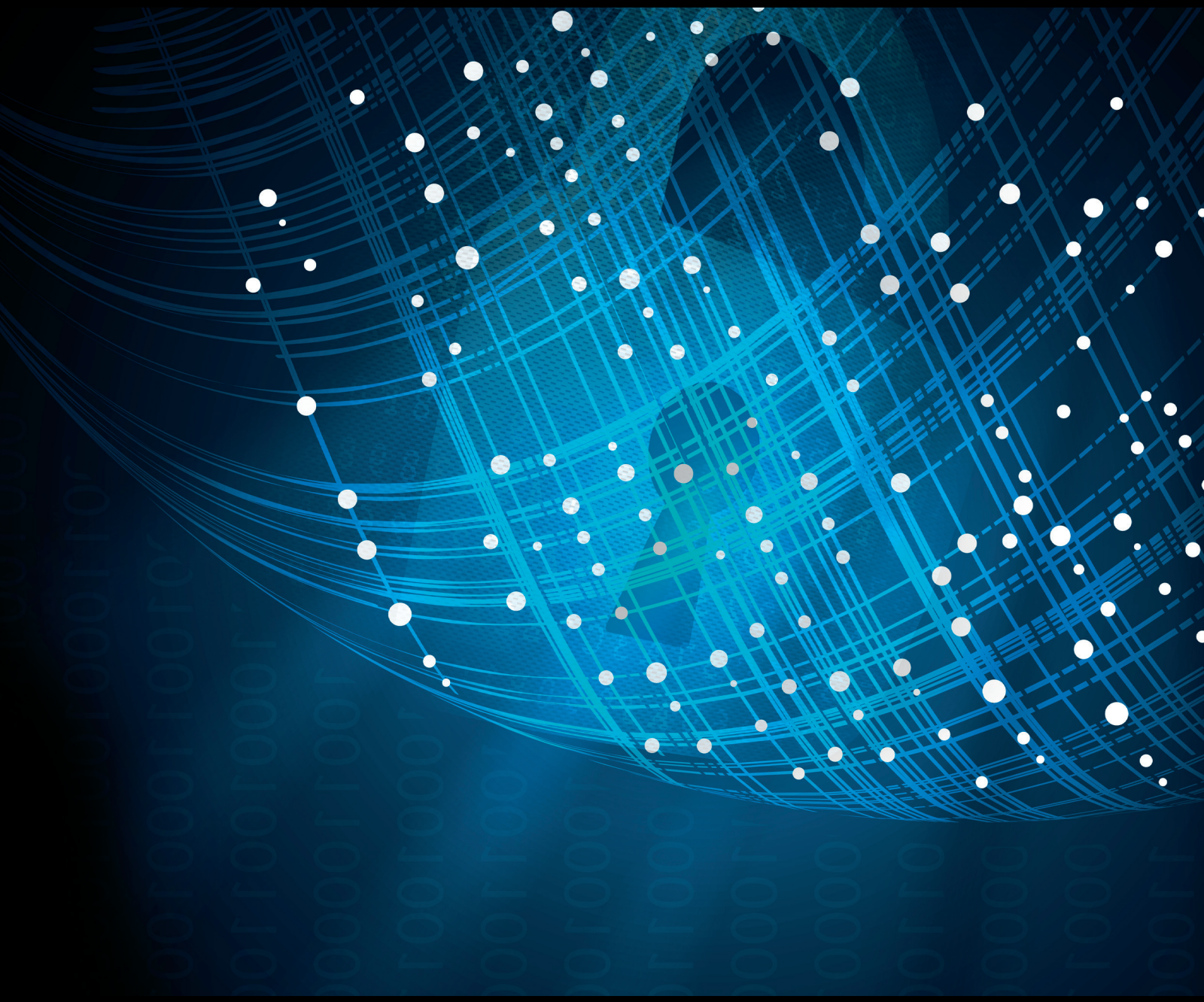
Security and Communication Networks

Intrusion Detection and Prevention in Cloud, Fog, and Internet of Things

Lead Guest Editor: Xuyun Zhang
Guest Editors: Yuan Yuan, Zhili Zhou, Shancang Li, Lianyong Qi, and Deepak Puthal



Intrusion Detection and Prevention in Cloud, Fog, and Internet of Things

Security and Communication Networks

Intrusion Detection and Prevention in Cloud, Fog, and Internet of Things

Lead Guest Editor: Xuyun Zhang
Guest Editors: Yuan Yuan, Zhili Zhou, Shancang Li, Lianyong Qi, and Deepak Puthal

Editorial Board

Contents

Hindawi
Security and Communication Networks
Volume 2019, Article ID 4529757, 4 pages
https://doi.org/10.1155/2019/4529757

Editorial

Intrusion Detection and Prevention in Cloud, Fog, and Internet of Things

Xuyun Zhang,[1] Yuan Yuan,[2] Zhili Zhou,[3] Shancang Li,[4] Lianyong Qi,[5] and Deepak Puthal[6]

[1]*Department of Electrical, Computer and Software Engineering, University of Auckland, Auckland 1023, New Zealand*
[2]*Department of Computer Science and Engineering, Michigan State University, Michigan, MI 48824, USA*
[3]*Nanjing University of Information Science and Technology, Nanjing, 210044, China*
[4]*FET-Computer Science and Creative Technologies, University of the West of England, Bristol BS16 1QY, UK*
[5]*School of Information Science and Engineering, Chinese Academy of Education Big Data, Qufu Normal University, Qufu 276826, China*
[6]*Faculty of Engineering and IT, University of Technology Sydney, Ultimo, NSW 2007, Australia*

Correspondence should be addressed to Xuyun Zhang; xuyun.zhang@auckland.ac.nz

Received 9 April 2019; Accepted 9 April 2019; Published 23 May 2019

We are pleased to announce the publication of the special issue focusing on intrusion detection and prevention in cloud, fog, and Internet of Things (IoT). Internet of Things (IoT), cloud, and fog computing paradigms are as a whole provision a powerful large-scale computing infrastructure for many data and computation intensive applications. Specifically, the IoT technologies and deployment can widely perceive our physical world at a fine granularity and generate sensing data for further insight extraction. The fog computing facilities can provide computing power near the IoT devices where data are generated, aiming to achieve fast data processing for time critical applications or save the amount of data transmitted into cloud for storage or further processing. The cloud computing platforms can offer big data storage and large-scale processing services for cheap long-term storage or data intensive analytics with more advanced data mining models. Hence, it can be seen that the IoT/fog/cloud computing infrastructures can support the whole lifecycle of large-scale applications where big data collection, transmission, storage, processing, and mining can be seamlessly integrated. However, these state-of-the-art computing infrastructures still suffer from severe security and privacy threats because of their built-in properties such as the ubiquitous-access and multitenancy features of cloud computing, or the limited computing capability of IoT devices. The expanded attack surface and the lack of effective security and privacy protection measures are still one of the barriers of widely deploying applications on the IoT/fog/cloud infrastructure.

Intrusion detection and prevention systems that monitor the devices, networks, and systems for malicious activities and policy violations are one of the key countermeasures against cybersecurity attacks. With a wide spectrum, the detection and prevention systems vary from antivirus software to hierarchical systems monitoring the traffic of an entire backbone networks. In general, intrusion detection systems can be categorized into two groups, that is, signature-based detection (malicious patterns are already known) and anomaly-based detection (no patterns are given). Traditional methods and systems might fail to be directly applicable to the state-of-the-art computing paradigms and infrastructure as mentioned above. Novel intrusion detection and prevention algorithms and systems are in demand to cater for the new computing infrastructure and newly emerging cybersecurity attacks and threats, taking into account the factors such as algorithmic scalability, computing environment heterogeneity, data diversity, and complexity. Extensive research is required to conduct more scalable and effective intrusion

detection and prevention in IoT/fog/cloud. Many relevant theoretical and technical issues have not been answered well yet. As such, it is high time to investigate the related issues in intrusion detection and prevision in IoT, fog, and cloud computing by examining intrusion detection and prevision algorithms, methods, architecture, systems, platforms, and applications in detail. This special issue gained substantial interests of researchers from all over the world and our editorial team consisting of six researchers in this field have rigorously selected 20 articles out of 60 submissions for publication. The research topics include intrusion detection system, intrusion prevention systems, DDoS attack detection, network/IoT anomaly detection, anomaly detection in cloud, malware detection, privacy-preservation technologies, and other closely related works on data deduplication, cloudlet placement, and fault analysis.

In the paper entitled "Fingerprinting Network Entities Based on Traffic Analysis in High-speed Network Environment", *X. Gu et al.* studied the entity identification problem in high-speed network environment to detection potential intruders and proposed to use the PFQ kernel module and Storm to capture high-speed packet and analyse online traffic, respectively. Based on this, they further proposed a novel device fingerprinting technology based on the runtime environment analysis that employs a logistic regression model and the sliding window mechanism to implement online identification.

In the paper entitled "Test Sequence Reduction of Wireless Protocol Conformance Testing to Internet of Things", *W. Lin et al.* investigated the wireless protocol conformance testing problems which just judge whether a wireless protocol has been performed as expected and proposed an improved method based on an overlapping technique that makes use of invertibility and multiple unique input/output sequences. Specifically, the method consists of two steps: the maximum-length invertibility-dependent overlapping sequences (IDOSs) are constructed in the first step, and a minimum-length rural postman tour covering the just constructed set of maximum-length IDOSs is generated. Finally, a test sequence is extracted from the tour.

In the paper entitled "Flow Correlation Degree Optimization Driven Random Forest for Detecting DDoS Attacks in Cloud Computing", *J. Cheng et al.* investigated the Distributed Denial-of-Service (DDoS) attacks in cloud computing and proposed a DDoS attack detection method with the enhanced random forest (RF) technique optimized by a genetic algorithm based on the flow correlation degree (FCD) features. Specifically, the features of attack flow and normal flows are described by the two-tuple FCD feature consisting of package-statistical degree (PSD) and semidirectivity interaction abnormality (SDIA). A genetic algorithm based on the FCD feature sequences is used to optimize two key parameters of the decision tree in the RF, and the trained RF model with the optimized parameters is employed to generate the classifier for DDoS attack detection.

In the paper entitled "A Cooperative Denoising Algorithm with Interactive Dynamic Adjustment Function for Security of stacker in Industrial Internet of Things", *D. Huang et al.* studied the problem of security monitoring of stacker in Industry IoT (IIoT) and proposed a cooperative denoising algorithm with interactive dynamic adjustment function. Specifically, the denoising framework named as IDVSLMS-EEMD was constructed based on the advantages of LMS, VSLMS, and improved VSLMS-EEMD. Real-world data applied in Power Grid of China was used to verify and simulate the effectiveness of the proposed algorithms.

In the paper entitled "A Constraint-aware Optimization Method for Concurrency Bug Diagnosis Service in a Distributed Cloud Environment", *L. Bo and S. Jiang* investigated the performance problems in concurrency bug diagnosis services which analyse concurrent software and detect concurrency bugs and proposed a static constraint-aware method to simplify concurrent program buggy traces. Specifically, the maximal sound dependence relations of original buggy traces are calculated based on the constraint models. The simplified traces can be obtained after checking the dependent constraints iteratively and forwarding current events to extend thread execution intervals.

In the paper entitled "Applying Catastrophe Theory for Network Anomaly Detection in Cloud Computing Traffic", *L. Khatibzadeh et al.* examined the network traffic anomaly detection problems in cloud computing environments and proposed a catastrophe theory based approach aiming to depict sudden change processes of the network effectively caused by the dynamic nature of the cloud. Exponential Moving Average (EMA) was applied for the state variable in sliding window to better show the dynamicity of cloud network traffic, and entropy was used as one of the control variables in catastrophe theory to analyse the distribution of traffic features.

In the paper entitled "A Privacy Protection Model of Data Publication Based on Game Theory", *L. Kuang et al.* investigated the user privacy protection problem in sensor acquisition technology because the attacker may identify the user based on the combination of user's quasi-identifiers and the fewer quasi-identifier fields result in a lower probability of privacy leaks. Specifically, they tried to determine an optimal number of quasi-identifier fields under the constraint of trade-offs between service quality and privacy protection. To this aim, the service development process is modelled as a cooperative game between the data owner and consumers, and the Stackelberg game model is leveraged to determine the number of quasi-identifiers that are published to the data development organization. Experiment showed that the data loss of our model is less than that of the traditional k-anonymity especially when strong privacy protection is applied.

In the paper entitled "A Quantum-based Database Query Scheme for Privacy Preservation in Cloud Environment", *W. Liu et al.* studied the privacy protection problems when users access sensitive cloud data and proposed a quantum-based database query scheme for privacy preservation in cloud environment to achieve privacy preservation and reduce the communication complexity. Specifically, all the data items of a database are encrypted by different keys for protecting server's privacy, and the server is required to transmit all these encrypted data items to the client with the oblivious transfer strategy to guarantee the users' privacy. Moreover,

two oracle operations, i.e., modified Grover iteration and special offset encryption mechanism, are combined together to ensure that a user can correctly query a desirable data item.

In the paper entitled "Application of Temperature Prediction based on Neural Network in Intrusion Detection of IoT", *X. Liu et al.* studied the security of network information in IoT and proposed to use a neural network to construct the farmland Internet of Things intrusion detection system to detect anomalous intrusion. They used the temperature data from an IoT acquisition system as the case study and adopted different time granularities for feature analysis. Results showed that the neural network can predict the temperature sequence of varying time granularities better and ensure a small prediction error.

In the paper entitled "Semantic Contextual Search based on Conceptual Graphs over Encrypted Cloud", *Z. Wang et al.* explored the problem of ignorance of context information of the topic sentence when constructing conceptual graph in cloud searchable encryption. To address this problem, the authors defined and constructed semantic search encryption scheme for context-based conceptual graph (ESSEC). The contextual contact was associated with the central key attributes in the topic sentence and its semantic information was extended, so as to improve the accuracy of the retrieval and semantic relevance. Experiments on real data showed that the scheme is effective and feasible.

In the paper entitled "Adaptive DDoS attack detection method based on multiple-kernel learning", *J. Cheng et al.* investigated the distributed denial of service (DDoS) attack problems for Internet security and proposed an adaptive DDoS attack detection method (ADADM) based on multiple-kernel learning (MKL). Five features from the burstiness of DDoS attack flow, the distribution of addresses and the interactivity of communication, were employed to describe the network flow characteristics. A classifier was established to identify an early DDoS attack by training simple multiple-kernel learning (SMKL) models with two characteristics including interclass mean squared difference growth (M-SMKL) and intraclass variance descent (S-SMKL). The sliding window mechanism is used to coordinate the S-SMKL and M-SMKL to detect the early DDoS attacks. The experimental results indicate that this method can detect DDoS attacks early and accurately.

In the paper entitled "A Sequence Number Prediction based Bait Detection Scheme to Mitigate Sequence Number Attacks in MANETs", *R. H. Jhaveri et al.* explored the sequence number attacks which can degrade the network functioning and performance by attracting the sender to establish a path through the adversary node and proposed a proactive secure routing mechanism which makes use of linear regression mechanism to predict the maximum destination sequence number that the neighbouring node can insert in the RREP packet. As an additional security checkpoint, a bait detection mechanism is used to establish the confidence in marking a suspicious node as a malicious node. Results showed that the approach improves the network performance in the presence of adversaries as compared to previous schemes.

In the paper entitled "RoughDroid: Operative Scheme for Functional Android Malware Detection", *K. Riad and L. Ke* studied the malware problems in mobile applications and proposed a floppy analysis approach *RoughDroid*, which can discover Android malware applications directly on a smartphone. *RoughDroid* is based on seven feature sets from the XML manifest file of an Android application and three feature sets from the Dex file. Those feature sets are fed to the Rough Set algorithm to classify the Android application as either benign or malicious elastically. The experimental results showed that *RoughDroid* has 95.6% detection performance for the malware families at 1% false-positive rate.

