}_q^\perp(\mathbf{A})$, denoted by $\mathbf{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_m)$. Size reduction can be done within polynomial time; thus it suffices to prove the last two conditions hold for $\mathbf{D}$. From Lemmas 2 and 6, we have that, for $i = 1, \ldots, m - n$,

$$\|\mathbf{d}_i^*\| = \|\mathbf{u}_i^*\| = \frac{q}{\|\mathbf{b}_{m+1-i}^*\|} \leq \gamma^{m-n-i}\left(b\sqrt{m}\right)^{n/(m-n)}, \tag{11}$$

and then

$$\begin{aligned}\|\mathbf{d}_i\| &\leq \sqrt{\|\mathbf{d}_i^*\|^2 + \frac{1}{4}\sum_{j=1}^{i-1}\|\mathbf{d}_j^*\|^2} \\ &\leq \sqrt{\frac{i+3}{4}}\gamma^{m-n}\left(b\sqrt{m}\right)^{n/(m-n)}.\end{aligned} \tag{12}$$

Let $\mathcal{L}'' = \mathcal{L}(\boldsymbol{\pi}_{m-n}(\mathbf{d}_{m-n+1}), \ldots, \boldsymbol{\pi}_{m-n}(\mathbf{d}_m))$ where $\boldsymbol{\pi}_{m-n}(\cdot)$ is the projection to the orthogonal complement of $\mathrm{span}(\mathbf{d}_1, \ldots, \mathbf{d}_{m-n})$. Observing that $\mathrm{vol}(\mathcal{L}_q(\mathbf{A}))\mathrm{vol}(\mathcal{L}_q^\perp(\mathbf{A})) = q^m$ and $\|\mathbf{d}_i^*\| = q/\|\mathbf{b}_{m+1-i}^*\|$ for $i \leq m - n$, together with Lemma 6, we have

$$\mathrm{vol}\left(\mathcal{L}''\right) = \frac{\mathrm{vol}\left(\mathcal{L}_q^\perp(\mathbf{A})\right)}{\prod_{i=1}^{m-n}\|\mathbf{d}_i^*\|} = \frac{q^n}{\prod_{i=1}^{n}\|\mathbf{b}_i^*\|} \geq \frac{q^n}{\left(b\sqrt{m}\right)^n}. \tag{13}$$

On the basis of Gaussian heuristic, we conclude that, for $m - n + 1 \leq i \leq m$,

$$\|\mathbf{d}_i\| \geq \|\boldsymbol{\pi}_{m-n}(\mathbf{d}_i)\| \geq \lambda_1\left(\mathcal{L}''\right) \geq \sqrt{\frac{n}{2\pi e}}\frac{q}{b\sqrt{m}}. \tag{14}$$

We now complete the proof. □

Remark 9. We ran experiments under parameters $(q, m, n) = (2^{20}, 300, 120)$ and tested 20 instances for each b ranging from 2 to 2^{18}. Figure 3 provides a geometric intuition of $\mathbf{D}$. There also exists a large gap between $\|\mathbf{d}_{m-n}\|$ and $\|\mathbf{d}_{m-n+1}\|$ when b is small. As illustrated in Figure 4, the gap between $\|\mathbf{d}_{m-n}\|$ and $\|\mathbf{d}_{m-n+1}\|$ is shrinking as b grows. However, when $b < q^{m-n/m}/\sqrt{m}$ (marked by the dashed line), the length of $\mathbf{d}_{m-n}$ is far less than q.

3.2. Attack Against Weak Compact-LWE Instances. Figure 1 illustrates a staircase-shaped profile of the basis of $\mathcal{L}_q(\mathbf{A})$. Exploiting this feature, we can prove that it is possible to efficiently recover a candidate error whose norm is close to that of the original error for certain parameters. The following lemma will be used in the later discussion.

Lemma 10. *Let $\mathcal{L} \subseteq \mathbb{R}^m$ be a lattice of rank n and $\mathbf{B}$ be a basis of $\mathcal{L}$. Let $\mathbf{t} \in \mathbb{R}^m$ and $\mathrm{dist}(\mathbf{t}, \mathcal{L}) = \min_{\mathbf{v}\in\mathcal{L}}\|\mathbf{t} - \mathbf{v}\|$. If*

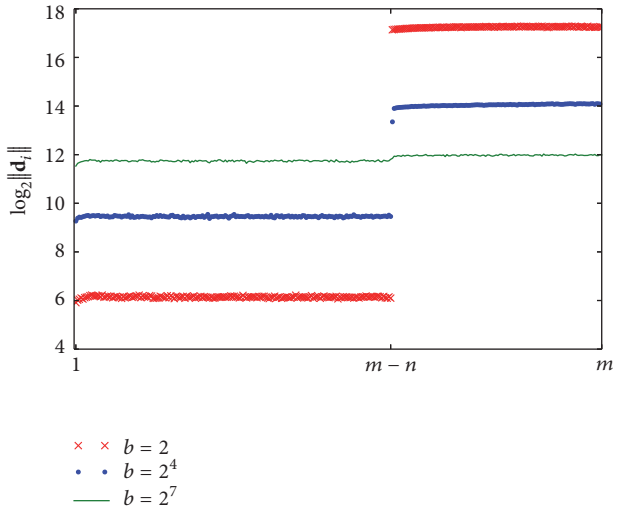


Figure 3: Experimental measure of $\{\log_2\|\mathbf{d}_i\|\}_{i=1}^m$ for different b.

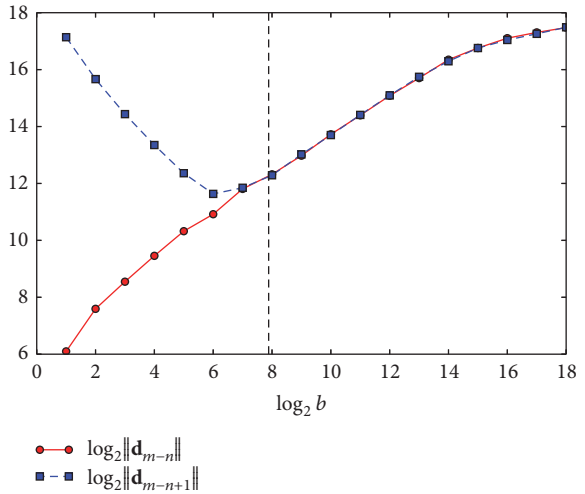


Figure 4: Experimental measure of the gap between $\log_2\|\mathbf{d}_{m-n}\|$ and $\log_2\|\mathbf{d}_{m-n+1}\|$.

$\mathrm{dist}(\mathbf{t}, \mathscr{L}) \le r < (1/2)\min_{i=1}^n\|\mathbf{b}_i^*\|$, *then there exists a unique vector in* $\mathscr{B}(\mathbf{t}, r) \cap \mathscr{L}$.

Proof. We denote by $\mathbf{v} = \sum_{i=1}^n v_i\mathbf{b}_i$ the vector output by Babai's nearest plane algorithm [32] on the lattice $\mathscr{L}$ and target vector $\mathbf{t}$. Assume, by contradiction, that $\mathbf{v}' \neq \mathbf{v}$ is another vector in $\mathbf{B}(\mathbf{t}, r) \cap \mathscr{L}$ and $\mathbf{v}' = \sum_{i=1}^n v_i'\mathbf{b}_i$. Let k be the largest index such that $v_k \neq v_k'$. According to the process of Babai's algorithm, we conclude that

$$\left\|\mathbf{v}' - \mathbf{t}\right\| \ge \frac{1}{2}\left\|\mathbf{b}_k^*\right\| > r, \tag{15}$$

which implies a contradiction. □

Next we demonstrate a class of provably weak instances of Compact-LWE and also an attack aiming at them.

Theorem 11. *Let* q, m, n, b, r *be positive integers satisfying* $b \le q^{(m-n)/m}/\sqrt{m}$ *and* $(\sqrt{mn}/2)b < r \le (q/2)\gamma^{n-m+1}(b\sqrt{m})^{-n/(m-n)}$ *where* $\gamma = 1/\sqrt{\delta^2 - 1/4}$ *for* $\delta \in (1/2, 1)$ *and a constant* $c > 0$. *Let* χ *be a* $(r, (1/2)\sqrt{4r^2 - mnb^2})$*-confidence distribution. Under Gaussian heuristic, there exists a probabilistic polynomial time algorithm solving Compact-LWE*$_{q,m,n,b,\chi}$.

Proof. Given a random sample $(\mathbf{A}, \mathbf{b} = \mathbf{As} + \mathbf{e} \bmod q)$, we can obtain a basis of $\mathscr{L}_q(\mathbf{A})$, denoted by $\mathbf{B}$, by applying LLL algorithm with parameter δ on $(\mathbf{A} \mid q\mathbf{I}_m)$. Exploiting Babai's algorithm on $\mathscr{L}_q(\mathbf{A})$ and target vector $\mathbf{b}$, we get a pair of solution $(\mathbf{s}', \mathbf{e}')$. We are to prove that $(\mathbf{s}', \mathbf{e}')$ is legal for Compact-LWE, that is, $\|\mathbf{e}'\| \le r$, with nonnegligible probability.

From Lemma 6, we get that $r < (1/2)\|\mathbf{b}_i^*\|$ for $n+1 \le i \le m$. We denote by $\boldsymbol{\pi}_n(\cdot)$ the projection to the orthogonal complement of $\mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_n)$. Let $\mathscr{L}' = \boldsymbol{\pi}_n(\mathscr{L})$ and $\mathbf{b}' = \boldsymbol{\pi}_n(\mathbf{b})$. Lemma 10 shows that there exists a unique vector in $\mathscr{B}(\mathbf{b}', r) \cap \mathscr{L}'$, namely, $\boldsymbol{\pi}_n(\mathbf{e}) = \boldsymbol{\pi}_n(\mathbf{e}')$. Then we have

$$\begin{aligned}\left\|\mathbf{e}'\right\|^2 &= \left\|\mathbf{e}' - \boldsymbol{\pi}_n\left(\mathbf{e}'\right)\right\|^2 + \left\|\boldsymbol{\pi}_n\left(\mathbf{e}'\right)\right\|^2 \\ &\le \frac{1}{4}\sum_{i=1}^n\left\|\mathbf{b}_i^*\right\|^2 + \|\mathbf{e}\|^2 \le \frac{mnb^2}{4} + \|\mathbf{e}\|^2.\end{aligned} \tag{16}$$

Since χ is $(r, (1/2)\sqrt{4r^2 - mnb^2})$-confidence, it implies that $\|\mathbf{e}\| \le (1/2)\sqrt{4r^2 - mnb^2}$ with nonnegligible probability. Thus the probability of $\|\mathbf{e}'\| \le r$ is nonnegligible. □

Remark 12. In such weak instances, it can be verified that

$$q > \sqrt{n} \cdot \gamma^{m-n-1}\left(b\sqrt{m}\right)^{m/(m-n)}, \tag{17}$$

and thus parameters are overstretched [33, 34]. The inequalities given in Lemma 6 follow the worst-case result of LLL, but LLL behaves much better in practice. Hence our attack may apply to more Compact-LWE instances. Moreover, note that, for usual LWE distribution χ such as discrete Gaussian, it is easy to set α, β such that χ is (α, β)-confidence.

4. Attack against Compact-LWE-Based Schemes

In this section, our analysis of Section 3 is applied to attack concrete Compact-LWE-based lightweight encryption schemes. We successfully recover the plaintexts in IoT-oriented public key encryption proposed by Liu et al. in [25] following a totally different way with [29]. However, we fail to give an effective cryptanalysis of the binary LWE-based lightweight encryption in [24].

4.1. Liu et al.'s Compact-LWE-Based Scheme. Firstly, we briefly recall the public key encryption in [25]. The scheme is

specified by a tuple of public parameters (q, n, m, t, w, b) satisfying

$$\begin{aligned} n+1 &< m < n^2, \\ 2\log_2 b &< n < b, \\ (2b\log_2 b + 2)\cdot b &< q. \end{aligned} \tag{18}$$

We list below three main algorithms: key generation **Gen**, encryption **Enc**$(\cdot)$, and decryption **Dec**$(\cdot)$.

(i) **Gen**: sample $\mathbf{s} \hookleftarrow U(\mathbb{Z}_q^n)$, and choose sk, r, p from $\mathbb{Z}_q$ satisfying

$$\begin{aligned} t &\le p, \\ b &< r, \\ sk\,(t-1) + wrp &< q, \end{aligned} \tag{19}$$

and sk, p, q are pairwise coprime. Sample $\mathbf{A} \hookleftarrow U(\mathbb{Z}_b^{m\times n})$ and $\mathbf{e} \hookleftarrow U(\mathbb{Z}_r^m)$. Let $sk_q^{-1} \in \mathbb{Z}_q$ such that $sk \cdot sk_q^{-1} = 1 \bmod q$. Output $\mathbf{SK} = (\mathbf{s}, sk, r, p)$ as the secret key and $\mathbf{PK} = (\mathbf{A}, \mathbf{pk} = \mathbf{As} - sk_q^{-1} \cdot p \cdot \mathbf{e} \bmod q)$ as the public key.

(ii) **Enc**$(v \in \mathbb{Z}_t, \mathbf{PK})$: uniformly and independently sample $i_1, \ldots, i_w \hookleftarrow U(\mathbb{Z}_m)$, and calculate $\mathbf{c}' = \sum_{j=1}^{w} (\mathbf{a}_{i_j}, pk_{i_j}) \bmod q$ where $(\mathbf{a}_i, pk_i)$ is the ith row of $\mathbf{PK}$. Let $\mathbf{c}' = (\mathbf{a}, pk)$, output $\mathbf{c} = (\mathbf{a}, v - pk \bmod q)$ as the ciphertext.

(iii) **Dec**$(\mathbf{c} = (\mathbf{a}, x), \mathbf{SK})$: calculate $c = \langle \mathbf{a}, \mathbf{s} \rangle + x \bmod q$ and then calculate $skv = sk \cdot c \bmod q$. Let sk_p^{-1} be the multiplicative inverse of sk modulo p. Output the plaintext $v = [sk_p^{-1} \cdot skv]_p$.

In [25], the authors also proposed concrete parameters to instantiate the scheme. The parameters are listed as follows:

(i) Public parameters: $(q, n, m, t, w, b) = (2^{32}, 13, 74, 2^{16}, 86, 16)$

(ii) Secret parameters: $(sk, p, r) = (2x+1, t+2y+1, \le (q-1-sk\cdot(t-1))/(w\cdot p))$ where $(x, y) \in [0, 50]\times[0, 500]$ or $[0, 500] \times [0, 50]$.

4.2. Attack against Liu et al.'s Scheme. According to the average profile of bases shown in Lemmas 6 and 8 under the parameters $(q, m, n, b) = (2^{32}, 74, 13, 16)$ (see Figures 5 and 6) as suggested in [25], it seems that Liu et al.'s scheme is fragile. We propose a new attack against Liu et al.'s scheme with the help of our analysis towards Compact-LWE in Section 3.

Our attack consists of two steps: *guessing the mask coefficient* (sk, p) and *recovering the plaintext*. In the first step, one can almost determine the pair (sk, p) (sometimes together with several possible candidate pairs) by enumerating and checking. In the second step, combined with (sk, p), one can calculate a pair of legal solution $(\mathbf{s}', \mathbf{e}')$ to the Compact-LWE problem and recover the plaintext as well. Now we are to show the details of our attack.

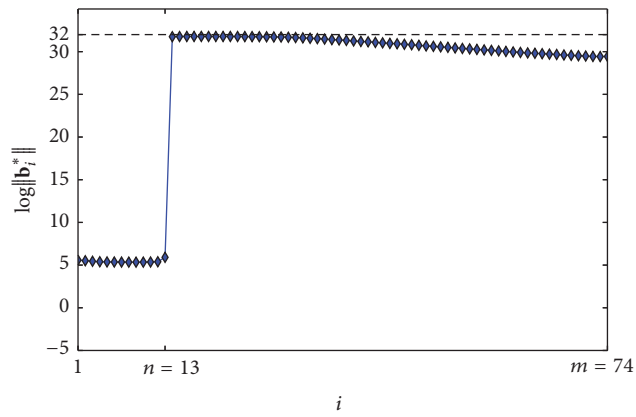


Figure 5: Experimental measure of $\{\log_2\|\mathbf{b}_i^*\|\}_{i=1}^m$.

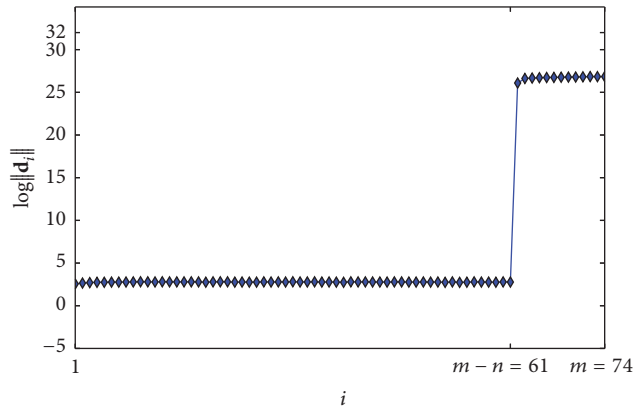


Figure 6: Experimental measure of $\{\log_2\|\mathbf{d}_i\|\}_{i=1}^m$.

Step 1 (guessing the mask coefficient (sk, p)). Firstly, we prove that it is possible to recover efficiently the secret parameters sk and p only from the public key $\mathbf{PK} = (\mathbf{A}, \mathbf{pk})$.

Let $\mathbf{D} = (\mathbf{d}_1, \ldots, \mathbf{d}_n)$ be a basis of $\mathcal{L}_q^{\perp}(\mathbf{A})$ as described in Lemma 4 with $\delta = \sqrt{0.99}$. Let $p_q^{-1} \in \mathbb{Z}$ such that $p_q^{-1} \cdot p = 1 \bmod q$; then we have that

$$sk \cdot p_q^{-1} \cdot \langle \mathbf{d}_i, \mathbf{pk} \rangle = -\langle \mathbf{d}_i, \mathbf{e} \rangle \bmod q. \tag{20}$$

Since $\|\mathbf{e}\| \le r\sqrt{m} \le (q-1-sk\cdot(t-1))/(w\cdot p)\sqrt{m}$ and $\|\mathbf{d}_i\|$ is also small when $1 \le i \le m-n$, a routine computation yields that $|\langle \mathbf{d}_i, \mathbf{e} \rangle| \le \|\mathbf{d}_i\| \cdot \|\mathbf{e}\| < q/2$ under the parameter setting suggested in [25]. Then it holds that

$$\begin{aligned} \left| \left[sk \cdot p_q^{-1} \cdot \langle \mathbf{d}_i, \mathbf{pk} \rangle \right]_q \right| &= \left| \langle \mathbf{d}_i, \mathbf{e} \rangle \right| \\ &\le \frac{q-1-sk\cdot(t-1)}{w\cdot p} \sqrt{m} \cdot \|\mathbf{d}_i\|, \end{aligned} \tag{21}$$

for $i = 1, \ldots, m-n$. We try all possible pairs $(sk, p) \in \{2x+1 \mid x \in [0, 50] \cap \mathbb{Z}\} \times \{t + 2y + 1 \mid y \in [0, 500] \cap \mathbb{Z}\}$ and check inequality (21) for $\mathbf{d}_1, \ldots, \mathbf{d}_{m-n}$, respectively; then (sk, p) is viewed as a candidate when it holds for all $i = 1, \ldots, m-n$.

Experiments indicate that this step can indeed determine the unique correct (sk, p) at most times, and output a few

Table 1: Experimental results.

Parameter	Time for Step 1	Time for Step 2	Time for attack	Determining unique (sk, p)	Success rate
ParaA	0.61 s	1.94 s	2.55 s	67%	100%
ParaB	0.60 s	1.86 s	2.46 s	71%	100%

Table 2: Experimental results for optimized attack.

Parameter	Time for Step 1	Time for Step 2	Time for attack	Success rate
ParaA	0.57 s	1.14 s	1.71 s	100%
ParaB	0.57 s	1.14 s	1.71 s	100%

candidates (including the correct pair) of the form $(\lambda \cdot sk, p)$ for small factor λ at other times. Therefore, by guessing sk and p, we can actually remove the secret scaling factor and transform **PK** into a standard Compact-LWE sample.

Step 2 (recovering the plaintext). After the previous step, we obtain one or more (sk, p) pairs. Next, we are to show how to recover the plaintext combined with the ciphertext.

Let $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m)$ be the basis of $\mathscr{L}_q(\mathbf{A})$ described in Lemma 6. Given a candidate pair of the mask coefficient $(sk', p') = (\lambda \cdot sk, p)$ where λ is small, let $\mathbf{pk}' = sk' \cdot p_q^{-1} \cdot \mathbf{pk} \bmod q$. Running Babai's algorithm on $\mathscr{L}_q(\mathbf{A})$ and target vector $\mathbf{pk}'$, we obtain $\mathbf{v}' \in \mathscr{L}_q(\mathbf{A})$ and let $\mathbf{e}' = \mathbf{v}' - \mathbf{pk}'$. We observe that the distance from $\mathbf{pk}'$ to $\mathscr{L}_q(\mathbf{A})$ is at most $\|\lambda \cdot \mathbf{e}\|$, and $\|\lambda \cdot \mathbf{e}\| \le \lambda \cdot r\sqrt{m} < (1/2)\|\mathbf{b}_i^*\|$ for $n+1 \le i \le m$. Following a similar argument of (16) in Theorem 11, we know that

$$\left\|\mathbf{e}'\right\|^2 \le \|\lambda \cdot \mathbf{e}\|^2 + \frac{n}{4}\max_{1\le i\le n}\left\|\mathbf{b}_i^*\right\|^2 \le \left(\lambda^2 r^2 + \frac{n}{4}b^2\right)m. \quad (22)$$

Let $\mathbf{s}' \in \mathbb{Z}_q^m$ such that $\mathbf{v}' = sk' \cdot p_q^{-1}\mathbf{A}\mathbf{s}' \bmod q$, then $\mathbf{pk} = \mathbf{A}\mathbf{s}' - sk_q'^{-1} \cdot p' \cdot \mathbf{e}' \bmod q$ where $sk_q'^{-1} \cdot sk_q' = 1 \bmod q$. Exploiting the substitute secret key $(sk', p', \mathbf{s}', \mathbf{e}')$, we can decrypt the ciphertext $\mathbf{c} = (\mathbf{a}, x)$ as follows:

(1) Calculate $c' = [\langle \mathbf{a}, \mathbf{s}' \rangle + x]_q$.

(2) Calculate $skv' = \lfloor sk' \cdot c' \rceil_q$.

(3) Return $v' = [sk_p'^{-1} \cdot skv']_p$ where $sk_p'^{-1} \cdot sk' = 1 \bmod p$.

We now explain why the ciphertext can be decrypted correctly by above algorithm. It can be checked that $sk' \cdot c' = sk' \cdot v + p' \sum_{j=1}^{w} e'_{i_j} \bmod q$. Noticing that $\|\mathbf{e}'\|$ is well-bounded and some coordinates e'_i of $\mathbf{e}'$ could be negative, we may assert that $sk' \cdot v + p' \sum_{j=1}^{w} e'_{i_j} \in (-q/2, q/2)$ with a high probability. Thus the term $sk' \cdot v + p' \sum_{j=1}^{w} e'_{i_j}$ can be recovered (as skv') correctly, which implies that v' is the plaintext.

Experiments show that the plaintext can indeed be recovered, even if $(sk', p') = (\lambda \cdot sk, p)$ for some $\lambda \ne 1$. When λ is large, the norm of $\mathbf{e}'$ may exceed the upper bound $r\sqrt{m}$, which implies that (sk', p') is a wrong guess. Therefore, we may eliminate some wrong guesses of (sk', p') further in this step. Moreover, one may also try more middle terms such as $skv' = \lfloor sk' \cdot c' \rceil_q, \lfloor sk' \cdot c' \rceil_q \pm q$ during the "decryption" to ensure that the correct value of $sk' \cdot v + p' \sum_{j=1}^{w} e'_{i_j}$ is not missed. However, from our experimental results, we observe that trying only one $skv' = \lfloor sk' \cdot c' \rceil_q$ is enough to recover the plaintext in practice.