In the paper entitled "Secure Deduplication Based on Rabin Fingerprinting over Wireless Sensing Data in Cloud Computing", *Y. Zhang et al.* explored the data deduplication problem in cloud computing because existing data deduplication technologies still suffer security and efficiency drawbacks and proposed two secure data deduplication schemes based on Rabin fingerprinting over wireless sensing data in cloud computing. The first scheme is based on deterministic tags and the other one adopts random tags. The proposed schemes realize data deduplication before the data is outsourced to the cloud storage server, and hence both the communication cost and the computation cost are reduced. Our security analysis shows that the proposed schemes are secure against offline brute-force dictionary attacks, and the random tag makes the second scheme more reliable.

In the paper entitled "Enhanced Adaptive Cloudlet Placement Approach for Mobile Application on Spark", *Y. Zhang et al.* investigated the cloudlet placement problem for facilitating mobile computation offloading and proposed an enhanced adaptive cloudlet placement approach named EACP-CA (Enhanced Adaptive Cloudlets Placement approach based on Covering Algorithm) for mobile applications in a given network area. The CA (Covering Algorithm) was used to adaptively cluster the mobile devices based on their geographical locations, and the cloudlet destination locations were also determined according to the clustering centres. The algorithms were implemented on Apache Spark, and the experiment results showed the effectiveness and efficiency of the proposed approach.

In the paper entitled "A Security Sandbox Approach of Android Based on Hook Mechanism", *X. Jiang et al.* studied the security problems in the Android systems and proposed a new security sandbox approach of Android based on hook mechanism to further enrich Android malware detection techniques. The sandbox monitors the behaviours of a target application by using a process hook-based dynamic tracking method during its running period. It can create an isolated virtual space where *apk* can be installed, run, and uninstalled and builds a risk assessment approach based on behaviour analysis. Experiments on malware and normal application samples verified the security of the sandbox.

In the paper entitled "Towards Optimized DFA Attacks on AES under Multibyte Random Fault Model", *R. Wang et al.* investigated the Differential Fault Analysis (DFA) attack problems and pointed out that the state-of-the-art attack is not fully optimized since no clear optimization goal was set. Accordingly, the authors proposed two optimization goals,

i.e., the fewest ciphertext pairs and the least computational complexity, for optimization. To achieve these goals, the corresponding optimized key recovery strategies are identified to further increase the efficiency of DFA attacks on AES. Then, a more accurate security assessment of AES can be completed.

In the paper entitled "Street-Level Landmark Evaluation Based on Nearest Routers", *R. Li et al*. examined the evaluation issues of street-level landmarks for IP geolocation and proposed a street-level landmark evaluation approach based on the nearest router given that the location organization declared is regarded as an area not a point. Specifically, the declared location of preevaluated landmark is verified by IP location databases, and landmarks are grouped according to their nearest routers. The distance constraint is obtained using the delay value between a landmark and its nearest router by delay-distance correlation, based on which a relation model is established among distance constraint, organization's region radius, and distance between two landmarks. The experiment results showed that geolocation errors decrease obviously using evaluated landmarks.

In the paper entitled "Energy-Efficient Cloudlet Management for Privacy Preservation in Wireless Metropolitan Area Networks", *X. Xu et al*. investigated the energy and privacy protection problems in cloudlet based wireless metropolitan area networks (WMAN) and proposed an energy-efficient cloudlet management method, named ECM, for privacy preservation in WMAN. The problem was formulated with an optimization model. Based on the live virtual machine (VM) migration technique, a corresponding privacy-aware VM scheduling method for energy saving was designed to determine which VMs should be migrated and where they should be migrated. Experimental results demonstrated that the proposed method is both efficient and effective.

In the paper entitled "Scheduling Parallel Intrusion Detecting Applications on Hybrid Clouds", *Y. Zhang et al*. examined the scheduling problems in Parallel Intrusion Detection (PID) which can be regarded as a Bag-of-Tasks (BoT) application and proposed to construct an Iterated Local Search (ILS) algorithm which uses an effective heuristic to obtain the initial task sequence and an insertion-neighbourhood-based local search method to explore better task sequences with lower makespans. Specifically, the authors constructed a Fast Task Assignment (FTA) method by integrating an existing Task Assignment (TA) method with an acceleration mechanism to achieve efficiency without loss of any effectiveness. The experiment results showed that the proposed method can outperform the state-of-the-arts.

We strongly believe that this special issue will advance the understanding and research of various intrusion detection and prevention techniques and the closely related privacy and security technologies in cloud, edge/fog and IoT. We hope that the audience will enjoy reading these novel contributions.

Conflicts of Interest

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

Acknowledgments

We would also like to cordially thank all the reviewers who have participated in the review process of the articles submitted to this special issue and the special issue coordinators and the technical supports from the publishing team.

Xuyun Zhang
Yuan Yuan
Zhili Zhou
Shancang Li
Lianyong Qi
Deepak Puthal

Hindawi
Security and Communication Networks
Volume 2019, Article ID 5306395, 11 pages
https://doi.org/10.1155/2019/5306395

Research Article

Applying Catastrophe Theory for Network Anomaly Detection in Cloud Computing Traffic

Leila Khatibzadeh,[1] **Zarrintaj Bornaee**,[1] **and Abbas Ghaemi Bafghi**[2]

[1] *Electrical Engineering and Information Technology Department, Iranian Research Organization for Science and Technology (IROST), Tehran 3353136846, Iran*
[2] *Computer Department, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran*

Correspondence should be addressed to Zarrintaj Bornaee; bornaei@irost.org

Received 18 August 2018; Revised 25 December 2018; Accepted 27 February 2019; Published 2 May 2019

Guest Editor: Yuan Yuan

In spite of the tangible advantages of cloud computing, it is still vulnerable to potential attacks and threats. In light of this, security has turned into one of the main concerns in the adoption of cloud computing. Therefore, an anomaly detection method plays an important role in providing a high protection level for network security. One of the challenges in anomaly detection, which has not been seriously considered in the literature, is applying the dynamic nature of cloud traffic in its prediction while maintaining an acceptable level of accuracy besides reducing the computational cost. On the other hand, to overcome the issue of additional training time, introducing a high-speed algorithm is essential. In this paper, a network traffic anomaly detection model grounded in Catastrophe Theory is proposed. This theory is effective in depicting sudden change processes of the network due to the dynamic nature of the cloud. Exponential Moving Average (EMA) is applied for the state variable in sliding window to better show the dynamicity of cloud network traffic. Entropy is used as one of the control variables in catastrophe theory to analyze the distribution of traffic features. Our work is compared with Wei Xiong et al.'s Catastrophe Theory and achieved a maximum improvement in the percentage of Detection Rate in week 4 Wednesday (7.83%) and a 0.31% reduction in False Positive Rate in week 5 Monday. Additional accuracy parameters are checked and the impact of sliding window size in sensitivity and specificity is considered.

1. Introduction

Nowadays cloud computing is the fastest-growing distributed computational platform in domains such as industries and research communities. In general, connected resources through various distributed networks form the cloud [1]. The network is a pivotal part of the cloud which provides quality of service, namely, ensuring the time constraints. Without it, integrations of various computation and storage resources are impossible [2]. It fulfills two important roles in the cloud environment: interacting with user application for connecting to the appropriate resource and sending back the output to the users [3]. Therefore, the importance of cloud networks has led to attacks on such networks by intruders via malicious attacks which will affect user applications and cloud resources causing a delay in the execution process within the overall cloud computing application [4].

Characterizing and monitoring network traffic, specifically resulting from the outburst in traffic arising from the massive number of cloud tenants that are connected to the internet, is becoming a more complex task. Nowadays, the fast-rising networks duplication, data transfer speed, and unpredictable internet usage have added further anomaly problems [5]. This challenge is even greater in cloud computing environments because its traffic may undergo sudden changes, and the elastic and scalable nature of cloud may easily be confused with traffic anomalies and lead to improper network management [6].

In a traditional network, the nodes are fixed, whilst in the cloud, the nodes are likely to move from one physical machine to others [7]. In the scene of cloud computing, traditional intrusion detection methods lack practicality [8]. The anomaly detection system used in a traditional network cannot be applied to such systems because of the dynamic

nature of logical resources [7]. Traditional network traffic predictors are often modelled on large historical databases. These databases are used for training algorithms. This may not be suitable for such highly unstable environments, where the interaction between past and current values might change quickly over time [9]. In such an environment, coping with the novelty of attacks in such situations is difficult due to various constraints like the unavailability of cloud networks, abundance of network links and devices, network virtualization, unpredictability of the network data, high bandwidth, fast moving network data, dissimilarity, and multitenancy, which lead intruders to exploit cloud networks with different attacks [10]. One of the challenges in predicting network traffic in the cloud is to minimize the computational cost besides maintaining an acceptable levels of accuracy. This issue is not clear while many of the current prediction models are unable to maintain a low computational complexity and dealing with a high degree of workload information over a short span of time [9]. These prediction models form the basis of the anomaly detection algorithm. In past decades, the majority of studies on anomaly detection systems have applied various soft computing, data mining and machine learning approaches for designing anomaly detection systems. Nevertheless, these systems are still inaccurate and involve more computational complexity [7].

Due to these challenges, we present another dynamic method based on Xiong et al.'s Catastrophe Theory [4] to detect network anomalies in the cloud environment. The reason why this theory was chosen in cloud platform is the distribution of processes and their sudden changes in the cloud environment which leads to the malfunctioning of anomaly instead of the dynamics associated factor; entropy is used as a disorderness factor due to its speed and light computing power. Because of the dynamic nature of cloud network traffic, the exponential moving average is introduced to diminish the impact of weights through time. To evaluate the performance of our approach, our methods are validated on the standard Defense Advanced Research Projects Agency data sets and our results are compared with Xiong et al.'s catastrophe model. The results depict that our approach based on modified Catastrophe Theory is effective in the processing time of randomness calculation and Detection Rate to detect cloud network anomalies. Additional accuracy parameters are calculated and compared with Xiong et al.'s.

The rest of the paper is organized as follows. Section 2 covers some of the most outstanding related work. In Section 3, the main conceptual theory of the article and our proposed method with its applying methodology are presented as the materials and methods section. Then we indicate some experimental results and validate the performance of our methods in Section 4. The final section concludes the paper and suggests directions for future work.

2. Related Works

Detecting anomaly, particularly in a safety critical system, is of considerable importance to mitigate any system failures in the future [17]. In some systems, such failures could lead to tremendous environmental catastrophes. Anomaly detection methods make use of a wide range of techniques based on statistics, classification, clustering, nearest neighbor search, and information theory [18]. Network anomaly detection is a source of difficulty due to the dynamic nature of network traffic. In another study, they classified solutions based on techniques taken from statistics, data mining, and machine learning [19].

There are numerous methods for detecting anomalies, some of which will be reviewed in the following section. Bhat et al. introduced the virtual machine monitor anomaly detection model on cloud virtual machines using the machine learning approach. In the first part of their framework, the Naive Bayes (NB) Tree algorithm is applied to classify network connections into intrusion and normal data based on a labeled training dataset. The dataset aids the construction of classification patterns. In the second part of anomaly detection, a hybrid approach of NB Tree and Random forest algorithm is applied based on the likeness of connection features [11]. The method outperforms the traditional Naive Bayes in terms of detection accuracy, error rate, and misclassification cost. In a similar study, Fu et al. proposed another machine learning approach in which one-class and two-class Support Vector Machines are used in cloud computing. The first class formed abnormality score while the second class is retrained for new records appearance [12]. It does not need a prior failure history and is self-adaptive through learning from observed failure events, but the accuracy of failure detection is not fool proof [20].

Zhao and Jin proposed an automated approach to intrusion detection for keeping sufficient performance and reducing execution environment dependence in a VM-based environment. They presented a dynamic graph structure to monitor the dynamic changes in the environment. Based on this structure, a Hidden Markov Model (HMM) strategy for detecting abnormality using frequent system call sequences was considered to automatically and efficiently identify attacks and intrusions. An automated mining algorithm, which is called AGAS, was proposed to generate frequent system call sequences. The AGAS algorithm utilizes related probabilities to identify frequent sequences instead of setting a user-defined threshold on relevant sequences. According to the execution state, the detection performance is adaptively tuned every period. But if the system behavior changes intensively, the overhead of the dynamic graph might be increased and the benefit of their approach will diminish [13].

Jabez et al. proposed a new approach for detecting intrusions within computer networks which is called Outlier Detection [5]. Their training model is made up of big datasets with a distributed environment which is quite similar to the cloud. The performance of their method is superior to other existing machine learning approaches and can significantly detect all anomaly data in computer networks. In another study of this kind, Xinlong Zhao et al. proposed a new intrusion detection method based on improved K-means. The method is designed to fit the characteristics and security requirements of cloud computing. It includes a clustering algorithm and a method for distributed intrusion detection modelled on it. The new method can detect known attacks

in addition to anomaly attacks in the cloud computing environment. The result of a simulation test proves that this improved method can decrease the false positive and false negative rate and speed up the intrusion detection process [8]. Pandeeswari and Kumar also used the clustering method and proposed an anomaly detection system at the hypervisor layer which employs a hybrid algorithm (FCM-ANN), a combination of the Fuzzy C-Means clustering algorithm and Artificial Neural Network, in order to improve the accuracy of the detection system. FCM-ANN can automatically capture new attack patterns, so there is no necessity to manually update the database. Compared with the Naïve Bayes classifier and Classic ANN algorithm, their system can detect anomalies with a high detection accuracy and a low false alarm rate. Low frequent attacks can also be detected. It outperforms the Naïve Bayes classifier and Classic ANN [14]. Later, in 2016, they proposed an anomaly detection system named hypervisor detector in the cloud environment. It is designed with ANFIS (integration of fuzzy systems with adaptation and learning proficiencies of neural network called adaptive neurofuzzy inference system) in order to detect anomalies in the cloud network. The ANFIS used the back propagation gradient descent technique in combination with the least square scheme for the purpose of training and testing the system. The comparison results with other IDS using ANN, Naive Bayes, and a hybrid approach that combined Naïve Bayes and random forest indicate that the proposed model is both efficient and effective in discovering the anomalies in the cloud environment and is ideal for detecting anomalies with high detection accuracy and a minimum false negative rate. By employing hypervisor detector in very large datasets, it is possible to attain the best performance in the cloud environment, but the complexity of the algorithm makes its implementation difficult for such an environment [7].

Xiong et al. proposed two anomaly detection models based on catastrophe theory in network traffic [21, 22]. After that, in 2014 they proposed an intrusion detection system to detect network traffic anomaly based on synergetic neural networks and the catastrophe theory to reduce security risks in the cloud network. The results show high detection and low false alarm rates [4]. They adopted impulsive neural networks in addition to fast compression algorithms to examine network traffic anomaly in the cloud computing environment. They subsequently proposed the idea of network-based intrusion detection on the cloud platform but did not provide a clearer definition of the anomaly [8]. Approaches based on pattern recognition, such as first part of Xiong et al. which is related to neural networks, involve a severe disadvantage: they learn the training patterns but lack the ability to make generalizations, so the model may give inaccurate results for unknown patterns [9]. In the second part of Xiong et al.'s paper, the sudden change process of the network is shown by the catastrophe potential function. An index referred to as catastrophe distance is presented to assess deviations from normal behavior in detecting network anomalies.

In some cases, detecting network anomalies is performed based on traffic prediction. Network traffic prediction has received a great deal of attention for facilitating monitoring and managing computer networks [6]. In this field, most research efforts are focused on classical methods strongly based on historical data. Predicting network traffic is pertinent to many management applications such as resource allocation, admission control, and congestion control [10]. The major issue with these models is the computational overhead in relation to the size of the input data [9], which could be more intense in cloud computing due to its volatile and extensive environment. Yuehui Chen et al. use genetic programming to build a Flexible Neural Tree (FNT) for predicting network traffic online [15]. This approach was employed for a better comprehension of the main features of traffic data. Moreover, the proposed method is able to predict short time scale traffic measurements and can reproduce statistical features related to real traffic measurements. However, it needs initial input which is dependent on the characteristics of data that is being evaluated to achieve proper results [10]. Moayedi and Shirazi proposed a different model for predicting network traffic and detecting anomalies based on Autoregressive Integrated Moving Average (ARIMA). In order to isolate anomalies from normal traffic variation, they decompose the data flow. The authors try to predict anomalies independently from normal traffic. They evaluated their work with synthetic data, which depends on large historic data [16]. Although these works allow online traffic prediction, due to their dependence on large historical data for training the algorithms, they are inappropriate in the cloud environment [6]. To address this issue, Dalmazo et al. proposed a dynamic window size methodology for traffic prediction. The size of the window is related to the amount of traffic for traffic prediction and varies according to bounded historical data by using network traffic features such as short-range dependence [6]. They estimated network traffic through a statistical method based on a Poisson process. Their mechanism can be applied to determine the scope of data to be analyzed for any traffic prediction approach.