Experimental Results. We implemented our attack using the NTL library [35]. All experiments were run on a single core of a 3.40 GHz Core i7-4930K PC.

We follow the parameter setting suggested in [25]: the public parameters $(q, n, m, t, w, b) = (2^{32}, 13, 74, 2^{16}, 86, 16)$ and the secret parameters $(sk, p) = (2x + 1, t + 2y + 1)$. We denote the cases $(x, y) \in [0, 50] \times [0, 500]$ and $[0, 500] \times [0, 50]$ by **ParaA** and **ParaB**, respectively. For **ParaA** and **ParaB**, we respectively generated 100 random instances and calculated the ciphertexts of 100 random messages for each instance. Then we ran the attack on these 10000 ciphertexts. Experimental results are given in Table 1.

As mentioned before, we may obtain several (sk', p') pairs in Step 1. In fact, it suffices to take use of the pair with the minimal sk' to recover the plaintext. This observation leads to an optimization of the attack: one may search (sk', p') in increasing (dictionary) order and move to Step 2 once a candidate is found. Experimental results for optimized attack are given in Table 2.

Comparison with Bootle and Tibouchi's Attack. We note that Bootle and Tibouchi also proposed a practical attack [29] against Liu et al.'s encryption scheme. They deployed the technique of embedding lattices to compute the nonce sequence $i_1, \ldots, i_w$ in encryption process $\mathbf{Enc}(\cdot)$, while we start from a different angle and recover a substitutable tuple of private keys $(sk', p', \mathbf{s}', \mathbf{e}')$. We hold the view that the insecurity of Liu et al.'s scheme is not only a result of the small value of n as claimed in [29], but also the overstretched magnitude relation between the modulus q and parameters b, m, and n, which is clarified in Theorem 11.

4.3. Attack against Galbraith's Scheme. In [24], Galbraith proposed a class of LWE-based encryption for constrained devices with more compact parameters; that is, the public matrix $\mathbf{A}$ is binary. We tried to attack Galbraith's scheme exploiting short vectors of $\mathscr{L}_q^{\perp}(\mathbf{A})$ as described before, but it was ineffective even for the parameters totally broken in [27]. That is because the modulus q in Galbraith's scheme is not so

overstretched. However, the binary public matrix and encryption nonce may still be problematic as suggested in [27].

5. Conclusion

In this paper, we target the variant of LWE called Compact-LWE which may be applied to design IoT-oriented lightweight cryptography. We give an explicit analysis of Compact-LWE and point out some weak instances with extreme compactness and overstretched moduli. As an application of our results, we propose a practical attack against the lightweight public key scheme in [25]. Consequently, we claim that the security estimation in [25] is incorrect.

The fragility of the scheme in [25] comes not only from its small parameters but also from the weak hardness of Compact-LWE. It would be interesting to generally figure out a theoretical hardness relation between Compact-LWE and other lattice problems.

Compact-LWE may be still of some interest under refined parameters. We leave to future work the issues of tradeoff between efficiency and security, in particular the practical parameter selections achieving given security levels for IoT devices.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors thank Léo Ducas and Professor Xiaoyun Wang for helpful discussions and comments. This research was supported by the National Key Research and Development Program of China (Project no. 2017YFA0303903), 973 National Program on Key Basic Research Project of China (Project no. 2013CB834205), and National Natural Science Foundation of China (no. 61502269).

References

[1] I. Kotenko, I. Saenko, and A. Kushnerevich, "Parallel big data processing system for security monitoring in internet of things networks," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 8, no. 4, pp. 60–74, 2017.

[2] R. Sanchez-Iborra, J. S. Gómez, J. Santa et al., "Integrating LP-WAN communications within the vehicular ecosystem," *Journal of Internet Services and Information Security*, vol. 7, no. 4, pp. 45–56, 2017.

[3] G. Pau, M. Collotta, S. Tirrito, and R. Caponetto, "An innovative approach for the management of cross-coupling interference in street lighting networks," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 8, no. 2, pp. 44–63, 2017.

[4] K. Zhang, X. Liang, R. Lu, and X. Shen, "Sybil attacks and their defenses in the internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 5, pp. 372–383, 2014.

[5] V. Desnitsky, D. Levshun, A. Chechulin, and I. Kotenko, "Design technique for secure embedded devices: Application for creation of integrated cyber-physical security system," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, vol. 7, no. 2, pp. 60–80, 2016.

[6] S. Aram, R. A. Shirvani, E. Pasero, and M. F. Chouikha, "Implantable medical devices; networking security survey," *Journal of Internet Services and Information Security*, vol. 6, no. 3, pp. 40–60, 2016.

[7] H. Seo, Z. Liu, J. Großschädl, and H. Kim, "Efficient arithmetic on ARM-NEON and its application for high-speed RSA implementation," *Security and Communication Networks*, vol. 9, no. 18, pp. 5401–5411, 2016.

[8] Z. Liu, X. Huang, Z. Hu, M. K. Khan, H. Seo, and L. Zhou, "On emerging family of elliptic curves to secure internet of things: ECC comes of age," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 3, pp. 237–248, 2017.

[9] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing, STOC 2005*, pp. 84–93, Baltimore, MD, USA, 2005.

[10] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pp. 197–206, ACM, Victoria, British Columbia, Canada, 2008.

[11] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," in *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pp. 97–106, USA, October 2011.

[12] S. Gorbunov, V. Vaikuntanathan, and H. Wee, "Attribute-based encryption for circuits," in *Proceedings of the 45th Annual ACM Symposium on Theory of Computing, STOC 2013*, pp. 545–554, June 2013.

[13] M. S. Rahman, A. Basu, and S. Kiyomoto, "Decentralized ciphertext-policy attribute-based encryption: a post-quantum construction," *Journal of Internet Services and Information Security*, vol. 7, no. 3, pp. 1–16, 2017.

[14] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in *Proceedings of the Advances in Cryptology—EUROCRYPT 2010*, vol. 6110 of *Lecture Notes in Comput. Sci.*, pp. 1–23, Springer, French Riviera, France, 2010.

[15] A. Langlois and D. Stehlé, "Worst-case to average-case reductions for module lattices," *Designs, Codes and Cryptography*, vol. 75, no. 3, pp. 565–599, 2015.

[16] L. Ducas, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehlé, "CRYSTALS – Dilithium: Digital signatures from module lattices," in *Cryptology ePrint Archive*, 2017, Report 2017/633, http://eprint.iacr.org/2017/633.

[17] J. Bos, L. Ducas, E. Kiltz et al., "CRYSTALS – Kyber: a cca-secure module-lattice-based kem," in *Cryptology ePrint Archive*, 2017, Report 2017/634, http://eprint.iacr.org/2017/634.

[18] E. Alkim, L. Ducas, T. P, P. Schwabe, and T. Pöppelmann, "Post-quantum key exchange—a new hope," in *Proceedings of the 25th USENIX Security Symposium*, vol. 16, pp. 327–343, 2016, https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/alkim.

[19] Z. Liu, H. Seo, S. S. Roy, J. Großschädl, H. Kim, and I. Verbauwhede, "Efficient ring-LWE encryption on 8-bit AVR processors," *CHES*, vol. 9293, pp. 663–682, 2015.

[20] Z. Liu, R. Azarderakhsh, H. Kim, and H. Seo, "Efficient Software Implementation of Ring-LWE Encryption on IoT Processors," *IEEE Transactions on Computers*, 2017.

[21] T. Güneysu, V. Lyubashevsky, and T. Pöppelmann, "Practical lattice-based cryptography: a signature scheme for embedded systems," in *Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems, CHES 2012*, vol. 7428, pp. 530–547.

[22] D. Micciancio and C. Peikert, "Hardness of SIS and LWE with small parameters," in *Proceedings of the Advances in Cryptology—CRYPTO 2013. Part I*, vol. 8042 of *Lecture Notes in Comput. Sci.*, pp. 21–39, Springer, Santa Barbara, CA, USA, 2013.

[23] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé, "Classical hardness of learning with errors," in *Proceedings of the 45th Annual ACM Symposium on Theory of Computing, STOC 2013*, pp. 575–584, USA, June 2013.

[24] S. D. Galbraith, "Space-efficient variants of cryptosystems based on learning with errors," 2013, https://www.math.auckland.ac.nz/~sgal018/compact-LWE.pdf.

[25] D. Liu, N. Li, J. Kim, and S. Nepal, "Compact-LWE: Enabling practically lightweight public key encryption for leveled IoT device authentication," in *Cryptology ePrint Archive*, 2017, Report 2017/685, http://eprint.iacr.org/2017/685.

[26] S. Bai and S. D. Galbraith, "Lattice decoding attacks on binary LWE," *ACISP*, vol. 8544, pp. 322–337, 2014.

[27] G. Herold and A. May, "LP solutions of vectorial integer subset sums - cryptanalysis of Galbraith's binary matrix LWE," in *Proceedings of the Public-key cryptography—PKC 2017. PART I*, vol. 10174 of *Lecture Notes in Comput. Sci.*, pp. 3–15, Springer, Amsterdam, The Netherlands, 2017.

[28] E. Kirshanova, A. May, and F. Wiemer, "Parallel implementation of BDD enumeration for LWE," *ACNS*, vol. 9696, pp. 580–591, 2016.

[29] J. Bootle and M. Tibouchi, "Cryptanalysis of Compact-LWE," in *Cryptology ePrint Archive*, 2017, Report 2017/742, http://eprint.iacr.org/2017/742.

[30] A. K. Lenstra, J. Lenstra, and L. Lovász, "Factoring polynomials with rational coefficients," *Mathematische Annalen*, vol. 261, no. 4, pp. 515–534, 1982.

[31] P. Q. Nguyen and B. Vallée, *The LLL algorithm: Survey and applications*, Springer, 2010.

[32] L. Babai, "On Lovász' lattice reduction and the nearest lattice point problem," in *STACS*, vol. 182 of *Lecture Notes in Comput. Sci.*, pp. 13–20, Springer, Saarbrücken, Germany, 1985.

[33] M. Albrecht, S. Bai, and L. Ducas, "A subfield lattice attack on overstretched NTRU assumptions: cryptanalysis of some FHE and graded encoding schemes," in *Proceedings of the Advances in Cryptology—CRYPTO 2016. Part I*, vol. 9814 of *Lecture Notes in Comput. Sci.*, pp. 153–178, Springer, Santa Barbara, CA, USA, 2016.

[34] P. Kirchner and P.-A. Fouque, "Revisiting lattice attacks on overstretched NTRU parameters," in *Proceedings of the Advances in Cryptology—EUROCRYPT 2017. Part I*, vol. 10210 of *Lecture Notes in Comput. Sci.*, pp. 3–26, Springer, Paris, France, 2017.

[35] V. Shoup, "NTL: A library for doing number theory," http://www.shoup.net.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 1746809, 14 pages
https://doi.org/10.1155/2018/1746809

Research Article

International Network Performance and Security Testing Based on Distributed Abyss Storage Cluster and Draft of Data Lake Framework

ByungRae Cha,[1] **Sun Park**,[1] **JongWon Kim**,[1] **SungBum Pan**,[2] **and JuHyun Shin**[3]

[1]*School of Electrical Engineering and Computer Science, GIST, Gwangju, Republic of Korea*
[2]*Department of Electronic Engineering, Chosun University, Gwangju, Republic of Korea*
[3]*Department of ICT Convergences, Chosun University, Gwangju, Republic of Korea*

Correspondence should be addressed to JuHyun Shin; jhshinkr@chosun.ac.kr

Received 18 September 2017; Revised 7 December 2017; Accepted 1 January 2018; Published 18 February 2018

Academic Editor: Ilsun You

The megatrends and Industry 4.0 in ICT (Information Communication & Technology) are concentrated in IoT (Internet of Things), BigData, CPS (Cyber Physical System), and AI (Artificial Intelligence). These megatrends do not operate independently, and mass storage technology is essential as large computing technology is needed in the background to support them. In order to evaluate the performance of high-capacity storage based on open source Ceph, we carry out the network performance test of Abyss storage with domestic and overseas sites using KOREN (Korea Advanced Research Network). And storage media and network bonding are tested to evaluate the performance of the storage itself. Additionally, the security test is demonstrated by Cuckoo sandbox and Yara malware detection among Abyss storage cluster and oversea sites. Lastly, we have proposed the draft design of Data Lake framework in order to solve garbage dump problem.

1. Introduction

Most new technologies improve product performance, and these technologies are called persistent technologies. Persistent technologies can be either disconnected or radical because of their nature, but many of them have a gradual character. Sometimes destructive technologies arise, and innovative technologies bring to market a value proposition that is quite different from what was used in the past. The trend of software field specially had been much changed in aspect of software development process, application architecture, deployment and package, and application infrastructure as shown in Figure 1. In this paper, we intend to develop high-capacity, distributed storage, and network acceleration technologies based on open source to gain the opportunity of transition and growth from existing storage technologies of existing leading companies to innovative storage technologies for emerging small and medium enterprises. For this, we are using open source Ceph [1–3]. Ceph is an open source project launched to implement large SDS (Software-Defined Storage) [4] using Intel processor-based general purpose H/W. SDS creates a virtualized network of storage resources by separating the control and management software from the underlying hardware infrastructure. This can be used to create storage networks that may tie together large pools of storage resources that can appear as one virtual entity. Thus, costs and limitations and storage services are possible without a monopoly.

For industrial IoT, the emerging Data Lake concept is proposing to turn things upside down for enterprise; instead of defining a database structure first and then populating it with data that fits into this structure, the Data Lake simply stores any and all kinds of data and then makes this data available when it is needed, in whatever format is needed. Also, we want to be able to keep this data for a longer time, in order to perform long-term pattern analysis. Data Lake repository can be queried on an ad hoc basis, along with a data refinery [5]. The concept of a Data Lake has evolved over time in enterprises, starting with concepts of data warehouse which contained data for long-term retention and

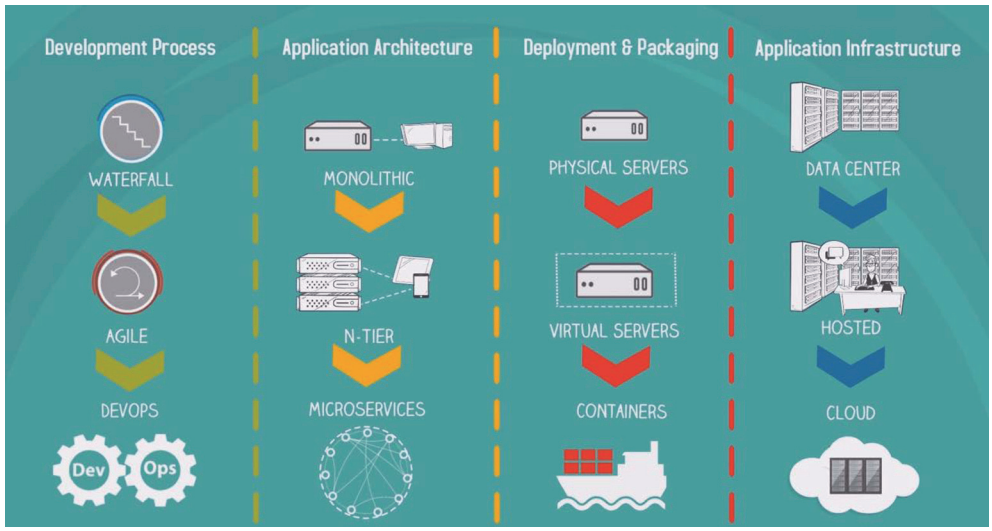


Figure 1: Trend changings in aspect of SW.

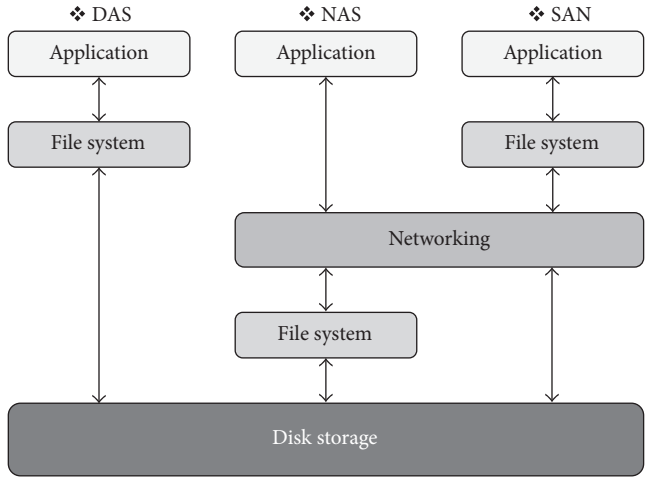


Figure 2: Classification of DAS, NAS, and SAN.

stored differently for reporting and historic needs. Data Lake evolved with these concepts as a central data repository for an enterprise that could capture data as us, produce processed data, and serve the most relevant enterprise information. Data Lake can be defined as a vast repository of a variety of enterprise-wide, raw information that can be acquired, processed, analyzed, and delivered. A Data Lake is expected to be able to derive enterprise-relevant meanings and insights from this information using various analysis and machine learning algorithms [6].

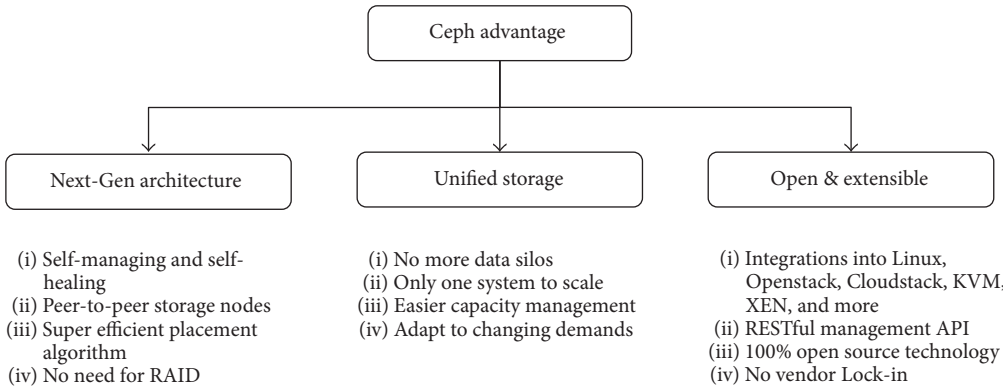
In this paper, using KOREN network, the disk media performance test, network bonding, and network traffic test and security tests of Cuckoo sandbox and Yara malware detection are performed to improve performance and security of mass distributed Abyss storage cluster based on open source Ceph. Lastly, the Data Lake framework using Abyss storage cluster has the potential to become a quite useful foundation for analytical processing. In order to solve the demerit of one-way Data Lake called garbage dump, we have proposed the applying topology and machine learning technology and the draft design of Data Lake framework.

2. Related Work

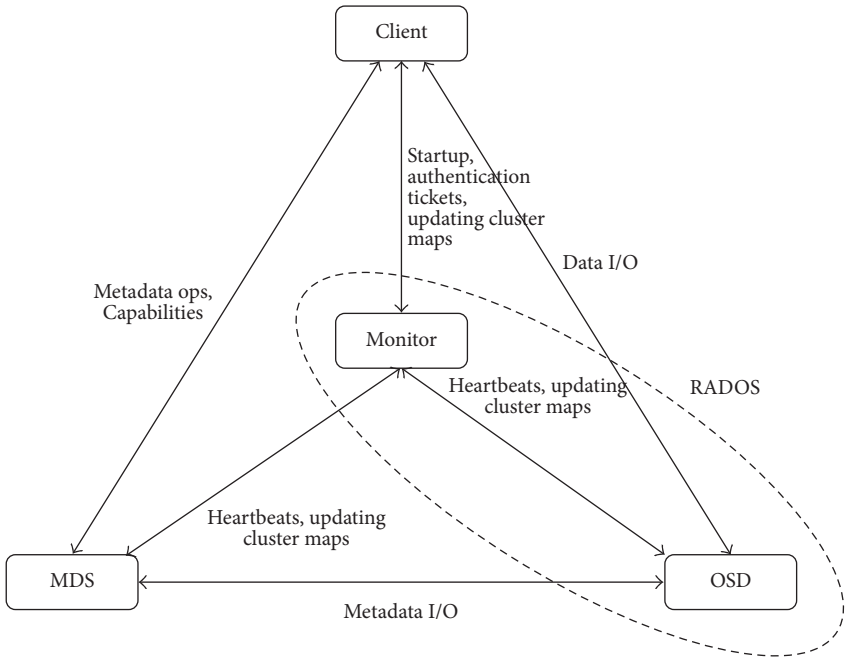
In this section, we briefly described the related terms of open source Ceph [1–3], Data Lake [5–9], and Docker-based security tools of Cuckoo sandbox [10] and Yara malware detection [11, 12].

First, before describing open source Ceph, the techniques of storage are divided into DAS (Direct Attached Storage), NAS (Network Attached Storage), and SAN (Storage Area Network) as shown in Figure 2. Open source Ceph for mass storage is a SAN and Linux distributed file system of a petabyte scale and started with a doctoral research project on Sage Weil's storage systems at UCSC (University of California, Santa Cruz). The advantages of open source Ceph include the next-generation architecture, integrated storage, and openness and scalability, as shown in Figure 3 (advantage of Ceph).

❖ Advantages of Ceph



❖ Component interactions of Ceph



❖ Triple-copy & self-healing by Ceph

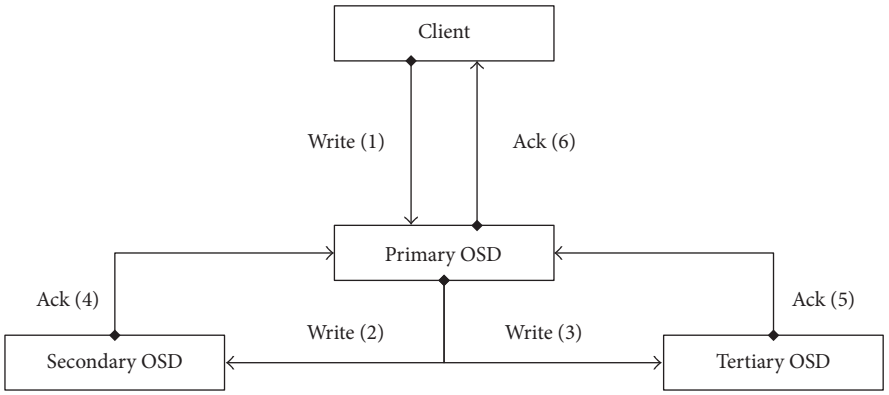


Figure 3: Advantages, architecture and interaction, and triplication of open source Ceph.

Table 1: Key differences between data warehouse and Data Lake.

Items	Data warehouse	Data Lake
Data	Structured, processed	Structured/semistructured/unstructured, raw
Processing	Schema-on-write	Schema-on-read
storage	Expensive for large data volumes	Designed for low-cost storage
Agility	Less agile, fixed configuration	Highly agile, configure and reconfigure as needed
Security	Mature	Maturing
users	Business professionals	Data scientists, etc.

The architecture and interaction with open source Ceph system can be expressed as shown in Figure 3 (component interaction of Ceph), and Figure 3 (triple-copy and self-healing by Ceph) briefly shows the Ceph self-healing and data triplication procedures. Thus, open source Ceph comes with a built-in benchmarking SW tool called the RADOS (Reliable Autonomic Distributed Object Store) bench, which can be used to measure the performance of Ceph clusters at the resource pool level [1–3].

Second, a Data Lake refers to a massively scalable storage repository that holds a vast amount of raw data in its native format until it is needed. While a hierarchical data warehouse stores data in files or folders, a Data Lake uses a flat architecture to store data [5–9].

Lastly, Yara is a tool aimed at helping malware researchers to identify and classify malwares. With Yara you can create descriptions of malware families (or whatever you want to describe) based on textual or binary patterns. Each description, a.k.a. rule, consists of a set of strings and a Boolean expression which determine its logic. And Cuckoo sandbox is a malware analysis system. In other words, admin can throw any suspicious file at Cuckoo sandbox and in a matter of seconds Cuckoo will provide you back with some detailed results outlining what such file did when executed inside an isolated environment.