Table 1 presents these approaches and groups them according to their methods outlining their advantages and disadvantages.

According to works related to anomaly detection, Xiong et al. [4] used fast compression algorithms to examine network traffic anomaly in the cloud computing environment. They pay much attention to the sudden change process of network traffic which other works have not addressed. This is why we have chosen to use this theory in the present study.

3. Materials and Methods

Cloud computing provides a dynamic environment with complex network traffic behavior and without having linear trend pattern; therefore, high degree polynomials are required to fit the network traffic baseline [9]. The dynamic nature of cloud network traffic flow depends on equilibrium changes determined by primary factors. In normal network traffic, when no anomalies occur, the network state is referred to as the normal state of equilibrium. When anomalies occur, the network state will transform from a normal equilibrium state to an abnormal one driven by abnormal factors. The change process of the network traffic is regarded to be

Table 1: Comparison of anomaly detection approaches via their methods.

Methods	Author Name	Techniques	Pros and Cons
Classification	Bhat et al. [11] Fu et al. [12] Feng Zhao and Hai Jin [13]	NB Tree & Random Forest One Class and Two Class Support Vector Machines Hidden Markov Model & mining algorithm	Through powerful algorithms, the method can distinguish between different class instances but they depend on labels. This is often impossible to achieve.
Nearest neighbor	Jabez et al. [5]	Outlier Detection	They are commonly used because they are unsupervised and do not require any data distribution, but for unsupervised techniques, if the normal data instances lack close enough neighbors or the anomalies have close enough neighbors, the technique fails to label them and computing complexity is, therefore, a challenge.
Clustering	Xinlong Zhao et al. [8] Pandeeswari and Kumar [14]	K-means Clustering Fuzzy C-means Clustering & Artificial Neural Network	The method is relatively faster than distance-based methods and they could reduce the computational complexity during the process of detecting intrusions in large datasets, but in smaller datasets, they may not provide accurate insights at the desired level of detail and dynamic updating of profiles is time consuming.
Statistical	Kumar and Pandeeswari [7] Xiong et al. [4]	Fuzzy system & Neural Network Neural Network & Catastrophe theory	A statistically-justifiable solution for detecting anomalies can be yielded from these methods, if the assumptions considering the data distribution hold true and the confidence interval for the anomaly score can be applied for decision making as additional information, but they are dependent on the assumption that the data is generated via a specific distribution.
Prediction	Yuehui Chen et al. [15] Moayedi and Shirazi [16] Dalmazo et al. [6]	Flexible Neural Tree Autoregressive Integrated Moving Average Poisson Moving Average	In some cases, predicting anomalies is done independently from normal traffic, but they almost depend on large historic data.

transient and catastrophic [4]. As a result, the Catastrophe Theory is used to better display fluctuations in network traffic. But as mentioned in the previous section, Xiong et al. did not provide a clearer definition of anomalies which leads to not clearly defining network traffic behavior. In this article, in order to better describe traffic behavior, EMA is used.

In the following sections, we will first explain the details of this theory and then present the proposed method based on it.

3.1. Catastrophe Theory. Catastrophe Theory is a method for describing the evolution of forms in nature and is one of the main branches of dynamic systems. It was invented by Rene Thom in the 1960s [23] who explained the philosophy behind the theory in his 1972 book *Structural Stability and Morphogenesis*. Catastrophe Theory is specifically applicable to situations where gradually changing forces produce sudden effects. The applications of catastrophe theory in classical physics (or more generally in any subject governed by a minimization principle) provide us with a better understanding of what diverse models have in common [24]. The theory is derived from a branch of mathematics (topology) concerned with the properties of surfaces in many dimensions. Topology is involved because smooth surfaces of equilibrium describe the underlying forces in nature: when the equilibrium breaks down, catastrophe occurs. The strength of the model derived from catastrophe theory is that it can account for the bimodal distribution of probabilities [25]. "The line that marks the edges of the pleat in the behavior surface, when the top and bottom sheets fold over to form the middle sheet, is called the fold curve. When it is projected back onto the plane of the control surface, the result is a cusp-shaped curve" [26]. For this reason, the model is called the cusp catastrophe. It is one of the simplest of the seven elementary catastrophes, and so far, it has been the most productive [27]. In this paper, the cusp or Riemann-Hugoniot [28] catastrophe model is used to reveal the network traffic anomaly in the cloud.

In a sense, elementary catastrophe theory is a generalization of theorems about critical points or singularities of real-valued functions of n real variables to one about parameterized families of such functions. Catastrophe theory analyzes degenerated critical points of the potential function. The critical points satisfy the condition that the first derivative and higher derivatives of the potential function are zero [4]. These are called the germs of the catastrophe geometries. The degeneracy of the critical points can be unfolded by expanding the potential function as a Taylor series in small perturbations of the parameters. What Thom has done is to determine conditions on derivations of f which ensure the existence and which give a local normal form for a stable unfolding of f. He also shows that whenever the parameter space or control space has dimension ≤ 5, there is a finite classification of stable unfolding [28].

The potential function $F(x)$ of the cusp catastrophe model is shown in [24]

$$F(x) = x^4 + aux^2 + bvx \tag{1}$$

where x is a state variable, u, v are control variables, and a, b are the coefficients [22]. The equilibrium surface has the equation $F'(x) = 0$ and $G(x, u, v)$ which can be achieved by

$$4x^3 + 2aux + bv = 0. \tag{2}$$

The normal to the surface is vertical when $F''(x) = 0$ and the singularity set of the cusp catastrophe is achieved by

$$6x^2 + au = 0. \tag{3}$$

The bifurcation set (cusp) is the critical image of the projection $(u,v,x) \longrightarrow (u,v)$ from the equilibrium surface onto the control space. The equation difference set $G(x, u, v)$ of the cusp is as follows:

$$8a^3u^3 + 27b^2v^2 = 0 \tag{4}$$

It is obtained by eliminating x from (2) and (3) for the fold curve.

The equilibrium surface has equation $4x^3 + 2aux + bv = 0$ where (u, v) are coordinates on the control space and the vertical coordinate x is only state variable. As the control (u, v) varies, the state (u, v, x) will be forced to jump to the other sheet when it crosses the fold curve. The curve over the cusp is shown in Figure 1. The top surface is the equilibrium surface of the cusp catastrophe model, which is divided into the upper sheet and lower sheet [24]. When the state of the system transfers from the stable equilibrium state to another stable equilibrium state, there is a sudden jump between the stable states and then sudden change appears. The bottom one is the control space illustrated by the control variables u, v.

3.2. Proposed Method. In this part of the section, we propose our anomaly detection method based on the modified Catastrophe Theory and discuss our contributions based on Xiong et al.'s Catastrophe Theory to detect anomalies in cloud network traffic. The reason why we followed this theory could be due to their attention to sudden change process of cloud network traffic that most of the previous works did not pay much attention to. By studying this theory, we realized that it could be possible to generalize it to similar problems in detecting abnormalities in similar conditions.

How to extract state and control variables plays a significant role in the accurate analysis of the model. In our model, the Hurst index [29] reflects the degree of the self-similarity between the current and next state of the network traffic and was selected as u like the one that Xiong et al. used [4]. Nevertheless, the majority of algorithms for the similarity-based detection of anomalies use a multivariate distance function and these functions are susceptible to the problem of dimensionality, which means they are unable to provide a reliable measurement of the similarity of high-dimensional data because of the data dispersion in high dimensional spaces [18]. In addition, such functions cannot localize the source of anomaly and detect the specific dimensions that cause anomalous patterns [18]. We use a damping coefficient to diminish the effect of similarity versus randomness in cloud network traffic. In the field of engineering, the damping

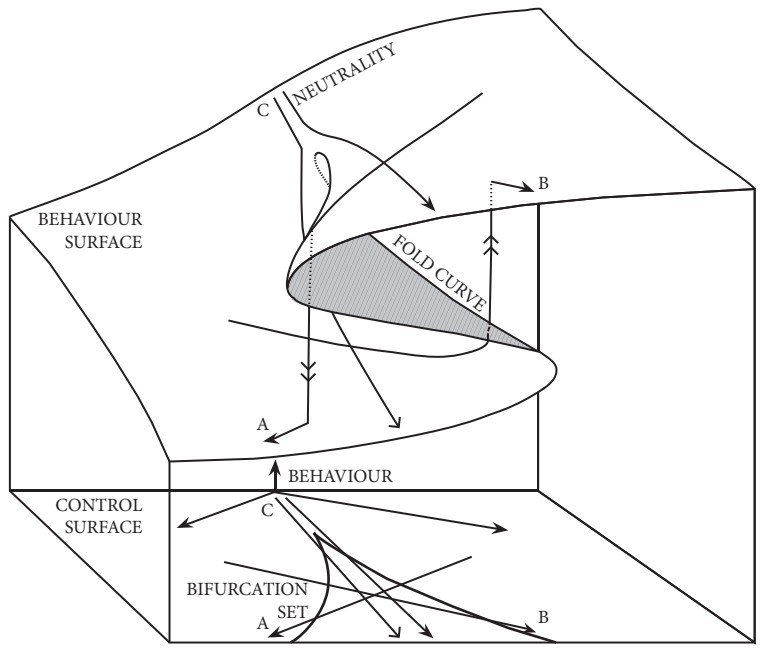


Figure 1: Cusp catastrophe model [27].

coefficient is a dimensionless measure which describes how oscillations decay after a disturbance [30]. Many systems show oscillatory behavior upon being disturbed from their position of static equilibrium [31]. Larger values of the damping coefficient or damping factor produce transient responses with a minor oscillatory nature, which is similar to the dynamic nature of cloud network traffic.

Entropy reflects the level of irregularities that occur or in other words, it is a measure of disorder, and we have selected it as control variable v. The entropy-based method needs little computing power and is fast enough for detecting anomalies [17]. These features are appropriate for our purpose in our case. We utilize the entropy concept for analyzing the randomness distribution of features in cloud network traffic.

The state variable x (of the cusp catastrophe model) was taken as the volume of the network traffic. We use a sliding window to construct a vector set. Exponential Moving Average (EMA) is applied in constructing x value due to the volume of traffic accumulated in each sliding window.

In [32], Frank Klinker defines a mathematical tool for the prediction of market trends. Specifically, he states that it is possible to use the Exponential Moving Average (EMA) in order to efficiently forecast network traffic with short historical data. In the case of EMA, weighting coefficients increase exponentially through the time in the sliding window. The weighing for the oldest values in each sliding window reduces exponentially and never reaches zero unlike most other moving averages, so this approach reacts faster to recent value changes. Similar to other techniques that make use of a moving average, EMA should strictly be used for data that does not involve seasonal behavior [9] and according to the dynamic nature of cloud traffic and because of online detection; we have chosen to use this kind of moving average instead of the others. The formula for calculating EMA is as follows:

$$EMA = \frac{\sum_{i=s}^{n+s} \left(exp^{i} * y_i\right)}{\sum_{i=s}^{n+s} \left(exp^{i}\right)}. \tag{5}$$

In this formula, n is the size of time window and y is the number of packets which are received in each i seconds. Because of applying sliding window and various x for each time window, the s parameter which changes the basis of the formula indicates the start time of each time window and then the result reflected by EMA replaces the x variable for each time window.

After calculating the parameters (x, u, v) for each sliding window in train and test data, we must calculate the catastrophe distance (Dp) between the observed point in test data and the bifurcation set $G(x, u, v)$. Assume one point (P_t) of the equilibrium surface $G(x, u, v)$ and one point of test data (P_i); the catastrophe distance between these two points, labelled as "$D_E(P_i,P_t)$, is computed by the Euclidean distance" [22] as shown in

$$D_p\left(P_i, G\left(x, u, v\right)\right) = \min_{P_t e G(x,u,v)} \left\{D_E\left(P_i, P_t\right)\right\}. \tag{6}$$

When the catastrophe distance Dp is beyond a given threshold η, there is an anomaly which exists at the observing point in the test data.

4. Results and Discussion

This section presents the experimental results to evaluate the proposed anomaly detection method. The standard DARPA datasets used in our experiments are widely used in network intrusion detection, to name a few, Horng et al. [33], Shon

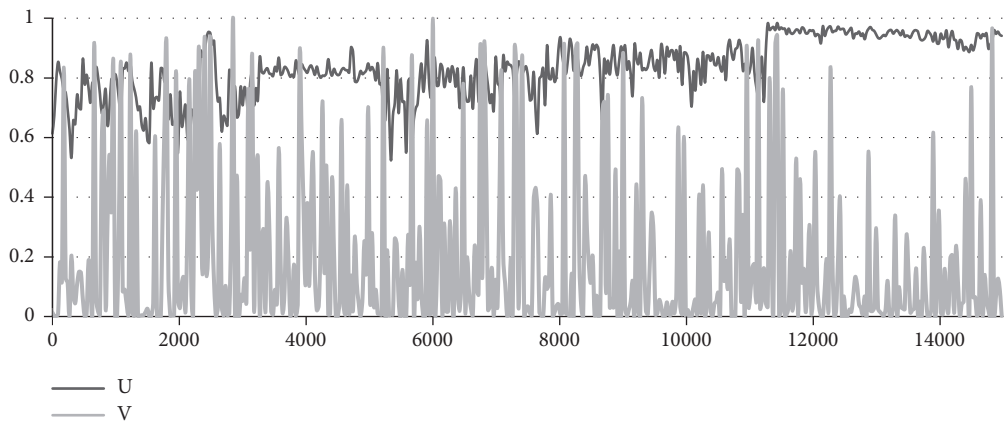


Figure 2: A trade-off between u & v control variables in week 5 Tuesday; the horizontal axis represents time.

and Moon [34], and Xiong et al. [4]. Although there are new intrusion datasets such as ISCX 2012, NSL-KDD 2013, and CIC 2017, DARPA dataset is used to compare the results with Xiong et al.'s paper [4]. It contains 5 weeks of data, whose 4 weeks are utilized in our implementation. Weeks 1 and 3 traffic data included no attack. Traffic data from these 2 weeks are used as train datasets. Weeks 4 and 5 traffic data included different types of attacks mixed with normal traffic and are used as test datasets. In this paper, we extract "the aggregated network traffic in packets per second from the tcpdump data files" [4] in the DARPA data set [35].

For all experiments, processing time, the number of false positives, and the number of true anomalies which can be detected are reported. According to attack file which was published with DARPA, false positive and true positive could be calculated. These experiments are processed at 8 Core Xenon Server with 2.19 GHz CPU frequency with 16 GB memory.

4.1. Parameter Analysis. We calculate each state and control variables based on weeks 1, 2, 4, and 5. As the input data, we produced a list of catastrophe distance before applying the algorithm for detecting anomalies. "The given parameters p, η were chosen as $p = 30$ and $\eta = 0.85$" like Xiong et al.'s [4]. We change the threshold in 5 days of test data, and in 3 cases, the number 0.8 reached better results but because of the comparison, 0.85 is chosen as η.

We randomly choose control parameters from test files in weeks 4 and 5. Then compare their distribution achieved in each random interval, namely, in the first 15000 seconds of week 5 Tuesday which is depicted in Figure 2. Due to the dynamic nature of the cloud, the randomness control parameter is dominant, but as they proceed and time elapsed, the self-similarity parameter does not show any significant changes. To reduce the impact of self-similarity control factor, the damping coefficient has been used. The selection of the damping coefficient for such an application needs a trade-off between the maximum percentage of self-similarity and the time of the peak in which self-similarity occurs. A smaller damping coefficient reduces the time, but it enhances the maximum percent of similarities which is not desirable for cloud network traffic. The final choice of the damping coefficient is subjective. It has been Shinners' experience that "the damping coefficient range is usually selected between 0.4 and 0.7 for general cases" [36]. Experiments were performed to determine the amount of damping coefficient whose [0.6, 0.7] interval results the best and in most cases, 0.69 resulted better, so in this experiment $\zeta = 0.69$ was considered.