3. Development of Abyss Storage Cluster Prototype and Draft Design Data Lake Framework

In order to design and expand Data Lake framework, first, we designed and developed the prototype of Abyss storage cluster H/W using open source Ceph based on Ubuntu Server.

3.1. Development of Abyss Storage Cluster Prototype. Abyss storage cluster prototype has been designed and developed as shown in Figures 4 and 5 [13]. The logical components of the mass volume Abyss storage cluster for SMB (Small and Medium Business) and Docker-based security tools are shown in Figure 4. And there is H/W prototype of the Abyss storage, 3D rendering image, and the 3D printing material of the product case of Abyss storage as shown in Figure 5, actually.

3.2. Docker-Based Security Tools (Cuckoo and Yara) on Abyss Storage Cluster. Abyss storage cluster supports the Docker of virtualization technology and Cuckoo and Yara of security tools as shown in ① and ② of Figure 4. Docker is lightweight virtualization technology, an open platform for developers, and sysadmins to build, ship, and run distributed applications, whether on laptops, data center VMs (Virtual Machines), or the cloud [14, 15]. The Linux kernel's support for namespaces mostly isolates an application's view of the operating environment, including process trees, network, user IDs, and mounted file systems, while the cgroups of kernel provide resource limiting, including the CPU, memory, block I/O, and network. To use virtualization technology, Docker-based security tools based on Abyss storage cluster are Cuckoo sandbox and Yara malware detection. Cuckoo sandbox is an application that provides a virtual sandbox for the automatic analysis of malware specimens. Another powerful feature of Cuckoo sandbox is the ability to utilize the Yara framework. Yara provides a rule-based approach to create descriptions of malware families based on textual or binary patterns. A description is essentially a Yara rule name, where these rules consist of sets of strings and a Boolean expression. Docker-based security tools in Abyss storage cluster have been operated and verified among domestic sites and oversea sites.

3.3. Draft Design of Data Lake Framework. A Data Lake is a scalable storage repository that is an advanced version of the data warehouse and data silo in the BigData era. There are some differences between Data Lake and data warehouse in aspect of data, processing, storage, agility, security, and users as shown in Table 1 [16]. It is important to recognize that while both the data warehouse and Data Lake are storage repositories, the Data Lake is not data warehouse 2.0 nor is it a replacement for the data warehouse.

And a data silo is a repository of fixed data that remains under the control of one department and is isolated from the rest of the organization; much like grain in a farm silo is closed off from outside elements. Data silos can have technical or cultural roots. Data silos tend to arise naturally in large organizations because each organizational unit has different goals, priorities, and responsibilities. Data silos can also occur when departments compete with each other instead of working with each other towards common business goals. Information silos are generally viewed as a hindrance to effective business operations and organizations are increasingly trying to break down silos that are a barrier to collaboration, accessibility, and efficiency [17].

There are many reasons for users to be frustrated with the information pooling in their Data Lakes. The core issue was that the larger the information lake grew, the more difficult analyzing the data became. A Data Lake of any significant

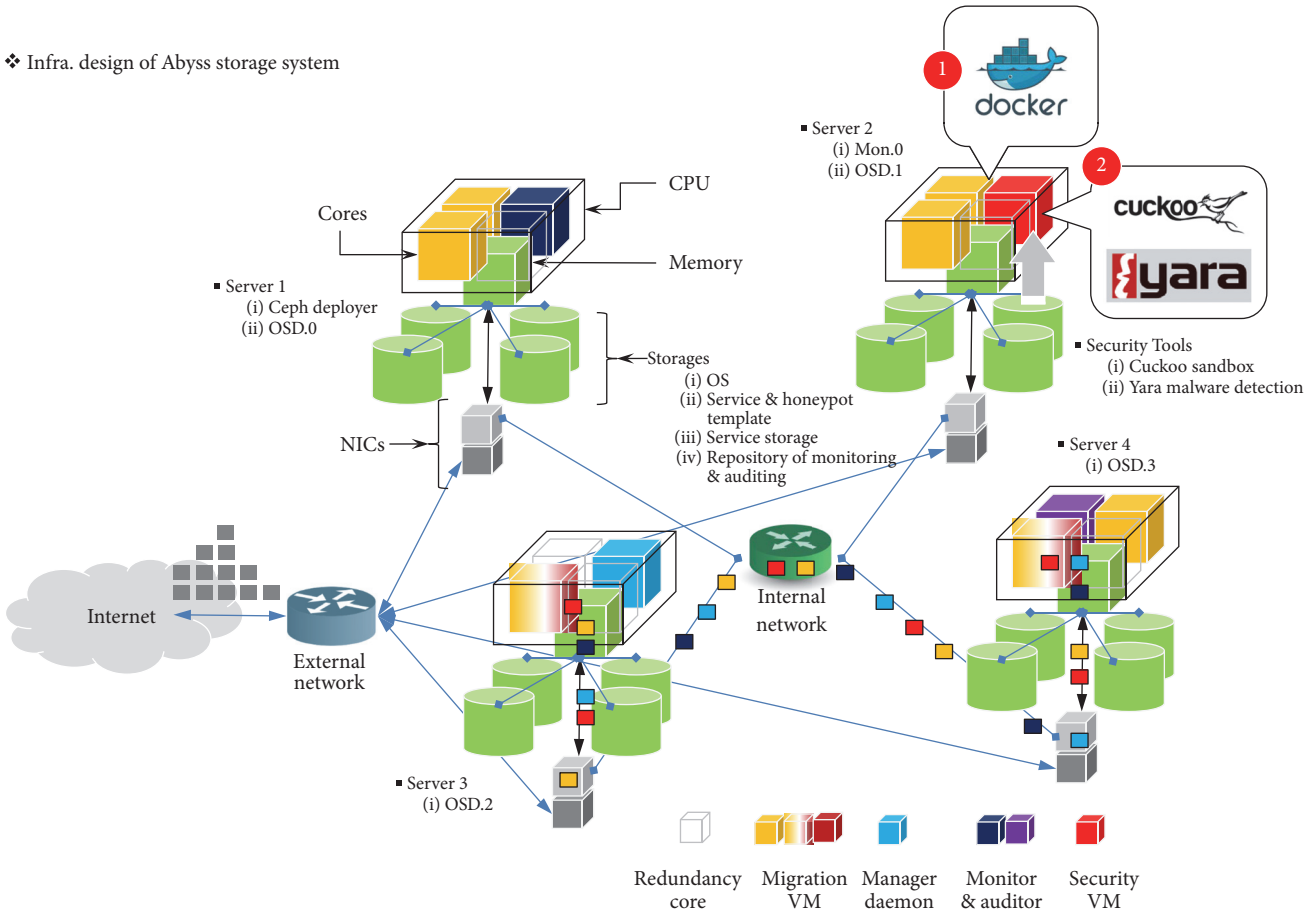


Figure 4: Concept diagram of mass distributed Abyss storage cluster and Docker-based security tools.

Figure 5: H/W prototype and 3D printing product of Abyss storage cluster.

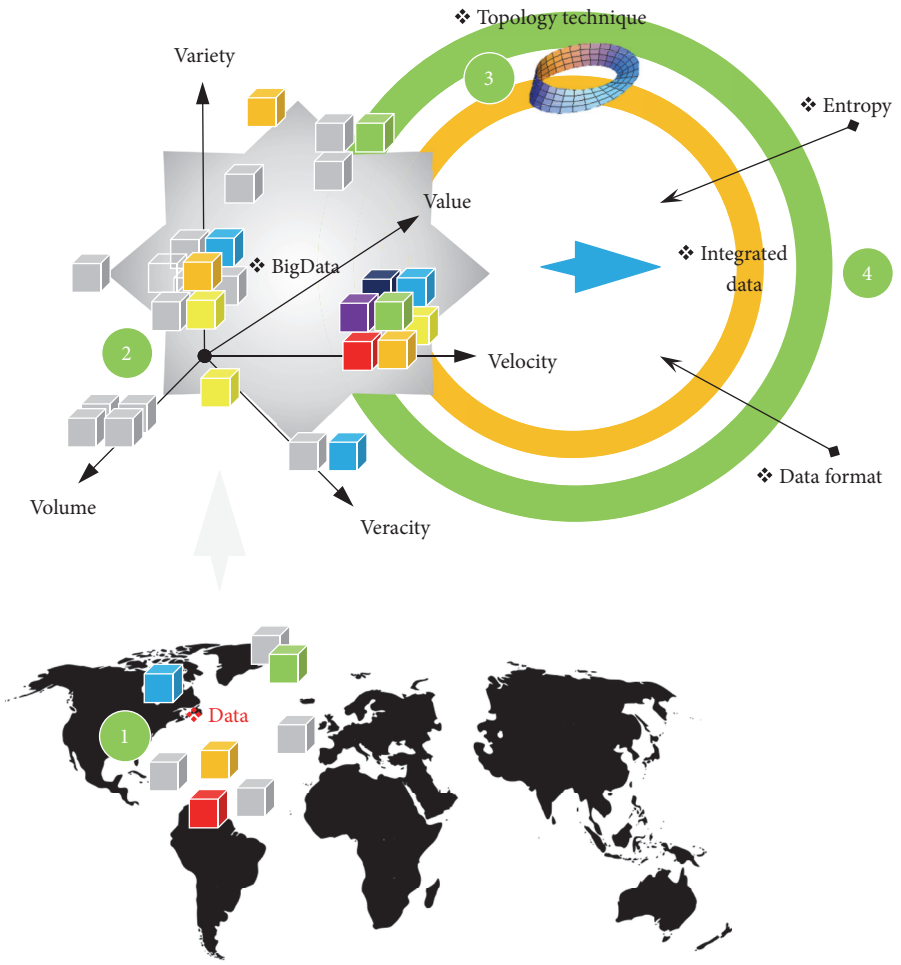


Figure 6: Concept of applying mathematics topology for BigData.

size was often dubbed a "one-way lake," since data is eternally pouring in, but data and any analysis are never taken out, or even accessed once the data is placed inside the Data Lake.

There is one reason why the Data Lake turns into a "one-way" Data Lake. But this issue traces its roots to how data was placed into the Data Lake in the first place: the intent was never to organize the data for future usage. Instead the Data Lake became a place just to "dump" data. So much effort was spent on gathering data from every possible source that few engineers and companies gave much thought to organizing the data for future usage. The Data Lake has the potential to become a quite useful foundation for analytical processing. In order to solve the demerit of one-way Data Lake called garbage dump, we have proposed the applying mathematics topology and machine learning technology and the draft design of Data Lake framework in Figures 6 and 7. We specially proposed applying the mathematics topology and machine learning technology in Data Lake framework as shown in ① and ② of Figure 7.

Mathematics topology [18–21] is concerned with the properties of space that are preserved under continuous deformations. This can be studied by considering a collection of subsets, called open sets, which satisfy certain properties, turning the given set into what is known as a topological space. Important topological properties include connectedness and compactness. The initial motivation is to study the shape of data. TDA (Topology Data Analysis) has combined algebraic topology and other tools from pure mathematics to allow mathematically rigorous study of "shape." The main tool is persistent homology, an adaptation of homology to point cloud data. Persistent homology has been applied to many types of data across many fields. Moreover, its mathematical foundation is also of theoretical importance. The unique features of TDA make it a promising bridge between topology and geometry.

Furthermore, ML (machine learning) is the subfield of computer science that evolved from the study of pattern recognition and computational learning theory in AI and ML explores the study and construction of algorithms that can learn from and make predictions on data; such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs. ML is employed in a range of computing tasks where designing and programming explicit algorithms with good performance are difficult or infeasible. Thus, iML is closely related to computational statistics,

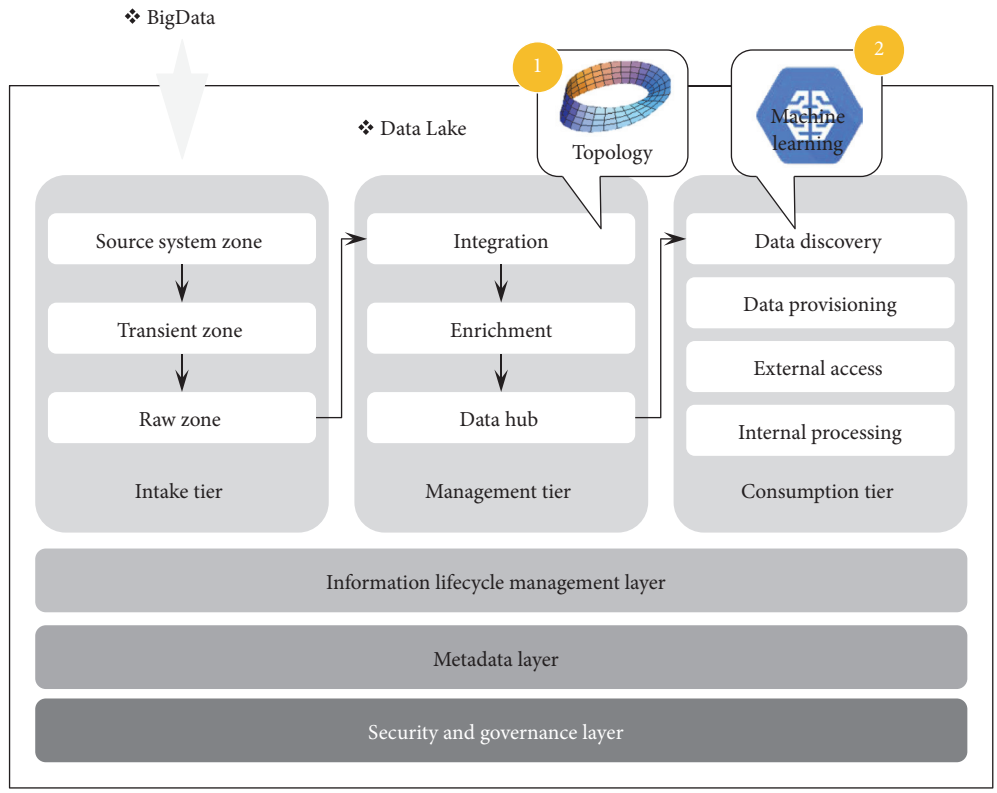


Figure 7: Draft design of Data Lake framework.

which also focuses on prediction-making through the use of computers. It has strong ties to mathematical optimization, which delivers methods, theory, and application domains to the field. Machine learning is sometimes conflated with data mining, where the latter subfield focuses more on exploratory data analysis and is known as unsupervised learning. ML can also be unsupervised and be used to learn and establish baseline behavioral profiles for various entities and then used to find meaningful anomalies [22]. And recently, variety of open source based ML tools (TensorFlow [23], Caffe [24], Torch7 [25], Cuda-convert [26], Chainer [27], and MXNet [28]) have been provided due to the influence of Deep Learning [29] neural network.

4. Performance and Security Testing of Abyss Storage Cluster Prototype

In order to improve performance of Abyss storage, we have tested performance of storage media by disk types, network bonding for acceleration of internal network of Abyss storage cluster, and international network performance test using KOREN [30].

4.1. Disk Media Test of Abyss Storage. The disk media tests of Abyss storage servers were performed by disk media types (HDD (Hard Disk Drive), SSHD (Solid State Hybrid Drive), and SSD (Solid State Drive)). Using RADOS Bench S/W, we performed the read and write operation of disk media types for 10 seconds and recorded the average of IOPS (Input/Output Operations per Second) using RADOS Bench SW as shown in Figure 8 [13].

4.2. Network Acceleration by Bonding. For network acceleration, the internal network of Abyss storage cluster is bonded with two 1 GB switches into VLAN (Virtual Local Area Network) as shown in Figure 9. We have performed the tests of write, sequence read, and random read operation between general network and network bonding, and Figure 10 shows the average IOPS comparison of test results. Test results of the system with network bonding for network acceleration were improved by at least 170% more than general network system [13].

5. Network Performance and Docker-Based Security Test of Abyss Storage Cluster Prototype

5.1. Environments of Network Performance and Security Test. We performed network performance test through uploading and downloading of multimedia data among GIST in Korea, Myren in Malaysia, Thailand, and Philippine sites. Figure 11 presents the test-bed environment using KOREN for real network performance and security test of Cuckoo sandbox and Yara malware detection among domestic sites and oversea sites.

5.2. Domestic Network Performance Test. The domestic network performance test has been performed between Abyss

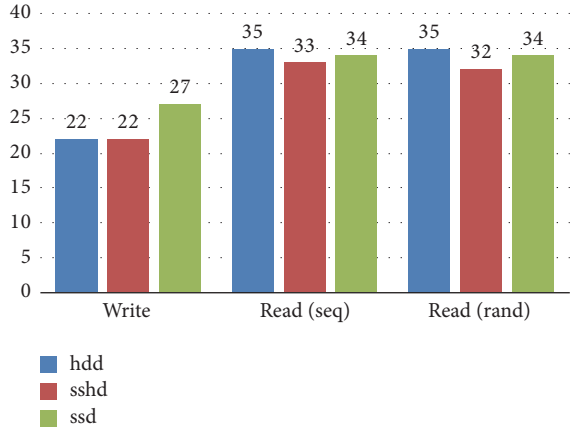


Figure 8: IOPS comparison of Abyss storage by disk media types.

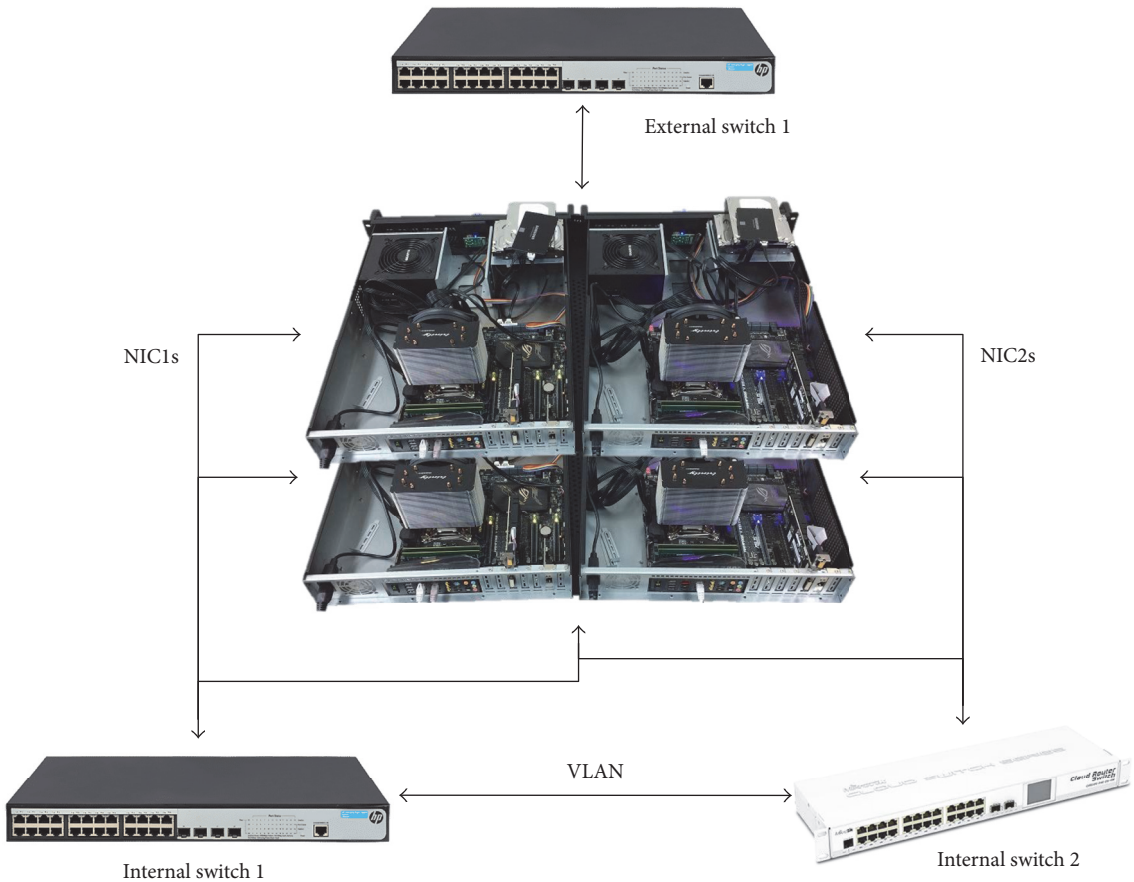


Figure 9: Network bonding of Abyss storage cluster.

storage cluster in GIST and another GIST site. And we measured the external network traffic of Abyss storage and the internal network traffic inside Abyss storage cluster using SpeedoMeter [31], a network real-time monitoring tool. Figure 12 shows test results of file uploading speed, and Figure 13 presents external and internal traffics on Abyss storage cluster in GIST during file uploading. Conversely, Figure 14 shows test results of file downloading speed, and Figure 15 presents external and internal traffics on Abyss storage cluster during file downloading. The comparison of uploading and downloading network traffic between domestic sites is depicted in Figure 16. Although it will be recognized later, the domestic network performance is higher than network performances among oversea sites.

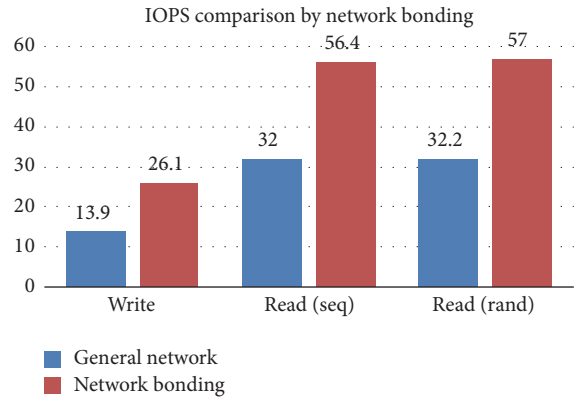


Figure 10: IOPS comparison between general network and bonding network.

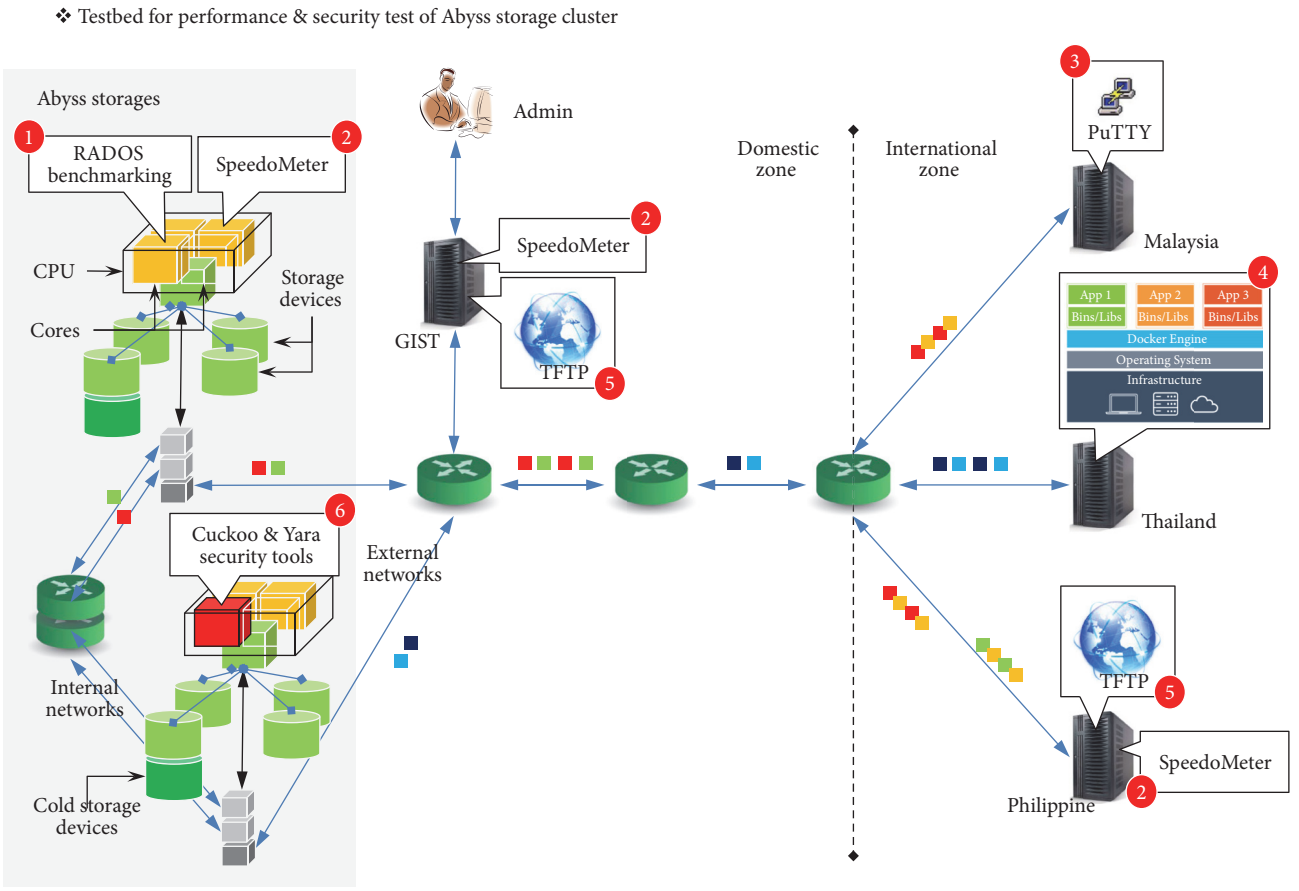


Figure 11: Testbed of Abyss storage cluster prototype for network performance test and security tools test of Cuckoo and Yara using KOREN network.