4.2. Experimental and Comparisons. Detection Rate and False Positive Rate are common metrics for assessing the effectiveness of an anomaly detection system. The Detection Rate (DR) is the number of correctly classified as normal packets divided by the total number of test data (or true negative plus false positive). The False Positive Rate (FPR) is defined as the total number of normal data, which was classified as anomalies wrongly, divided by the total number of normal data traffic (or true negative plus false positive) [37] as shown in the following, respectively:

$$DR = \frac{N_{detected}}{TN + FP} \tag{7}$$

$$FPR = \frac{NF_{detected}}{TN + FP}. \tag{8}$$

The detection results based on Xiong et al. are depicted in Figures 3 and 4 [4]. We implemented the Xiong et al. article with C# programming language and repeated the experiments. Then, our method is implemented and these improvements in DR and FPR are achieved; the average rates of these improvement percentages in two weeks are 2.24 and 0.069 promotion, respectively. The results show that, in most days, DR is improved, but FPR could not show the best results contrary to what we expected. It may be possible if we change the damping coefficient due to the trade-off between self-similarity percentages versus disorderness percentage.

The attack file, which was published by MIT Lincoln Laboratory [35], has two main parts for each attack; the first one is the ID of each attack that contains some subattacks with different start times and durations. In this comparison, we use each attack with its ID information and did not consider any subinformation which is distinct between each part of an

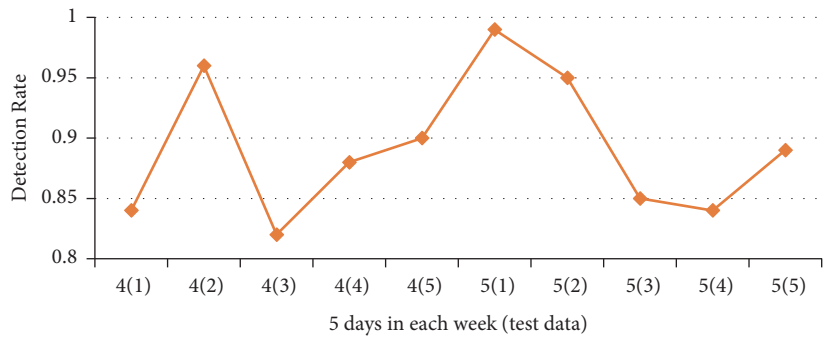


Figure 3: DR of each day in weeks 4 and 5 in Xiong et al.'s catastrophe theory [4].

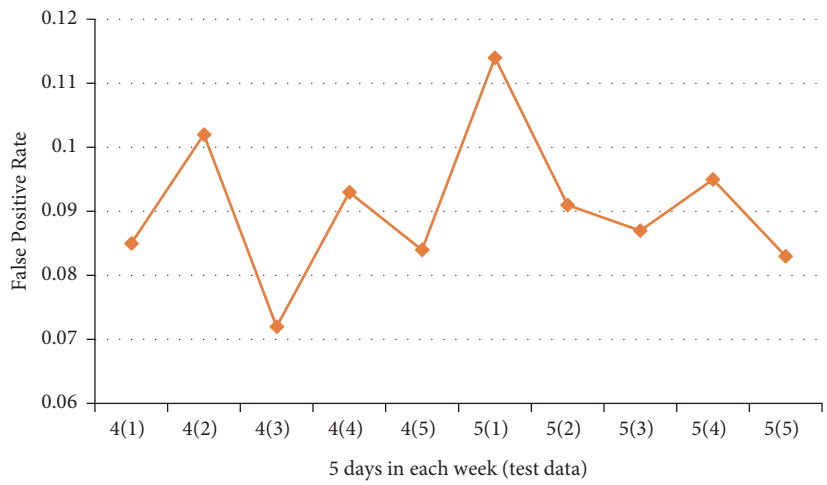


Figure 4: FPR of each day in weeks 4 and 5 in Xiong et al.'s catastrophe theory [4].

attack. In the second experiments, details are considered. Due to this change, the results are more accurate. For instance, in week 4 Thursday, the improvement percentage of DR is 7.83% in accordance with details implemented as Xiong et al.'s catastrophe theory. In the same day, the improvement percentage of FPR shows 0.042% reduction. But in week 5 Tuesday, we could not reach the ideal. In week 4 Friday and week 5 Wednesday, DR and FPR rates repeated exactly. The results of our implementation based on DR and FPR improvement percentage are indicated in Table 2. The result of the comparison on our model shows better precision.

How correctly an intrusion detection system works is measured by a metric referred to as accuracy. It measures the percentage of detection and failure as well as the number of false alarms produced [38]. We compared our model with Xiong et al.'s in some accuracy measures as follows.

(a) Misclassification rate estimates the probability of disagreement between the true and predicted cases by dividing the sum of FN and FP by the total number of pairs observed. In other words, misclassification rate is defined as [38] follows:

$$mis-classification\ rate = \frac{(\mathrm{FN}+\mathrm{FP})}{(\mathrm{TP}+\mathrm{FP}+\mathrm{FN}+\mathrm{TN})}. \quad (9)$$

The improvement percentage of misclassification rate between Xiong et al.'s model and our proposed model is indicated in Table 3.

(b) Precision is a measure of how a system identifies abnormality or normality. Precision is used to measure the exactness of the detection. We calculated this factor in our model and compared it with Xiong et al.'s, and the improvement percentage is shown in Table 3.

(c) The F-measure mixes the values of two previous measures (precision and recall). By considering only one metric for evaluation, F-measure is the most preferable. It is calculated as follows:

$$F-measure = \frac{2}{1/precision + 1/recall}. \quad (10)$$

Table 3 depicts the results of F-measure improvement in 2-week test data.

Considering processing time, one could notice that our method is more efficient than Xiong et al.'s in calculating randomness parameters. Comparisons of processing time in calculating ν control parameter for each day in train and test data are shown in Tables 4 and 5, respectively. In Table 4, for instance, in week1 Monday, the processing time with Xiong

Table 2: DR & FPR improvement percentage in each day of test data between Xiong et al. and proposed model.

Weeks include test data►	W4(1)	W4(2)	W4(3)	W4(4)	W4(5)	W5(1)	W5(2)	W5(3)	W5(4)	W5(5)
DR Improvement Percentage	0.15%	7%	3.2%	7.83%	-	4.29%	reduced	-	0.033%	0.30%
FPR Improvement Percentage	0.025%	0.03%	0.031%	0.042%	-	0.31%	increased	-	0.16%	0.15%

Table 3: Misclassification rate, precision, and F-measure improvement in each day of test data between Xiong et al. and proposed model.

Weeks include test data►	W4(1)	W4(2)	W4(3)	W4(4)	W4(5)	W5(1)	W5(2)	W5(3)	W5(4)	W5(5)
Misclassification rate Improvement Percentage	0.253%	0.232%	0.647%	0.207%	0	0.315%	reduced	0	0.106%	0.570%
Precision Improvement Percentage	0.008%	0.006%	0.016%	0.008%	0	0.028%	reduced	0	0.010%	0.018%
F-measure Improvement Percentage	0.04%	0.018%	0.104%	0.027%	0	0.019%	reduced	0	0.017%	0.064%

Table 4: Comparison of processing time in calculating randomness for each day in train weeks.

Weeks include train data►	W1(1)	W1(2)	W1(3)	W1(4)	W1(5)	W3(1)	W3(2)	W3(3)	W3(4)	W3(5)
Execution time of Xiong et al. Randomness algorithm [sec]	0.091	0.068	0.083	0.083	0.105	0.087	0.098	0.097	0.094	0.091
Execution time of our Randomness algorithm [sec]	0.045	0.05	0.036	0.039	0.042	0.047	0.049	0.053	0.054	0.057
Differences between two time [sec]	0.046	0.018	0.047	0.044	0.063	0.04	0.049	0.044	0.04	0.034
Improvement percentage (%)	50.55	26.47	56.63	53.01	60.00	45.98	50.00	45.36	42.55	37.36

Table 5: Comparison of processing time in calculating randomness for each day in test weeks.

Weeks include test data►	W4(1)	W4(2)	W4(3)	W4(4)	W4(5)	W5(1)	W5(2)	W5(3)	W5(4)	W5(5)
Execution time of Xiong et al. Randomness algorithm [sec]	0.104	0.124	0.11	0.117	0.147	0.123	0.125	0.129	0.142	0.161
Execution time of our Randomness algorithm [sec]	0.084	0.094	0.079	0.074	0.078	0.084	0.087	0.089	0.094	0.097
Differences between two time [sec]	0.02	0.03	0.031	0.043	0.069	0.039	0.038	0.04	0.048	0.064
Improvement percentage (%)	19.23	24.19	28.18	36.75	46.94	31.71	30.40	31.01	33.80	39.75

et al. algorithm is 0.091 s but in our algorithm, the time is 0.045 s which improved 50%. In Table 5, for example in week 4 Friday, the processing time with Xiong et al. algorithm is 0.147 s but in our algorithm, the time is 0.078 s which improved approximately 47%. As implicated in Tables 4 and 5, our method is approximately two times faster than Xiong et al.'s algorithm in terms of efficiency.

4.3. Sensitivity and Specificity Analysis. As mentioned about sliding window size and because of the comparison, we select the same window size as Xiong et al.'s. On the other hand, applying different window sizes and studying their impacts on sensitivity have always been a question for us. So, other implementations based on different window size have been performed. TPR which is also known as sensitivity and TNR which is also called specificity are considered. The results of Thursday week 5 are indicated in Table 6.

Table 6: Sensitivity and Specificity in Thursday Week 5 with different sliding window size.

Window Size►	30	40	50	60	70
TPR (Sensitivity)	86	98	100	99	97
TNR (Specificity)	80	86	86	86	85

5. Conclusions and Future Work

Security threats from inside and outside the cloud make security a major challenge in the widespread cloud adoption. One of the main challenges of the cloud is the tremendous amount of network traffic and the diversity of cloud tenants which make controlling the traffic and preventing intrusion difficult to achieve. We should enhance the security of networks in cloud computing by applying intrusion detection

systems which are capable of detecting sudden changes in traffic as fast as possible.

In our work, another dynamical method based on catastrophe theory is presented to detect anomalies in a cloud network. A damping coefficient is introduced for controlling the effectiveness of self-similarity factor. Entropy is used as disorderness factor versus self-similarity in control parameters of cusp catastrophe theory. To reduce the impact of weights through the time because of the dynamic nature of cloud traffic, the exponential moving average is applied. To evaluate the performance of our approaches, we consider DARPA datasets and compare the results with Xiong et al. catastrophe model. The results depict that our approach grounded in modified catastrophe theory is more effective in DR, FPR, misclassification, and F-measure rates to detect cloud network anomalies. We indicate that our randomness method based on entropy is two times faster than what Xiong et al. preferred. Different size of sliding window is applied and the maximum sensitivity is caught in window size 50. We prefer to repeat the experiments with new window size and compared the results in near future. As a future work, we would like to analyze a trade-off between the accuracy we achieved and the speed of detection. Also, we would like to test the impact of distinguishing protocol types in DARPA datasets and consider the differences. One of the ways which better indicates the performance of our model is to implement in a real environment and review the results. This work will be done as future work.

Data Availability

The DARPA datasets used to support the findings of this study were used in previously reported studies and supplied by Lincoln Laboratory and could be available in https://www.ll.mit.edu/r-d/datasets.

Conflicts of Interest

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

Acknowledgments

This research was supported by the Iranian Research Organization for Science and Technology (IROST), Ministry of Science, Research & Technology (MSRT). We thank technology and communication center of IROST who provided a server that greatly assisted the research.

References

[1] S. Khan, A. Gani, A. W. Abdul Wahab et al., "Towards an applicability of current network forensics for cloud networks: a SWOT analysis," *IEEE Access*, vol. 4, pp. 9800–9820, 2016.

[2] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Information Technology Laboratory, 2011.

[3] C. Gong, J. Liu, Q. Zhang et al., "The characteristics of cloud computing," in *Proceedings of the 39th International Conference on Parallel Processing Workshops (ICPPW '10)*, pp. 275–279, San Diego, California, Calif, USA, 2010.

[4] W. Xiong, H. Hu, N. Xiong et al., "Anomaly secure detection methods by analyzing dynamic characteristics of the network traffic in cloud communications," *Information Sciences*, vol. 258, pp. 403–415, 2014.

[5] J. Jabez and B. Muthukumar, "Intrusion detection system (IDS): anomaly detection using outlier detection approach," *Procedia Computer Science*, vol. 48, pp. 338–346, 2015.

[6] B. L. Dalmazo, J. P. Vilela, and M. Curado, "Online traffic prediction in the cloud: a dynamic window approach," in *Proceedings of the International Conference on Future Internet of Things and Cloud (FiCloud '14)*, Barcelona, Spain, August 2014.

[7] P. G. Kumar and N. Pandeeswari, "Adaptive neuro-fuzzy-based anomaly detection system in cloud," *International Journal of Fuzzy Systems*, vol. 18, no. 3, 2016.

[8] X. Zhao and W. Zhang, "An anomaly intrusion detection method based on improved k-means of cloud computing," in *Proceedings of the 6th International Conference on Instrumentation and Measurement, Computer, Communication and Control (IMCCC '16)*, pp. 2519–2523, July 2016.

[9] B. L. Dalmazo, J. P. Vilela, and M. Curado, "Performance analysis of network traffic predictors in the cloud," *Journal of Network and Systems Management*, vol. 25, no. 2, pp. 290–320, 2017.

[10] B. L. Dalmazo, J. P. Vilela, and M. Curado, "Online traffic prediction in the cloud," *International Journal of Network Management*, vol. 26, no. 4, pp. 269–285, 2016.

[11] A. H. Bhat, S. Patra, and D. Jena, "Machine learning approach for intrusion detection on cloud virtual machines," *International Journal of Application or Innovation in Engineering and Management*, vol. 2, no. 6, 2013.

[12] S. Fu, J. Liu, and H. Pannu, "A hybrid anomaly detection framework in cloud computing using one-class and two-class support vector machines," in *Advanced Data Mining and Applications*, pp. 726–738, Springer, Berlin, Germany, 2012.

[13] F. Zhao and H. Jin, "Automated approach to intrusion detection in VM-based dynamic execution environment," *Computing and Informatics*, vol. 31, no. 2, 2012.

[14] N. Pandeeswari and G. Kumar, "Anomaly detection system in cloud environment using fuzzy clustering based ANN," *Mobile Networks and Applications*, vol. 21, no. 3, pp. 494–505, 2016.

[15] Y. Chen, B. Yang, and Q. Meng, "Small-time scale network traffic prediction based on flexible neural tree," *Applied Soft Computing*, vol. 12, no. 1, pp. 274–279, 2012.

[16] H. Z. Moayedi and M. Masnadi-Shirazi, "ARIMA model for network traffic prediction and anomaly detection," in *Proceedings of the International Symposium on Information Technology (ITSim '08)*, vol. 4, pp. 1–6, 2008.

[17] A. Waskita, H. Suhartanto, and L. T. Handoko, "A performance study of anomaly detection using entropy method," in *Proceedings of the International Conference on Computer, Control, Informatics and its Applications (IC3INA '16)*, October 2016.

[18] E. Menahem, A. Schclar, L. Rokach, and Y. Elovici, "XML-AD: detecting anomalous patterns in XML documents," *Information Sciences*, vol. 326, pp. 71–88, 2016.

[19] F. Iglesias and T. Zseby, "Analysis of network traffic features for anomaly detection," *Machine Learning*, vol. 101, no. 1-3, pp. 59–84, 2015.

[20] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Procedia Computer Science*, vol. 60, pp. 708–713, 2015.

[21] W. Xiong, N. Xiong, L. T. Yang et al., "Network traffic anomaly detection based on catastrophe theory," in *Proceedings of the IEEE Globecom Workshops*, pp. 2070–2074, January 2011.

[22] W. Xiong, N. Xiong, L. T. Yang, J. H. Park, H. Hu, and Q. Wang, "An anomaly-based detection in ubiquitous network using the equilibrium state of the catastrophe theory," *The Journal of Supercomputing*, vol. 64, no. 2, pp. 274–294, 2013.

[23] R. Thom, "Structural stability, catastrophe theory, and applied mathematics," *SIAM Review*, vol. 19, no. 2, pp. 189–201, 1977.

[24] J. W. Robbin, "Thom's catastrophe theory and zeeman's model of the stock market," *Chaos and Complexity Seminar*, 2013.

[25] H. J. Sussmann and R. S. Zahler, "Catastrophe theory as applied to the social and biological sciences: a critique," *Synthese*, vol. 37, no. 2, pp. 117–216, 1978.

[26] S. Qin, J. Jimmy Jiao, S. Wang, and H. Long, "A nonlinear catastrophe model of instability of planar-slip slope and chaotic dynamical mechanisms of its evolutionary process," *International Journal of Solids and Structures*, vol. 38, no. 44-45, pp. 8093–8109, 2001.

[27] E. C. Zeeman, "Catastrophe theory," *Scientific American*, vol. 234, no. 4, pp. 65–83, 1976.

[28] M. Golubitsky, "An introduction to catastrophe theory and its applications," *SIAM Review*, vol. 20, no. 2, pp. 352–387, 1978.

[29] X. Wang and B.-X. Fang, "An exploratory development on the Hurst parameter variety of network traffic abnormity signal," *Journal of Harbin Institute of Technology*, vol. 37, no. 8, pp. 1046–1049, 2005.

[30] D. G. Alciatore, *Introduction to Mechatronics and Measurement Systems*, McGraw Hill, New York, NY, USA, 4th edition, 2012.

[31] M. Rao and H. Qiu, *Process Control Engineering: A Textbook for Chemical, Mechanical and Electrical Engineers*, CRC Press, Boca Raton, Fla, USA, 1993.

[32] F. Klinker, "Exponential moving average versus moving exponential average," *Mathematische Semesterberichte*, vol. 58, no. 1, pp. 97–107, 2011.

[33] S.-J. Horng, M.-Y. Su, Y.-H. Chen et al., "A novel intrusion detection system based on hierarchical clustering and support vector machines," *Expert Systems with Applications*, vol. 38, no. 1, pp. 306–313, 2011.

[34] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection," *Information Sciences*, vol. 177, no. 18, pp. 3799–3821, 2007.

[35] Lincoln Laboratory, "The 1999 DARPA intrusion detection datasets," 1999, https://www.ll.mit.edu/r-d/datasets.

[36] S. M. Shinners, *Modern Control System Theory and Design*, Wiley InterScience Publication, 2nd edition, 1992.

[37] B. L. Dalmazo, J. P. Vilela, P. Simoes, and M. Curado, "Expedite feature extraction for enhanced cloud anomaly detection," in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium, NOMS '16*, July 2016.

[38] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 303–336, 2014.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 4923590, 14 pages
https://doi.org/10.1155/2019/4923590

Research Article

A Quantum-Based Database Query Scheme for Privacy Preservation in Cloud Environment

Wenjie Liu,[1,2,3] Peipei Gao,[2] Zhihao Liu,[3,4] Hanwu Chen,[3,4] and Maojun Zhang[5,6]

[1] *Jiangsu Engineering Centre of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing, China*
[2] *School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China*
[3] *School of Computer Science and Engineering, Southeast University, Nanjing, China*
[4] *Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing, China*
[5] *School of Mathematics and Computing Science, Guilin University of Electronic Technology, Guilin, China*
[6] *College of Computer Science and Information Technology, Guangxi Normal University, Guilin, China*

Correspondence should be addressed to Wenjie Liu; wenjiel@163.com and Maojun Zhang; zhang1977108@sina.com

Received 16 August 2018; Accepted 26 February 2019; Published 1 April 2019

Guest Editor: Yuan Yuan

Cloud computing is a powerful and popular information technology paradigm that enables data service outsourcing and provides higher-level services with minimal management effort. However, it is still a key challenge to protect data privacy when a user accesses the sensitive cloud data. Privacy-preserving database query allows the user to retrieve a data item from the cloud database without revealing the information of the queried data item, meanwhile limiting user's ability to access other ones. In this study, in order to achieve the privacy preservation and reduce the communication complexity, a quantum-based database query scheme for privacy preservation in cloud environment is developed. Specifically, all the data items of the database are firstly encrypted by different keys for protecting server's privacy, and in order to guarantee the clients' privacy, the server is required to transmit all these encrypted data items to the client with the oblivious transfer strategy. Besides, two oracle operations, a modified Grover iteration, and a special offset encryption mechanism are combined together to ensure that the client can correctly query the desirable data item. Finally, performance evaluation is conducted to validate the correctness, privacy, and efficiency of our proposed scheme.

1. Introduction

Cloud computing is a powerful computing paradigm that enables ubiquitous access to shared infrastructure resources and higher-level services. It has shown the remarkable advantage in load balancing, data access control, and resources sharing, for database management [1]. Benefiting from the cloud paradigm, an increasing number of individuals and groups choose to put their massive data (including private part) into the cloud.

In recent years, database outsourcing has become an important component of cloud computing [2], where data owners outsource data management to a service provider (i.e., cloud database), and this mode is also called Database-as-a-Service (DaaS) [3]. Cloud database provides users with capabilities to store and process their data in the cloud, which has the advantages of scalability and high availability that users can access data anytime, anywhere and anyway. However, all the data of data owner is stored in the cloud environment, and some sensitive data (e.g., health records, financial transactions, and personal information) is at risk of being compromised. So, security and privacy have become the major challenges which inhibit the cloud computing wide acceptance in practice [4].

The privacy preservation is the main concern of cloud application, such as service recommendation [5–7], service quality prediction [8, 9], database query [10–16], etc. As an important research branch, the privacy-preserving database query (PPDQ) aims to protect database security and clients' privacy, while ensuring the correctness of database query. To be specific, any user can query data items from the cloud database without revealing its information, but his/her access to other data items is not permitted. There are a variety of techniques or methods for guaranteeing the privacy

preservation of database query, such as homomorphic encryption (HE) [10, 11], attribute-based encryption (ABE) [12–14], searchable encryption (SE) [15, 16], etc. Searchable encryption is a cryptographic system which offers secure search functions over encrypted data, which is considered to be a more effective technique to solve the problem of PPDQ. In 2000, Song et al. [15] proposed the first searchable encryption scheme based on symmetric key cryptography (SKC). Since then, other various SE schemes have been continuously proposed, such as public key cryptography-(PKC-) based searchable encryption [17], secure ranked search over encrypted cloud data [18], and so on.

As we all know, the security of classic cryptography protocols, including most private query schemes (also named privacy-preserving database query schemes), is based on mathematical complexity, and its security is based on the fact that computing power is limited. However, with the prevalence of new distributed computing models (especially cloud computing), a normal user is given the super computing power far beyond a single computer. Therefore, these cryptography protocols based on computational complexity are facing serious challenges.

On the other hand, quantum computing demonstrates the superior parallel computing power that the classical paradigm cannot match. For instance, Shor's algorithm [19] solves the problem of integer factorization in polynomial time, and Grover's algorithm [20] has a quadratic speedup to the problem of conducting a search through some unstructured database. Therefore, most classic cryptography protocols, including PPDQ schemes, are very vulnerable to the powerful quantum computer. Fortunately, quantum mechanics also provides a security mechanism against quantum attacks, and it holds the potential unconditional security based on some physical properties, such as noncloning theorem, uncertainty principle, quantum entanglement, etc. With the application of quantum mechanics in the field of information processing, some research findings have been proposed, including quantum key distribution [21, 22], quantum secret sharing [23, 24], quantum key agreement [25, 26], quantum direct communication [27, 28], quantum steganography [29], quantum teleportation and remote state preparation [30–32], quantum sealed-bid auction [33, 34], delegating quantum computation [35], and quantum machine learning [36, 37].

With the above observations, the security of classic database query schemes is facing the dual challenge of cloud computing and quantum computing, while quantum mechanics has been proven to be an effective method for solving such problem. In this study, in order to implement the privacy-preserving database query in cloud environment, we utilize some physical properties of quantum mechanics to design a quantum-based database query scheme for privacy preservation (QBDQ) in cloud environment and conduct its performance evaluation to show our scheme is feasible, secure, and efficient. To be specific, our main contributions include the three following aspects.

(1) We present a systematic framework for privacy preservation cloud database query scheme in the cloud environment.

(2) A feasible QBDQ is designed through oblivious transfer, the offset encryption mechanism, oracle operation, and the modified Grover iteration to achieve the privacy preservation for the cloud database query and reduce its communication complexity.

(3) The performance evaluation is conducted to verify the performance of our proposed QBDQ scheme, such as correctness, security, and efficiency.

The rest of this paper is organized as follows. In Section 2, we introduce the basic knowledge of quantum computing, while the framework of the privacy-preserving database query in cloud environment is presented. In Section 3, the problem of privacy-preserving database query in cloud environment is defined, and then the proposed QBDQ is elaborated step by step. Section 4 conducts the performance evaluation from the aspects of correctness, security, and efficiency. After that, Section 5 summarizes the related work on cloud database queries, SE, and quantum private queries. Finally, the conclusion of the paper and the prospection for future work are presented in Section 6.

2. Preliminaries

In this section, the basic knowledge of quantum computing is introduced firstly. Then, we introduced the principle of oblivious transfer (OT). And finally, a cloud computing framework for privacy preservation is designed.

2.1. Quantum Computing

2.1.1. Quantum Bit. The classic bit is the smallest unit in the classic computer, and its value is either 0 or 1. Unlike classical computers, the smallest unit of quantum computers is qubit (quantum bit), which is the quantum analog of the classic bit. A qubit is a unit vector in a two-dimensional complex Hilbert space, and its Dirac notation is represented as follows:

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{1}$$

where α and β are the probability amplitudes of the state $|\varphi\rangle$ and $|\alpha|^2 + |\beta|^2 = 1$. Since the vectors $|0\rangle$ and $|1\rangle$ are basis states and can be represented as

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{2}$$

the qubit $|\varphi\rangle$ can be expressed in vector form $|\varphi\rangle = \binom{\alpha}{\beta}$. In addition, the single qubit can be extended to multiple qubits; for example, an n-qubit system can exist in any superposed basis states

$$|\varphi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle + \cdots \alpha_{2^n-1}|2^n - 1\rangle. \tag{3}$$

Here, $\sum_{i=0}^{2^n-1}|\alpha_i|^2 = 1$. Quantum states $\{|0\rangle, |1\rangle, \ldots, |2^n - 1\rangle\}$ form a complete orthonormal basis in Hilbert space.

2.1.2. Unitary Operator. In a closed quantum system, the evolution of the system is characterized by a series of unitary operators; that is,

$$\left|\varphi'\right\rangle = U\left|\varphi\right\rangle, \tag{4}$$

where $UU^{\dagger} = U^{\dagger}U = I$ and $U^{\dagger}$ is the transpose conjugate of U. Each unitary operator corresponds to a quantum gate. Similar to a logic gate in classical calculations, the quantum gate can be represented in matrix form, and the quantum gate over a qubit is represented by a 2×2 unitary matrix. For instance, Pauli-X, Pauli-Z, and the Hadamard gate H are important quantum operators over one qubit described in

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{5}$$
$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

2.1.3. Quantum Measurement. The quantum state is in a superposition state, and it must be measured to collapse to a basis state to obtain a result. Assuming that the quantum state is $|\varphi\rangle = (1/\sqrt{N})\sum_{i=0}^{N-1}\alpha_i|i\rangle$ before measurement operator, quantum measurements are described by a collection $\{M_i\}$ of measurement operators which satisfy the completeness equation

$$\sum_{i=0}^{N-1} M_i^{\dagger} M_i = I, \tag{6}$$

where i indicates the possible outcome of the measurement. The quantum state is measured by the measurement basis $|i\rangle$, then the probability that result i occurs is given by

$$p(i) = \langle\varphi| M_i^{\dagger} M_i |\varphi\rangle, \tag{7}$$

and the postmeasurement state is

$$\frac{M_i|\varphi\rangle}{\sqrt{\langle\varphi| M_i^{\dagger} M_i |\varphi\rangle}}. \tag{8}$$

2.2. Oblivious Transfer. In cryptography, an oblivious transfer (OT) strategy is a type of strategy in which a sender transfers one of potentially many pieces of information to a receiver, but remains oblivious as to what piece (if any) has been transferred. The first form of oblivious transfer was introduced by Rabin [38]. In this form, the sender sends a message to the receiver with probability 1/2, while the sender remains oblivious as to whether or not the receiver received the message. OT is a basic strategy in the field of cryptography and has a wide range of applications. In general, the OT strategy involves two parties, the sender and the receiver, and satisfies the following characteristics:

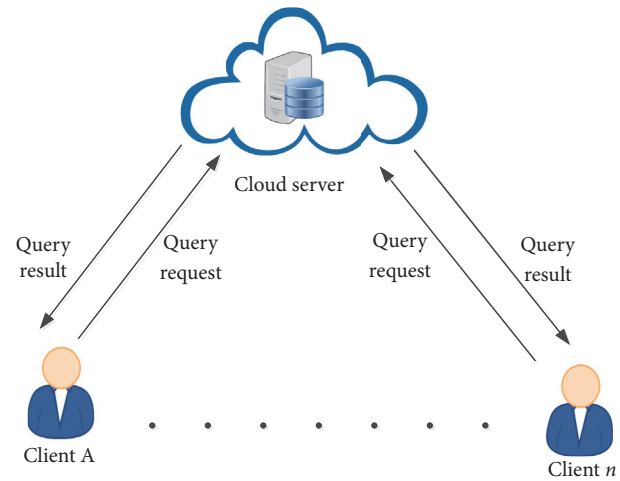


Figure 1: The framework of privacy-preserving database query in cloud environment.

(i) Whether the queried data can be obtained is entirely dependent on probability, rather than sender or receiver. That is, neither the sender nor receiver can affect the execution of the strategy.

(ii) After the execution of the strategy, the sender could not know whether the receiver got the data he wanted to query.

k-out-of-n (OT_n^k) (k<n) is the general form of all OT strategies. That is, the sender has n secrets, and the receiver can only get k secrets. The OT_n^k strategy consists of two parties, the sender with n secret data $(d_0, d_1, \ldots, d_{n-1})$, and the receiver with k indices $(\mathrm{i}_1, \mathrm{i}_2, \ldots, \mathrm{i}_k)$. The strategy meets the following requirements:

(i) *Correctness*. After executing the strategy, the receiver can obtain all of the d_{i} correctly.

(ii) *Receiver's Security*. When the receiver queries the data from the sender, the database cannot know the receiver's query items.

(iii) *Sender's Security*. The receiver cannot get more data items from the sender except queried data items.

2.3. The Framework of Privacy-Preserving Database Query in Cloud Environment. We first consider the framework model of privacy-preserving cloud database query system, which consists of two main entities (clients and cloud server) as illustrated in Figure 1.

As shown in Figure 1, there are n clients and a cloud database server, and every client sends a query request to the cloud server and gets the query result from the cloud server finally. In this framework, we suppose all the clients and server are semihonest: they are curious about cheating the privacy of other's, but honest to carry out the operations in the scheme. Here, two kinds of entity can be defined as follows.

Client is the entity that wants to query items from the database in the cloud server and can be the connected users

Table 1: Key notations and descriptions involved in proposed QBDQ scheme.