5.3. International Network Performance and Docker-Based Security Test. In this subsection, using KOREN, the international network performance test has been performed among Abyss storage cluster in GIST, Myren in Malaysia, Thailand, and Philippine sites as shown in Figure 11. Figure 17 shows the speeds of uploading and downloading for each file capacity between Myren in Malaysia and the developed Abyss storage cluster in GIST. Comparing the figures specially shows that the variance among download speeds is much higher than the upload speed. Additionally, the video test between Abyss storage cluster in GIST and Myren has been performed as shown in Figure 18. Figures 19 and 20 present speeds of uploading and downloading among GIST, Thailand, and Philippine sites. In international network performance test, the peculiar cases are more unsafe states of network traffic and high variances of network speed on all sites in

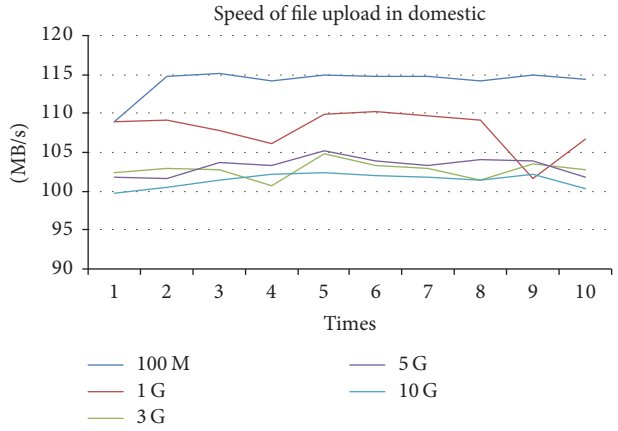
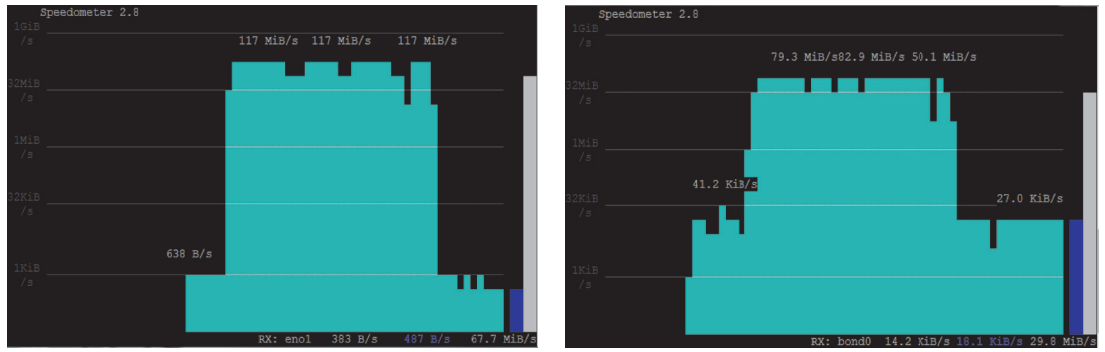


Figure 12: Test result of file uploading speed between Abyss storage cluster and another GIST site.



(a) External traffic of Abyss storage

(b) Internal traffic of Abyss storage cluster

Figure 13: External and internal traffics of Abyss storage cluster during files uploading.

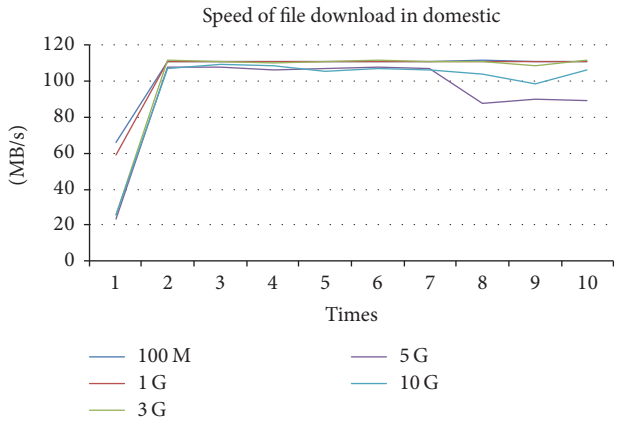
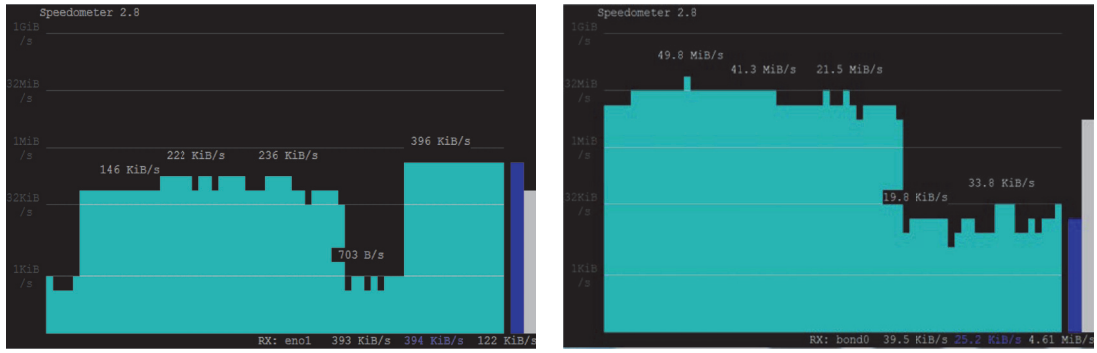


Figure 14: Test result of file downloading speed between Abyss storage cluster and another GIST site.

South-East Asia area than domestic site test. Moreover, the comparisons of upload and download network performance among GIST, Malaysia, Thailand, and Philippine are depicted in Figure 21. Lastly, we have performed the Docker-based security test using Yara malware detection tool among Abyss storage cluster in GIST and oversea sites. This has examined and verified the possibilities of virtualization-based security functions.



(a) External traffic of Abyss storage (b) Internal traffic of Abyss storage cluster

Figure 15: External and internal traffics of Abyss storage cluster during files downloading.

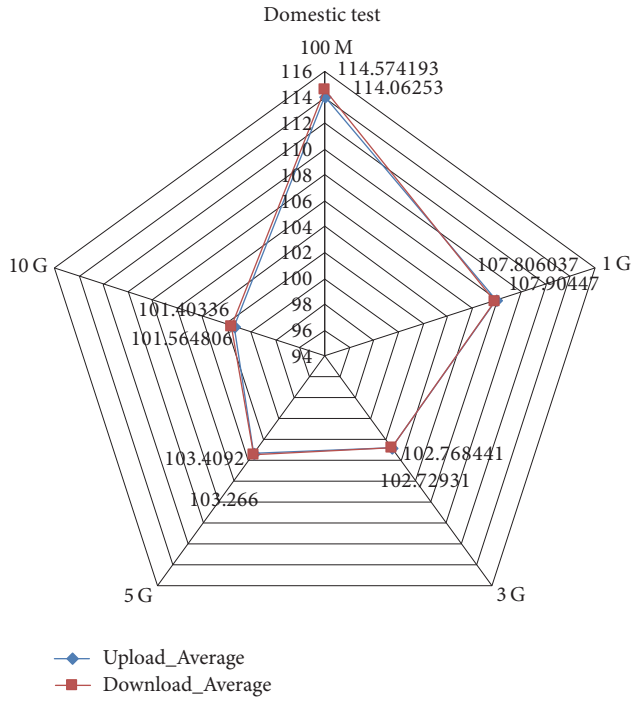


Figure 16: Comparison of uploading and downloading network traffics in domestic sites.

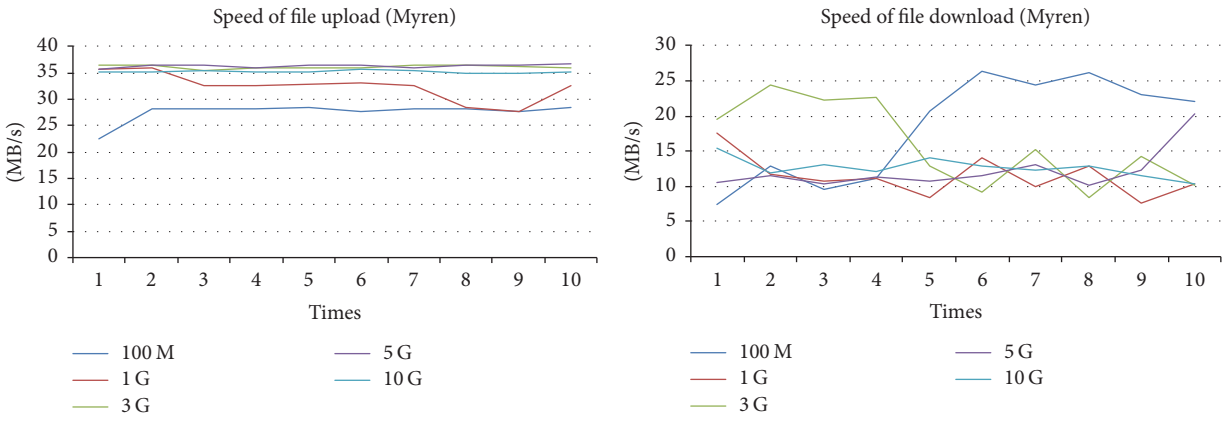


Figure 17: Test result of uploading and downloading speed between Abyss storage cluster in GIST and Myren in Malaysia.

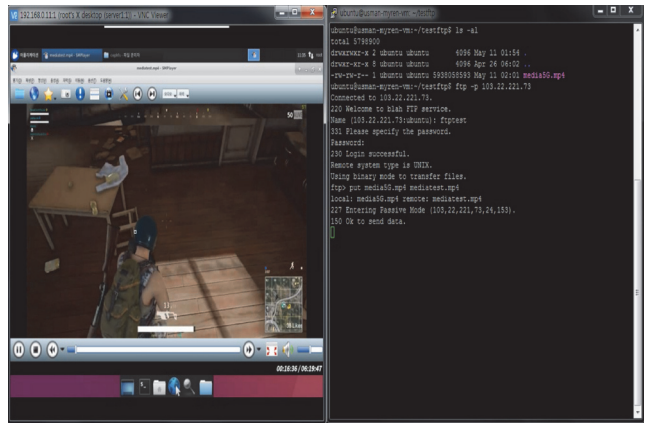


Figure 18: Video test between Abyss storage cluster in GIST and Myren in Malaysia.