Notation	Description
N	The number of items in cloud server's database
D	The cloud server's database, $D = \{D_0, D_1, \ldots, D_{N-1}\}$
D_i	The i-th data in D
n	The number of index qubits used to encode index of data items $n = \lceil \log N \rceil$
m	The number of qubits used to encode data D_i
p	The index of client Alice's query data
D_p	The data item Alice wants to query from D
q	The index of client Bob's query data
D_q	The data item Bob wants to query from D
Δs_A	The offset value of Alice
Δs_B	The offset value of Bob
K	The encryption key sequence belongs to Charlie, $K = \{K_0, K_1, \ldots, K_{N-1}\}$.
O_K	The oracle operation to encode Charlie's key sequence K
O_D^A	The oracle operation to encode Alice's query result D_p
O_D^B	The oracle operation to encode Bob's query result D_q
O_s	The oracle operation which conditionally changes the sign in the amplitudes of the query item D_p (D_q)
O_p	The oracle operation which perform a conditional phase shift of -1 with every computational bass state except $\|0\rangle$

or the individual user with mobile constrained devices such as smartphones, PDA, TPM chip, etc.

Cloud server is the entity which provides data services and computational resources to the clients dynamically.

In this paper, we take three parties as an example, i.e., the client Alice, client Bob, and the cloud server Charlie, to demonstrate the process of the privacy-preserving database query using quantum mechanics.

3. A Quantum-Based Database Query Scheme for Privacy Preservation in Cloud Environment

In this section, we first define the privacy-preserving database query problem and quantum-based privacy-preserving database query problem in cloud environment. To address this issue, a QBDQ scheme is proposed in detail. Before we introduce the relevant content, the key notations and descriptions used in this section are listed in Table 1.

3.1. Some Definitions. In order to clearly illustrate our scheme, we first define the problem to be solved.

Definition 1 (database query problem for privacy preservation in cloud environment). In the cloud environment, the cloud server has a collection of sensitive data $D = \{D_0, D_1, \ldots, D_{N-1}\}$, and each client wants to query a data item D_i $(0 \le i \le N-1)$ from the cloud server without revealing which item is queried. During the retrieving process, the client cannot gain any other data item except D_i.

Definition 2 (database query scheme for privacy preservation in cloud environment). Each client inputs the index of query item i $(0 \le i \le N-1)$, and cloud server inputs sensitive dataset $D = \{D_0, D_1, \ldots, D_{N-1}\}$. After executing this scheme, the client outputs the queried data item D_i. In addition, the scheme should satisfy the following:

(i) *Correctness*. The client successfully obtains the correct data item he(she) wants to query (i.e., D_i).

(ii) *Clients' Privacy*. During the retrieving process, the cloud server cannot get any private information about the query index of the client.

(iii) *Cloud Server's Privacy*. Clients cannot get any other data items from the cloud server except D_i.

3.2. A Quantum-Based Database Query Scheme for Privacy Preservation in Cloud Environment. For the sake of simplicity, we take three parties (one cloud server Charlie, and two clients Alice, Bob) as an example to describe our scheme. Suppose Charile has a private database D with N items $\{D_0, D_1, \ldots, D_{N-1}\}$ and an encryption key sequence $K = \{K_0, K_1, \ldots, K_{N-1}\}$, and Alice and Bob want to, respectively, query an item, the p-th item D_p, and the q-th item D_q $(0 \le p, q \le N-1)$, from server. The scheme consists of five steps as follows (also shown in Figure 2).

Step 1. Charlie prepares an $(n+m)$-qubit state $|\phi_K\rangle = (1/\sqrt{N}) \sum_{i=0}^{N-1} |i\rangle \otimes |0\rangle^{\otimes m}$, where $n = \lceil \log N \rceil$, $m = \lceil \log(\max\{k_i \mid 0 \le i \le N-1\} + 1) \rceil$. And then he applies an oracle operation O_K (its schematic circuit is sketched in Figure 3) on $|\phi_K\rangle$ referring to the sequence $K = \{K_0, K_1, \ldots, K_{N-1}\}$. Here, O_K is defined as follows:

$$O_K : \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \otimes |0\rangle^{\otimes m} \longrightarrow \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \otimes |K_i\rangle , \quad (9)$$

where $|i\rangle$ denotes the index of the data item and $|K_i\rangle$ is the encryption key originally assigned to encrypt the i-th data item. After the above operation, we can get the state, namely,

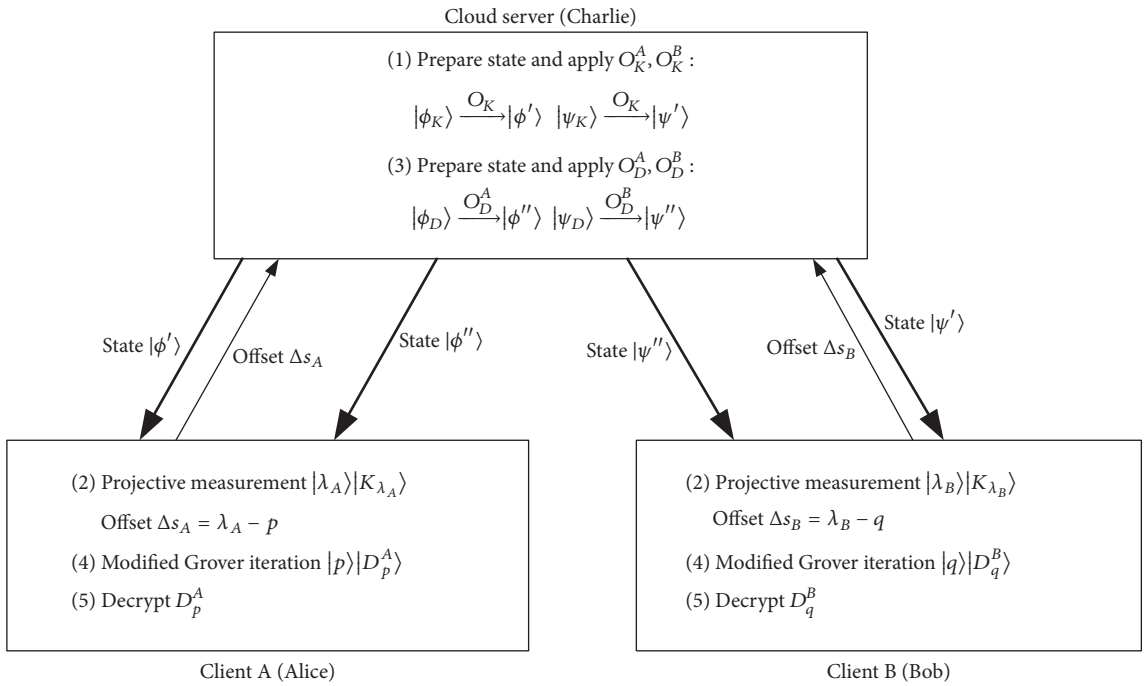


Figure 2: The five-step procedures of the QBDQ scheme among two clients and cloud server. The thick (thin) line represents quantum (classic) channel.

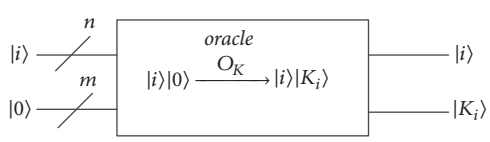


Figure 3: Schematic circuit of the oracle operation O_K.

$|\phi'\rangle = (1/\sqrt{N})\sum_{i=0}^{N-1}|i\rangle \otimes |K_i\rangle$, and then Charlie sends it to Alice with oblivious transfer strategy. Similar to Alice, Charlie also prepares another state $|\psi'\rangle = (1/\sqrt{N})\sum_{i=0}^{N-1}|i\rangle \otimes |K_i\rangle$ in the same way and sends it to Bob.

Step 2. After receiving $|\phi'\rangle$ from Charlie, Alice takes $\{|0\rangle, |1\rangle, \ldots, |N-1\rangle\}$ as the computational basis and performs projective measurement on the index qubits of $|\phi'\rangle$. Suppose the measurement result is λ_A ($\lambda_A \in \{0, 1, \ldots, N-1\}$); the remaining m qubits will collapse into $|K_{\lambda_A}\rangle$, which means Alice can obtain K_{λ_A} (i.e., one of the encryption keys) through projective measurement. Since Alice's retrieving index is p, she computes the offset $\Delta s_{\boldsymbol{A}} = (\lambda_A - p)$ and sends it to Charlie. As same as Alice, Bob also performs the same operations and announces the offset $\Delta s_{\boldsymbol{B}} = (\lambda_B - q)$ to Charlie, where λ_B is the measurement result, and q represents the index of the data item Bob wants to query.

Step 3. Having received the offsets Δs_A and Δs_B, Charlie updates every encryption key as

$$\begin{aligned} K_i^A &= K_{(i+\Delta s_A)\mathrm{mod}N} \\ K_i^B &= K_{(i+\Delta s_B)\mathrm{mod}N} \end{aligned} \tag{10}$$

and obtains the new key sequence K^A and K^B,

$$\begin{aligned} K^A &= \left\{K_i^A \mid 0 \le i \le N-1\right\} \\ K^B &= \left\{K_i^B \mid 0 \le i \le N-1\right\}. \end{aligned} \tag{11}$$

Then, Charlie encrypts every data items respectively with its new corresponding keys K_i^A and K_i^B as follows:

$$\begin{aligned} D_i^A &= D_i \oplus K_i^A, \quad 0 \le i \le N-1 \\ D_i^B &= D_i \oplus K_i^B, \quad 0 \le i \le N-1. \end{aligned} \tag{12}$$

After that, Charlie prepares two states $|\phi_D\rangle = (1/\sqrt{N})\sum_{i=0}^{N-1}|i\rangle \otimes |0\rangle^{\otimes m}$, $|\psi_D\rangle = (1/\sqrt{N})\sum_{i=0}^{N-1}|i\rangle \otimes |0\rangle^{\otimes m}$ and applies the oracle operation O_D^A, O_D^B as

$$\begin{aligned} O_D^A &: \frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}|i\rangle\,|0\rangle^{\otimes m} \longrightarrow \frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}|i\rangle\left|D_i^A\right\rangle \\ O_D^B &: \frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}|i\rangle\,|0\rangle^{\otimes m} \longrightarrow \frac{1}{\sqrt{N}}\sum_{i=0}^{N-1}|i\rangle\left|D_i^B\right\rangle \end{aligned} \tag{13}$$

and gets the final states $|\phi''\rangle = (1/\sqrt{N})\sum_{i=0}^{N-1}|i\rangle|D_i^A\rangle$, $|\psi''\rangle = (1/\sqrt{N})\sum_{i=0}^{N-1}|i\rangle|D_i^B\rangle$. Finally, Charlie sends $|\phi''\rangle$, $|\psi''\rangle$ to Alice and Bob, respectively, with oblivious transfer strategy.

Step 4. After receiving $|\phi''\rangle$ from Charlie, Alice performs the modified Grover iteration on it to obtain the target state $|p\rangle|D_p^A\rangle$. Figure 4 describes the detailed process of modified Grover iteration, which consists of at most $\lceil(\pi/4)\sqrt{2^{n+m}}\rceil$

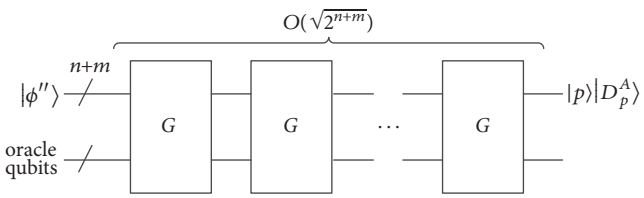


FIGURE 4: Schematic circuit of the modified Grover iteration applied on state $|\phi''\rangle$, where G is the quantum subroutine illustrated in Figure 5.

times application of a quantum subroutine, called the G operator. The whole process of G operator (also shown in Figure 5) can be subdivided into four steps as follows.

Step 4.1. Alice applies the oracle operation O_s on $|\phi''\rangle$, which conditionally changes the sign of the amplitudes of the query item

$$O_s : \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \left|D_i^A\right\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} (-1)^{f(i)} |i\rangle \left|D_i^A\right\rangle \quad (14)$$

Here, we call the resultant state $O_s|\phi''\rangle$, i.e., $O_s|\phi''\rangle = (1/\sqrt{N}) \sum_{i=0}^{N-1} (-1)^{f(i)}|i\rangle|D_i^A\rangle$, and $f(i)$ is the judgment function defined by

$$f(i) = \begin{cases} 1, & \textit{if } i \textit{ is the query address } (\textit{i.e.}, i = p) \\ 0, & \textit{else } (\textit{i.e.}, i \neq p) \end{cases} \quad (15)$$

Step 4.2. The Hadamard transformation $H^{\otimes(n+m)}$ is applied on $O_s|\phi''\rangle$,

$$\begin{aligned} &\frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} (-1)^{f(i)} |i\rangle \left|D_i^A\right\rangle \\ &\quad \longrightarrow H^{\otimes(n+m)} \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} (-1)^{f(i)} |i\rangle \left|D_i^A\right\rangle . \end{aligned} \quad (16)$$

Step 4.3. Alice applies conditionally phase transfer O_p on the state $H^{\otimes(n+m)}O_s|\phi''\rangle$,

$$\begin{aligned} O_p : &H^{\otimes(n+m)} \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} (-1)^{f(i)} |i\rangle \left|D_i^A\right\rangle \\ &\longrightarrow -(-1)^{\sigma_{i,Di}} H^{\otimes(n+m)} \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} (-1)^{f(i)} |i\rangle \left|D_i^A\right\rangle , \end{aligned} \quad (17)$$

where the function σ_{i,D_i} is defined as follows:

$$\sigma_{i,D_i} = \begin{cases} 1, & i = 0, D_i = 0, \\ 0, & \textit{else}. \end{cases} \quad (18)$$

Step 4.4. The Hadamard transformation $H^{\otimes(n+m)}$ is applied again on $O_p H^{\otimes(n+m)} O_s|\phi''\rangle$ and obtains the state

$$\left|\xi\right\rangle = H^{\otimes(n+m)} O_P H^{\otimes(n+m)} O_s \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} |i\rangle \left|D_i^A\right\rangle \quad (19)$$

Alice applies the above Grover iteration $\lceil(\pi/4)\sqrt{2^{n+m}}\rceil$ times and finally obtains the target state $|p\rangle|D_p^A\rangle$.

Similar to Alice, Bob also applies the modified Grover iteration on the received state $|\psi''\rangle = (1/\sqrt{N}) \sum_{i=0}^{N-1} |i\rangle|D_i^A\rangle$ and obtains the target query state $|q\rangle|D_q^B\rangle$.

Step 5. Alice and Bob measure the last m-qubit of state $|p\rangle|D_p^A\rangle$, $|q\rangle|D_q^B\rangle$ and extract the classic information of query result D_p^A, D_q^B, respectively.

In addition, in order to check eavesdropping in the quantum channel, we can use decoy-photon technology. That is, the sender randomly inserts several decoy photons into the qubit sequence, where every decoy photon is prepared randomly with either Z-basis $\{|0\rangle, |1\rangle\}$ or X-basis $\{(1/\sqrt{2})(|0\rangle + |1\rangle), (1/\sqrt{2})(|0\rangle - |1\rangle)\}$, and transmits them to the receiver. After confirming that the receiver has received the transmitted sequence, the sender announces the positions of the decoy photons and the corresponding measurement basis. The receiver measures the decoy photons according to the sender's announcements and tells the sender his (her) measurement results. Then, the sender compares the measurement results from the receiver with the initial states of the decoy photons in the transmitted sequence and calculates the error rate. If the error rate is higher than the threshold determined by the channel noise, they cancel this scheme and restart; else they continue the next step.

It is worth mentioning that we adopted the OT strategy and offset encryption mechanism in our scheme. In Step 3, the OT strategy is utilized to transfer Charlie's data to Alice and Bob. As we know, the transmitted state $|\psi\rangle$ is a superposition state which encapsulates all the encrypted data items $\{D_i \mid 0 \leq i \leq N-1\}$. So, the process of Charlie sending $|\phi'\rangle$, $|\psi'\rangle$ to Alice and Bob can be viewed as the oblivious transfer mechanism. The use of OT strategy ensures that information about Charlie cannot be leaked. In addition, our scheme also applied the offset encryption mechanism. The offsets Δs_A, Δs_B can be computed by using the index of the query data items and the keys determined by clients' measurement. Charlie updates the encryption keys according to these offsets and then encrypts data with these updated keys, respectively. The combination of OT strategy and offset mechanism allows Alice and Bob obtain the correct data they want to query, while Charlie cannot get their queried data, which guaranteed the privacy of client. At the same time, data encryption makes the data items into ciphertext, and neither the eavesdropper nor the clients can directly obtain the data item, thus ensuring the data security of the cloud server.