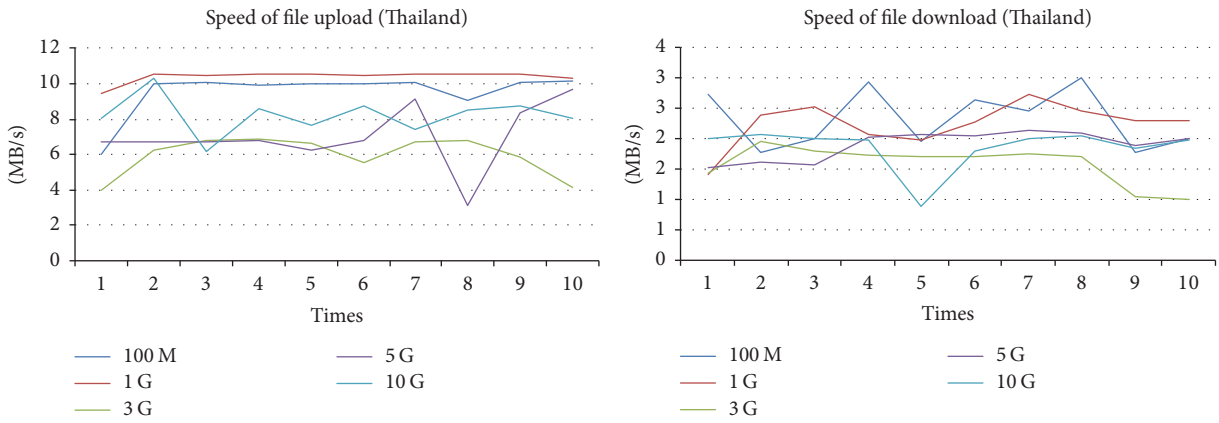


Figure 19: Test result of uploading and downloading speed between Abyss storage cluster in GIST and Thailand.

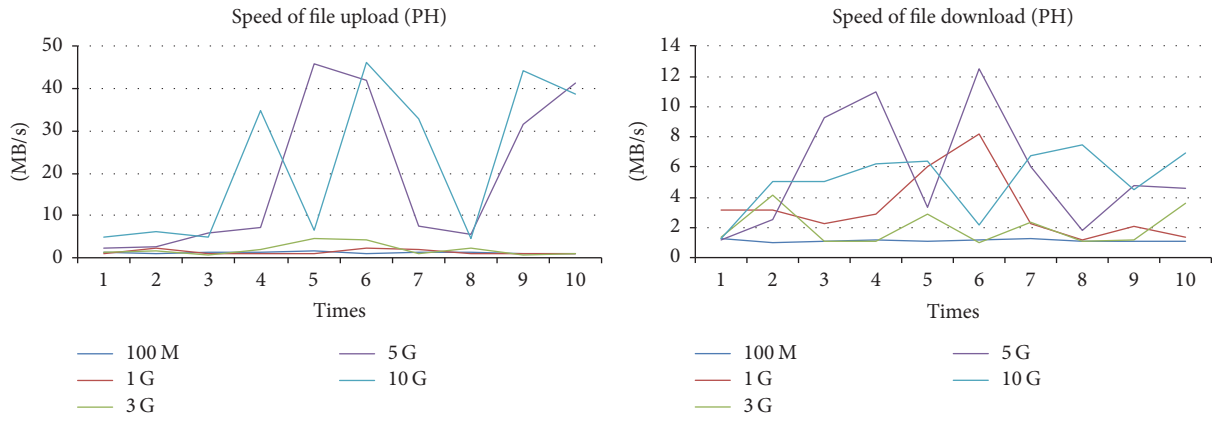


Figure 20: Test results of uploading and downloading speed between Abyss storage cluster in GIST and Philippine.

6. Conclusion

In this paper, the performance tests of the developed mass volume distributed Abyss storage cluster and real-world network performance tests using KOREN have been carried out. Based on this, we intend to explore ways to improve performance of Abyss storage cluster. Detailed tests to improve performance include performance testing of read and write operations for each disk media types (HHD, SSHD, SSD) of Abyss storage servers, internal network testing inside Abyss storage cluster by network bonding, and testing of network performance among domestic and oversea sites. Additionally, we have performed the Docker-based security test using Cuckoo sandbox and Yara malware detection tools among

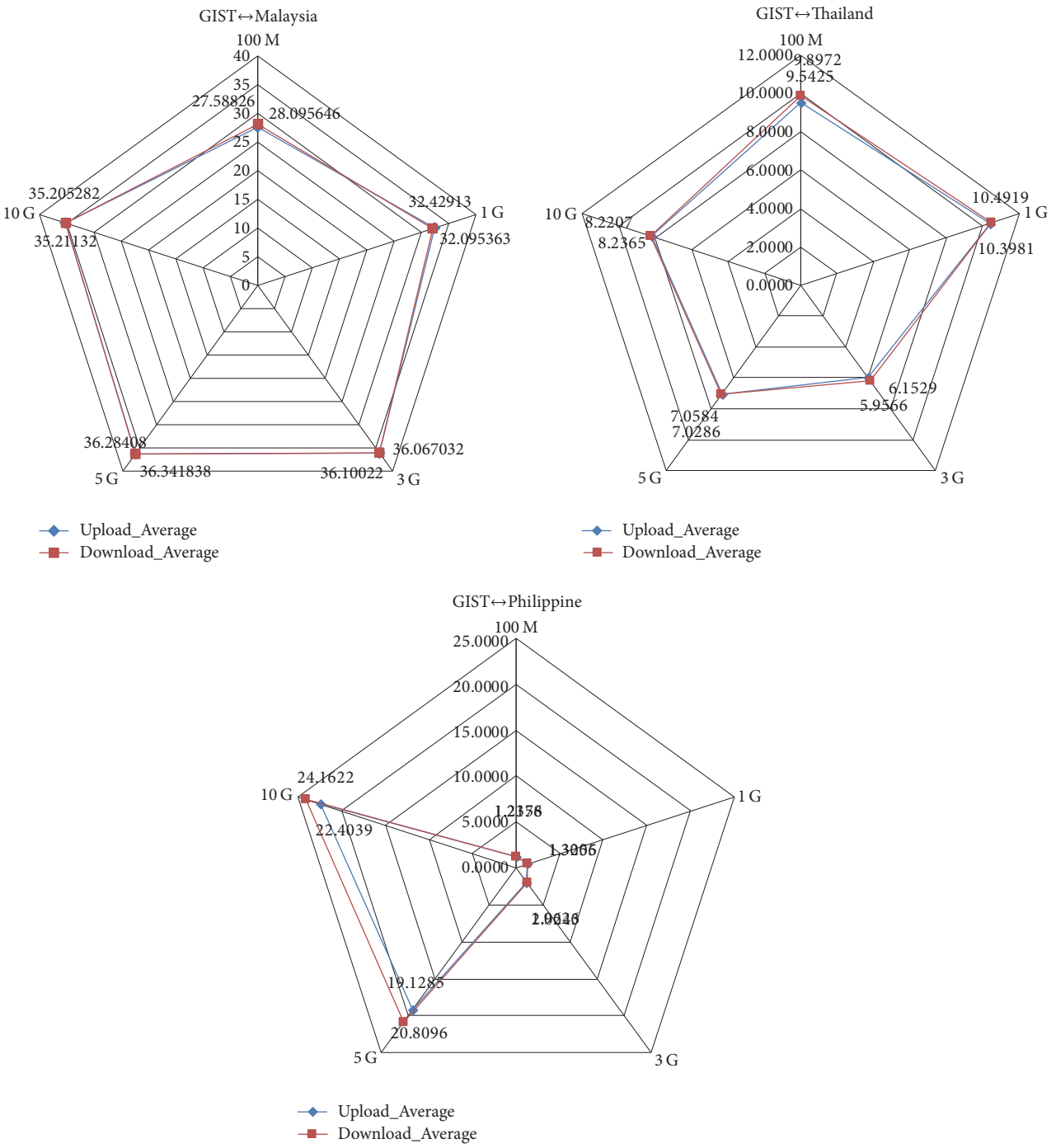


Figure 21: Comparisons of uploading and downloading network traffics among GIST, Malaysia, Thailand, and Philippine.

Abyss storage cluster in GIST and oversea sites. Lastly, we have proposed the draft design of Data Lake framework with mathematics topology and machine learning in order to solve garbage dump problem. In future research of this study, we will focus on MPTCP (MultiPath TCP) [32] for efficient operation of network and Searchable Encryption [33] for data retrieval and security enhancement.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Human Resource Training Program for Regional Innovation and Creativity through

the Ministry of Education and National Research Foundation of Korea (2015H1C1A1035823). And this research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (2017R1E1A1A03070059).

References

[1] Ceph, http://ceph.com/.

[2] Karan Singh, Learning Ceph,.

[3] K. Singh, *Ceph Cookbook*, 2016.

[4] Software-Defined Storage, https://www.sdxcentral.com/cloud/definitions/what-is-software-defined-storage/.

[5] D. Slama, F. Puhlmann, J. Morrish, and R. M. Bhatnagar, *Enterprise IoT - Strategies & Best Practices for Connected Products Services*, O'Reilly, 2016.

[6] T. John and P. Misra, *Data Lake for Enterprises Leveraging Lambda Architecture for Building Enterpeise Data Lake*, Packt Publishing, 2017.

[7] N. Miloslavskaya and A. Tolstoy, "Big Data, Fast Data and Data Lake Concepts," in *Proceedings of the 7th Annual International Conference on Biologically Inspired Cognitive Architectures, BICA 2016*, pp. 300–305, USA, July 2016.

[8] P. Pasupuleti, PACKT Publishing, Beulah Salome Purra, Data Lake Development with Big Data, PACKT Publishing, 2015.

[9] B. Inmon, Data Lake Architecture - Designing the Data Lake and Avoiding the Garbage Dump, Technics Publications, 2016.

[10] Cuckoo Sandbox, https://www.cuckoosandbox.org/.

[11] Yara, https://virustotal.github.io/yara/.

[12] D. Ricardo, *Intelligence-Driven Incident Response with YARA*, SANS.

[13] B. Cha, Y. Cha, S. Park, and J. Kim, "Performance testing of mass distributed abyss storage prototype for SMB," *Advances in Intelligent Systems and Computing*, vol. 611, pp. 762–767, 2018.

[14] Docker, https://www.docker.com/.

[15] Docker, https://en.wikipedia.org/wiki/Docker_(software).

[16] Tamara Dull, *Data Lakes vs Data Warehouse: Key Differences*, 2015, http://www.kdnuggets.com/2015/09/data-lake-vs-data-warehouse-key-differences.html.

[17] Data Silo, http://searchcloudapplications.techtarget.com/definition/data-silo.

[18] Topology, https://en.wikipedia.org/wiki/Topology.

[19] G. Carlsson, "Topology and data," *Bulletin of the American Mathematical Society*, vol. 46, no. 2, pp. 255–308, 2009.

[20] Gunnar Calsoon, Why Topological Data Analysis Works, https://www.ayasdi.com/blog/bigdata/why-topological-data-analysis-works/.

[21] Topological Data Analysis (TDA), https://en.wikipedia.org/wiki/Topological_data_analysis.

[22] Machine Learning, https://en.wikipedia.org/wiki/Machine_learning.

[23] TensorFlow, https://www.tensorflow.org/.

[24] Caffe, http://caffe.berkeleyvision.org/.

[25] Torch7, http://torch.ch/.

[26] Cuda-convert, http://mdtux89.github.io//12/11/torch-tutorial.html.

[27] Chainer, https://chainer.org/.

[28] MXNet, https://mxnet.incubator.apache.org/.

[29] Deep Learning, http://deeplearning.net/.

[30] KOREN, http://www.koren.kr/koren/eng/index.html.

[31] SpeedoMeter, https://hub.docker.com/r/opennsm/speedometer/, ~/ dockerfile/.

[32] Z. Jiaxin, K. Fenfen, Y. Zuo, L. Qinghua, H. Minghe, and C. Yuanlong, "Multi-attribute Aware Path Selection Approach for Efficient MPTCP-based Data Delivery," *Journal of Internet Services and Information Security*, vol. 7, no. 1, 2017.

[33] J. Xiuxiu, G. Xinrui, Y. Jia, K. Fanyu, C. Xiangguo, and H. Rong, "An efficient symmetric searchable encryption scheme for cloud storage," *Journal of Internet Services and Information Security*, vol. 7, no. 2, May 2017.