4. Performance Evaluation

Our proposed QBDQ scheme in cloud environment tends to ensure the correctness of query result, protect the privacy of clients and servers in cloud, and also improve the efficiency during querying the cloud database. Therefore, we take three

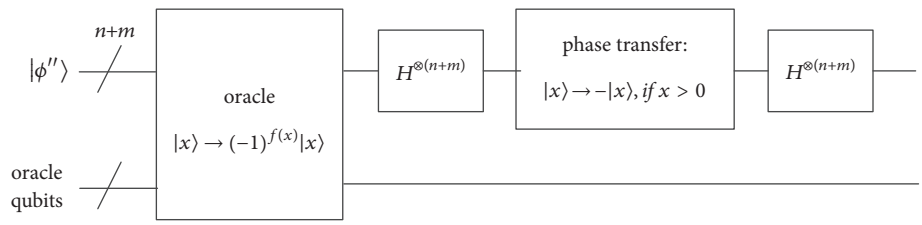


Figure 5: Schematic circuit of the G operator.

parties (i.e., clients Alice and Bob; cloud server Charlie) as an example and estimate the overall performance of the proposed scheme in terms of correctness analysis, security analysis, and the efficiency analysis.

4.1. Correctness Analysis. Now, we analyze the correctness of the proposed scheme. Without loss of generality, suppose that the server Charlie has a database of 16 items $D = \{5, 9, 6, 12, 2, 11, 11, 6, 5, 10, 7, 15, 6, 11, 6, 9\}$, and he holds the corresponding encryption key sequence $K = \{14, 8, 3, 4, 7, 1, 11, 6, 15, 2, 12, 13, 0, 5, 9, 10\}$. Since $N = 16$, the max value in K is 15, $n = \lceil \log N \rceil = \lceil \log 16 \rceil = 4$, $m = \lceil \log(15 + 1) \rceil = 4$. Here, we take Alice as an example to analyze the procedures of our QBDQ scheme as follows (suppose Alice wants to query the 9th item of the database).

In Step 1, Charlie prepares an initial state $|\phi_K\rangle = (1/4)\sum_{i=0}^{15} |i\rangle \otimes |0000\rangle$ and performs an oracle operation O_K on it to encode his encryption keys,

$$\begin{aligned}
|\phi'\rangle = {} & \frac{1}{4}(|0\rangle |K_0\rangle + |1\rangle |K_1\rangle + |2\rangle |K_2\rangle \\
& + |3\rangle |K_3\rangle + |4\rangle |K_4\rangle + |5\rangle |K_5\rangle + |6\rangle |K_6\rangle \\
& + |7\rangle |K_7\rangle + |8\rangle |K_8\rangle + |9\rangle |K_9\rangle + |10\rangle |K_{10}\rangle \\
& + |11\rangle |K_{11}\rangle + |12\rangle |K_{12}\rangle + |13\rangle |K_{13}\rangle \\
& + |14\rangle |K_{10}\rangle + |15\rangle |K_{11}\rangle) = \frac{1}{4}(|0\rangle |14\rangle \\
& + |1\rangle |8\rangle + |2\rangle |3\rangle + |3\rangle |4\rangle + |4\rangle |7\rangle \\
& + |5\rangle |1\rangle + |6\rangle |11\rangle + |7\rangle |6\rangle + |8\rangle |15\rangle \\
& + |9\rangle |2\rangle + |10\rangle |12\rangle + |11\rangle |13\rangle + |12\rangle |0\rangle \\
& + |13\rangle |5\rangle + |14\rangle |9\rangle + |15\rangle |10\rangle)
\end{aligned} \tag{20}$$

Then, he sends the resultant state $|\phi'\rangle$ to Alice. In Step 2, Alice performs projective measurement on the first four qubits (i.e., index qubits) of $|\phi'\rangle$ in the computational basis $\{|0000\rangle, |0001\rangle, \ldots, |1111\rangle\}$. Suppose the random measurement result is $|12\rangle$ (i.e., $\lambda_A = 12$), then the remaining qubits (i.e., the key qubits) collapse to the state $|K_{\lambda_A}\rangle = |0000\rangle$, which means $K_{\lambda_A} = 0000$. But the data Alice wants to query is the ninth data D_8, so she computes the difference between λ_A and the desirable query index q, $\Delta s = (\lambda_A - q) = 4$, and sends Δs to Charlie. After receiving Δs, Charlie updates the key sequence K through the formulation $K_i^A = K_{(i+\Delta s)\text{mod}N}$, then $K_i^A = \{7, 1, 11, 6, 15, 2, 12, 13, 0, 5, 9, 10, 14, 8, 3, 4\}$. He uses K_i^A to encrypt every data items: $D_i^A = D_i \oplus K_i^A$, that is, $\{2, 8, 13, 10, 13, 9, 7, 11, 5, 15, 14, 5, 8, 3, 5, 13\}$. Then, in Step 3, Charlie prepares another state $|\phi_D\rangle = (1/4)\sum_{i=0}^{15} |i\rangle \otimes |0000\rangle$ and applies the oracle operation O_D^A to embed the encrypted data items D_i^A,

$$\begin{aligned}
|\phi''\rangle = {} & \frac{1}{4}(|0\rangle |D_0\rangle + |1\rangle |D_1\rangle + |2\rangle |D_2\rangle \\
& + |3\rangle |D_3\rangle + |4\rangle |D_4\rangle + |5\rangle |D_5\rangle + |6\rangle |D_6\rangle \\
& + |7\rangle |D_7\rangle + |8\rangle |D_8\rangle + |9\rangle |D_9\rangle + |10\rangle |D_{10}\rangle \\
& + |11\rangle |D_{11}\rangle + |12\rangle |D_{12}\rangle + |13\rangle |D_{13}\rangle \\
& + |14\rangle |D_{14}\rangle + |15\rangle |D_{15}\rangle) = \frac{1}{4}(|0\rangle |2\rangle \\
& + |1\rangle |8\rangle + |2\rangle |13\rangle + |3\rangle |10\rangle + |4\rangle |13\rangle \\
& + |5\rangle |9\rangle + |6\rangle |7\rangle + |7\rangle |11\rangle + |8\rangle |5\rangle \\
& + |9\rangle |15\rangle + |10\rangle |14\rangle + |11\rangle |5\rangle + |12\rangle |8\rangle \\
& + |13\rangle |3\rangle + |14\rangle |5\rangle + |15\rangle |13\rangle).
\end{aligned} \tag{21}$$

Then, he sends the state $|\phi''\rangle$ to Alice.

Further, Alice performs modified Grover iteration on $|\phi''\rangle$ up to $R = \lceil (\pi/4)\sqrt{2^{n+m}} \rceil = \lceil (\pi/4)\sqrt{2^8} \rceil = 13$ times (actually, the number of iterations is 6), then she can obtain the encrypted query item $|p\rangle|D_p^A\rangle = |8\rangle|5\rangle$ with a high possibility, and measures it to get $D_p^A = 5$. Alice uses the obtained key $K_8^A = 0$ to decrypt the ninth item

$$D_8 = D_8^A \oplus K_8^A = 0101 \oplus 0000 = 0101. \tag{22}$$

Therefore, regardless of what measurement result Alice has obtained, she can finally obtain the query data correctly.

Figure 6 shows the entire execution process of Alice querying Charlie's database in a simplified way. At the same time, it also sketched the execution of the other user Bob (assuming it queries the fifth data).

4.2. Security Analysis

4.2.1. Privacy Analysis. Cloud Server's Privacy. Suppose the client Alice is dishonest, and she wants to obtain more information about Charlie's database. In Step 1 of our

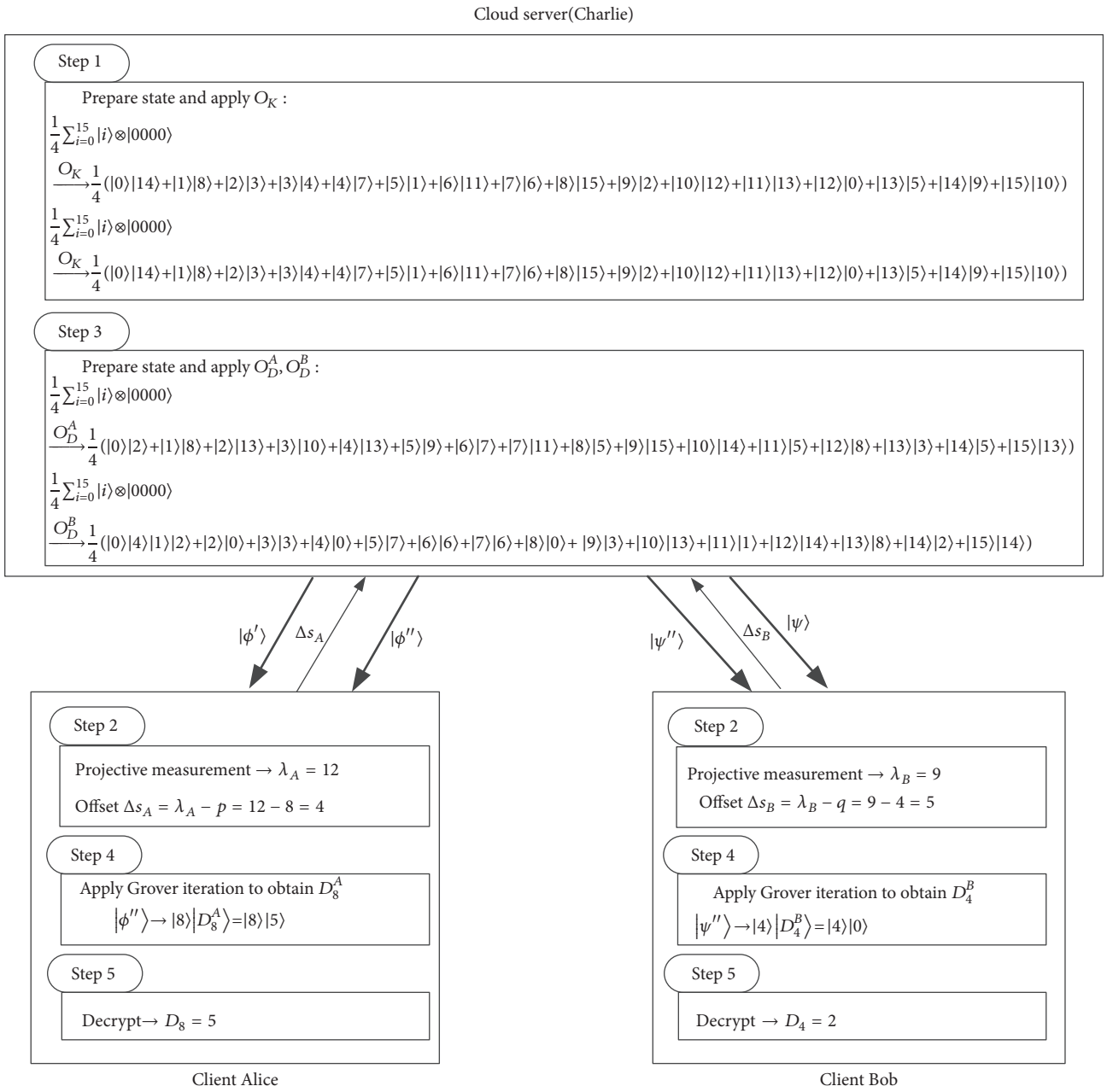


Figure 6: The schematic graph of the execution process of Alice and Bob in our QBDQ scheme, assuring they query the *9-th* and *5-th* items, respectively.

scheme, the server Charlie sends the quantum state $|\phi'\rangle = (1/\sqrt{N})\sum_{i=0}^{N-1}|i\rangle|K_i\rangle$ to client Alice through oblivious transfer strategy. Since all the information about the key sequence K is encoded in the state $|\phi'\rangle$, so Alice cannot extract the key form $|\phi'\rangle$ directly. Here we suppose the whole system of quantum state $|\phi'\rangle$ consisted of two subsystems, i.e., the n-qubit quantum subsystem C (index qubits $|i\rangle$) and the m-qubit subsystem D (key qubits $|K_i\rangle$). If Alice makes a projective measurement on the received state $|\phi'\rangle$, she will get the resultant state $|i\rangle|K_i\rangle$ for any i with the probability of $1/N$. The whole system can be represented by the quantum ensemble $\varepsilon = \{p_i, \rho(i)\}$, here $p_i = 1/N$,

$$\rho(i) = |i\rangle\,|K_i\rangle\,\langle K_i|\,\langle i|\,. \tag{23}$$

Here we get the upper limit of information that Alice can get from Charlie's is determined by the Holevo bound [38],

$$H(A:B) \le S(\rho) - \frac{1}{N}\sum_{i=0}^{N-1} S(\rho(i)) \tag{24}$$

Here $S(\rho)$ denotes Von Neumann entropy of quantum state ρ, $H(B{:}A)$ means the information Alice can get about Charlie's key information (including the address i and according keys K_i), and we have

$$S(\rho) = S\left(\frac{1}{N}\sum_{i=0}^{N-1} |i\rangle \left|K_i\right\rangle \left\langle K_i\right| \langle i|\right) = n + m. \quad (25)$$

and $S(\rho(i)) = S(|i\rangle|K_i\rangle\langle K_i|\langle i|) = 0$; therefore,

$$H(A:B) \le n + m. \quad (26)$$

Then, Alice can only get n-bit of address information (i.e., i) and the corresponding m-bit key (i.e., K_i) by measuring ρ. In addition, she will certainly lose the change to get her key K_i. This means Alice cannot extract more than one key from Charlie.

Besides, in Step 3, Charlie uses the offset key $K_i^A = K_{i+\Delta s}$ to encrypt the data items, and send its encoded state $|\phi''\rangle = (1/\sqrt{\Delta s})\sum_{i=0}^{N-1}|i\rangle|D_i^A\rangle$ to Alice with oblivious transfer strategy. Alice's privacy of query index i is protected by the oblivious transfer strategy. For example, the transmitted state $|\phi''\rangle$ Alice received is a superposition state, i.e., $|\phi''\rangle = (1/\sqrt{N})(|0\rangle|D_0\rangle + |1\rangle|D_1\rangle + \cdots + |N-1\rangle|D_{N-1}\rangle)$, which encapsulates all the query data $\{D_p^A \mid 0 \le p \le N-1\}$ including the desirable one D_p^A. Alice obtains the query item D_p^A through the Grover iteration and the previously obtained key K_{λ_A}. Suppose Bob is also dishonest, he has the same situation with Alice.

Client's Privacy. If Charlie is dishonest, he may try to obtain Alice's private query index p during the communication process. However, Alice only sends one classic message $\Delta s = \lambda_A - p$ to cloud server Charlie in Step 2, and Charlie does not know the encryption key which chosen by Alice, thus he cannot obtain any useful information about the data Alice wants to search. As same as Alice, Bob only sends a classic offset message $\Delta s = \lambda_B - p$ to Charlie, which prevents Charlie from obtaining his information.

4.2.2. Channel Security Analysis. The security of the quantum channel is guaranteed by the decoy-photon checking technology. The process of eavesdropping detected done by the two neighbor participants in our scheme is essentially equivalent to that in the BB84 scheme [36], which has been proved to be unconditionally secure. To be specific, the decoy qubits, which are randomly inserted into target qubits, are generated by randomly chosen from $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$. After one participant sends the mixed decoy qubits and encrypted target qubits to quantum center, he will ask quantum center to measure them with the same bases these qubits were produced. For any outside eavesdropper, the bases used by participants are all random; the eavesdropper cannot produce the same qubits like decoy qubits before quantum center receives the qubits. Just like the situation in the BB84 scheme, if any outside eavesdropper exists in the process of our scheme, the eavesdropping actions will be found by the two participants.

The outside eavesdropper cannot get the shared key because eavesdropper cannot distinguish target qubits form decoy qubits, and he can only choose one set of orthogonal basis to measure it, so the eavesdropper will certainly change the states of the qubit, and then he will be discovered. We assume that eavesdropper will do intercept-resend attack. Eavesdropper applies operation U_E and auxiliary system $|E\rangle$ which satisfies the following conditions:

$$U_E|0\rangle|E\rangle = a|0\rangle\left|E_{00}\right\rangle + b|1\rangle\left|E_{01}\right\rangle, \quad (27)$$

$$U_E|1\rangle|E\rangle = c|0\rangle\left|E_{01}\right\rangle + d|1\rangle\left|E_{11}\right\rangle. \quad (28)$$

Here, $|a|^2 + |b|^2 = 1$ and $|c|^2 + |d|^2 = 1$. If the eavesdropper wants to extract the encode information precisely, then U_E must satisfy

$$\begin{aligned} U_E|+\rangle|E\rangle &= \frac{1}{\sqrt{2}}\left(a|0\rangle\left|E_{00}\right\rangle + b|1\rangle\left|E_{01}\right\rangle + c|0\rangle\right. \\ &\quad \left.\cdot\left|E_{10}\right\rangle + d|1\rangle\left|E_{11}\right\rangle\right) = \frac{1}{2}\left(|+\rangle\right. \\ &\quad \left.\cdot\left(a\left|E_{00}\right\rangle + b\left|E_{01}\right\rangle + c\left|E_{10}\right\rangle + d\left|E_{11}\right\rangle\right)\right), \end{aligned} \quad (29)$$

$$\begin{aligned} U_E|-\rangle|E\rangle &= \frac{1}{\sqrt{2}}\left(a|0\rangle\left|E_{00}\right\rangle + b|1\rangle\left|E_{01}\right\rangle - c|0\rangle\right. \\ &\quad \left.\cdot\left|E_{10}\right\rangle - d|1\rangle\left|E_{11}\right\rangle\right) = \frac{1}{2}\left(|-\rangle\right. \\ &\quad \left.\cdot\left(a\left|E_{00}\right\rangle - b\left|E_{01}\right\rangle - c\left|E_{10}\right\rangle + d\left|E_{11}\right\rangle\right)\right), \end{aligned} \quad (30)$$

$$\begin{aligned} U_E|+y\rangle|E\rangle &= \frac{1}{\sqrt{2}}\left(a|0\rangle\left|E_{00}\right\rangle + b|1\rangle\left|E_{01}\right\rangle\right. \\ &\quad \left.+ ic|0\rangle\left|E_{10}\right\rangle + id|1\rangle\left|E_{11}\right\rangle\right) = \frac{1}{2}\left(|+y\rangle\right. \\ &\quad \left.\cdot\left(a\left|E_{00}\right\rangle - ib\left|E_{01}\right\rangle + ic\left|E_{10}\right\rangle - d\left|E_{11}\right\rangle\right)\right), \end{aligned} \quad (31)$$

$$\begin{aligned} U_E|-y\rangle|E\rangle &= \frac{1}{\sqrt{2}}\left(a|0\rangle\left|E_{00}\right\rangle + b|1\rangle\left|E_{01}\right\rangle\right. \\ &\quad \left.- ic|0\rangle\left|E_{10}\right\rangle - id|1\rangle\left|E_{11}\right\rangle\right) = \frac{1}{2}\left(|+y\rangle\right. \\ &\quad \left.\cdot\left(a\left|E_{00}\right\rangle + ib\left|E_{01}\right\rangle - ic\left|E_{10}\right\rangle + d\left|E_{11}\right\rangle\right)\right). \end{aligned} \quad (32)$$

From (29)–(32) we can obtain that

$$a\left|E_{00}\right\rangle - b\left|E_{01}\right\rangle + c\left|E_{10}\right\rangle - d\left|E_{11}\right\rangle = 0, \quad (33)$$

$$a\left|E_{00}\right\rangle + b\left|E_{01}\right\rangle - c\left|E_{10}\right\rangle - d\left|E_{11}\right\rangle = 0, \quad (34)$$

$$a\left|E_{00}\right\rangle + ib\left|E_{01}\right\rangle + ic\left|E_{10}\right\rangle - d\left|E_{11}\right\rangle = 0, \quad (35)$$

$$a\left|E_{00}\right\rangle - ib\left|E_{01}\right\rangle - ic\left|E_{10}\right\rangle - d\left|E_{11}\right\rangle = 0, \quad (36)$$

we can get that a = d = 1, b = c = 0, and $|E_{00}\rangle = |E_{11}\rangle$, then we get

$$U_E|0\rangle|E\rangle = |0\rangle\left|E_{00}\right\rangle, \quad (37)$$

$$U_E|1\rangle|E\rangle = |1\rangle\left|E_{11}\right\rangle, \quad (38)$$

and we can summarize that eavesdropper would not be found only when decoy qubits and target qubits are $\{|0\rangle, |1\rangle\}$, which is impossible. So there is no way for the eavesdropper to know the secret key.

4.3. Efficiency Analysis. As we know, quantum-based schemes have greater information capacity than classic ones. In order to evaluate the efficiency of our QDBQ scheme more objectively, we choose some of the most representative quantum schemes as comparison objects, for example, Jakobi et al.'s quantum private query (QPQ) scheme (J11 for short) [39], Gao et al.'s QPQ scheme (G12) [40], and Rao et al.'s QPQ) scheme (R13) [41].

To evaluate the efficiency of quantum communication schemes, there are mainly two indicators: the communication complexity (i.e., the number of transmitted qubits), and the consumption of exchanged classic messages (i.e., the number of exchanged classic bits).

4.3.1. Communication Complexity. The communication complexity, i.e., the number of quantum bits (qubits) transmitted in the communication process, is one of the key indicators of the efficiency for communication scheme. In J11 and G12 schemes, the cloud server (Charlie) sends $k \times N$ qubits to the client (Alice), where k is the number of divided substrings. These k substrings are added bitwise in order to reduce Alice's information on the key to roughly one bit (i.e., $\overline{n} = N(1/4)^k \approx 1$), so $k = \log \sqrt{N}$. In summary, $N \times \log \sqrt{N}$ qubits are transmitted in J11 and J12 schemes, and its communication complexity is $O(N \log N)$. But in the R13 scheme, the number of qubits that need to be exchanged is reduced to $O(N)$, so the communication complexity is $O(N)$.

In our QBDQ scheme, Charlie firstly transmits a $(\lceil \log N \rceil + m)$-qubit state $|\phi'\rangle$ $(|\psi'\rangle)$ for sending the encryption keys in Step 1, and the $(\lceil \log N \rceil + m)$-qubit state $|\phi''\rangle$ $(|\psi''\rangle)$ containing every encrypted data $D_i (0 \leq i \leq N-1)$ is transmitted to Alice(Bob) in Step 3. Considering that each data item the cloud server holds is an only one-bit message in J11, G12 and R13 schemes, here we let $m = 1$. Therefore, the transmitted qubits are $2(\log N + 1)$, so its communication complexity is $O(\log N)$.

To be more intuitive, we calculate the numbers of transmitted qubits in different database capacities for the J11, G12, R13, and our QBDQ schemes (see Table 2) and show the comparison results among them in Figure 7. As shown in this figure, J11 and G12 schemes have the same qubits consumption, R13 scheme reduces the consumption, and our QBDQ scheme has the lowest qubits consumption. That is, our scheme has the lowest communication complexity among them.

4.3.2. Consumption of Exchanged Classic Messages. For a communication scheme, it should also consider the consumption of the exchanged classic messages. In the J11, J12, and R13 schemes, $N \times 1$ bits of encrypted data, considering each data item is a one-bit message (i.e., $m = 1$), is transmitted from the cloud server to the client, so their exchanged classic messages are all $O(N)$ cbits. In our scheme, Alice (Bob) returns a classical message Δs, i.e., a $(\lceil \log N \rceil \times m)$-cbit classic message, to Charlie in Step 2. Since $m = 1$, the exchanged message is just $O(\log N)$ cbits.

Table 3 lists the numbers of transmitted qubits in different database capacities for the J11, G12, R13, and our QBDQ schemes, while Figure 8 gives a more intuitive comparison between our QBDQ scheme and the other QPQ schemes (J11, G12, and R13 schemes). Obviously, our scheme needs less consumption of exchanged classic messages than other QPQ protocols.

In summary, Table 4 lists the comparison among our QBDQ scheme and the other three QPQ schemes clearly. As shown in Table 4, our scheme achieves a great reduction on both the communication complexity and the consumption of exchanged classic messages. Besides, our QBDQ scheme just needs to perform quantum measurement two times, which is obviously less than the other ones.

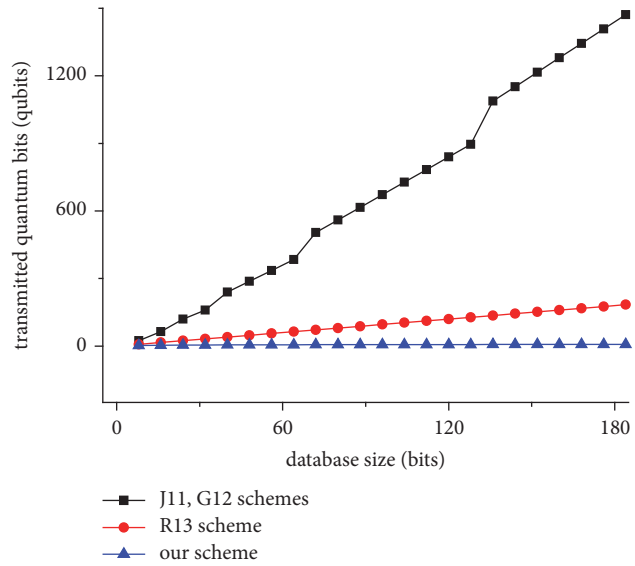


Figure 7: Comparison of transmitted qubits among our QBDQ scheme, J11, G12, and R13 schemes.

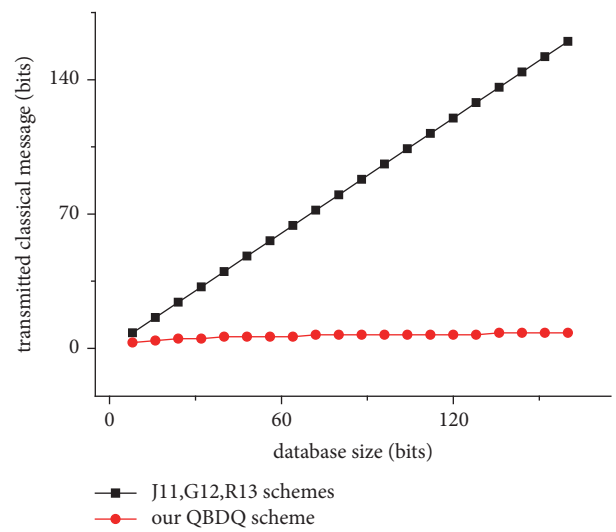


Figure 8: Comparison of exchanged classic bits between our QBDQ scheme and the other QPQ schemes (J11, G12, and R13 schemes).

Table 2: Numbers of transmitted qubits in different database capacities for J11, G12, R13, and our QBDQ schemes.

Database size	Transmitted messages(qubit)			Database size	Transmitted messages(qubit)		
	J11/ G12	R13	QBDQ		J11/G12	R13	QBDQ
8	24	8	3	208	1664	208	8
16	64	16	4	216	1728	216	8
24	120	24	5	224	1792	224	8
32	160	32	5	232	1856	232	8
40	240	40	6	240	1920	240	8
48	288	48	6	248	1984	248	8
56	336	56	6	256	2048	256	8
64	384	64	6	264	2376	264	9
72	504	72	7	272	2448	272	9
80	560	80	7	280	2520	280	9
88	616	88	7	288	2592	288	9
96	672	96	7	296	2664	296	9
104	728	104	7	304	2736	304	9
112	784	112	7	312	2808	312	9
120	840	120	7	320	2880	320	9
128	896	128	7	328	2952	328	9
136	1088	136	8	336	3024	336	9
144	1152	144	8	344	3096	344	9
152	1216	152	8	352	3168	352	9
160	1280	160	8	360	3240	360	9
168	1344	168	8	368	3312	368	9
176	1408	176	8	376	3384	376	9
184	1472	184	8	384	3456	384	9
192	1536	192	8	392	3528	392	9
200	1600	200	8	400	3600	400	9

Table 3: Exchanged classic messages in different database capacities for J11, G12, R13, and our QBDQ schemes.

Database size	Exchanged messages		Database size	Exchanged messages	
	J11/G12/R13	QBDQ		J11/G12/R13	QBDQ
8	8	3	88	88	7
16	16	4	96	96	7
24	24	5	104	104	7
32	32	5	112	112	7
40	40	6	120	120	7
48	48	6	128	128	7
56	56	6	136	136	8
64	64	6	144	144	8
72	72	7	152	152	8
80	80	7	160	160	8

Table 4: Comparison among our QBDQ scheme and the other QPQ schemes.

Schemes	Communication complexity (qubit)	Exchanged message (bit)	Measurement times
J11	$O(N \log N)$	N+1	kN
G12	$O(N \log N)$	N+1	kN
R13	$O(N)$	N+1	N
Ours	$O(\log N)$	1	2

5. Related Work

Cloud database services are typically run on cloud computing platforms, and access to cloud databases is provided as a service, which takes care of scalability and availability of the database, and it makes the underlying software-stack transparent to the user.

Benefit from cloud computing technologies and devices, more and more data owners are motivated to outsource their data to cloud servers for great convenience in data management, and cloud database query has attracted the attention of scholars. Cloud database query was firstly proposed by Chor et al. [42], where the privacy of the server cannot be guaranteed, which means that sensitive data (e.g., health records, financial transactions) stored in cloud database is threatened by information leaks. Therefore, how to preserve the privacy of sensitive data in the process of cloud database query has become an important topic. In order to solve the problem, many methods are proposed to guarantee the privacy preservation of database query [12–18]; one of the most popular methods is SE.

SE is a special kind of private query, which enables the user to store the encrypted data to the cloud and execute keyword search over ciphertext. Since Song et al.[15] proposed the first practical private database query scheme for searching on encrypted data in cloud and provided the security proofs for the scheme, some other schemes to address privacy protection issues in cloud database queries have also been proposed[17, 18]. In order to support more complex queries, the conjunctive keyword search scheme [14] over encrypted data has been proposed. After that, a more general approach, predicate encryption [16], which supports inner-product, was also proposed.

In general, most of the above schemes [12–18, 42] are based on public key cryptography such as RSA, and its security is based on mathematical NP-hard problems. Therefore, these schemes are difficult to crack in polynomial time for classic computers. All of the above protocols are based on public key cryptography such as RSA. On a quantum computer, to factor an integer N, Shor's algorithm [19] can run in polynomial time (the time taken is polynomial in $\log N$, which is the size of the input. Specifically, it takes quantum gates of order $\mathrm{O}((\log N)^2(\log\log N)(\log\log\log N))$ using fast multiplication [43], thus demonstrating that the integer-factorization problem (the large factorization problem is the security foundation of RSA) can be efficiently solved on a quantum computer and is consequently in the complexity class BQP. This is almost exponentially faster than the most efficient known classical factoring algorithm, so we can say that these schemes [12–18, 42] are not resistant to quantum attacks. Different from classic schemes based on mathematical complexity, the security of quantum-based schemes is guaranteed by some properties of quantum mechanics, such as noncloning theorem and uncertainty principle. They are considered to have potential unconditional security and of course also include resistance to quantum attacks.

Recently, some researchers have tried to utilize quantum mechanics to design private query schemes. In 2008, Giovannetti et al. [44] proposed the first quantum private query (QPQ) protocol. The client sends the query $|j\rangle_Q$ and a decoy state $(|j\rangle_Q + |0\rangle_Q)/\sqrt{2}$ to the server in random order, then Bob uses each of them to interrogate his database using a qRAM (which records the reply to her queries in a register R) and returns $|j\rangle_Q|A_j\rangle_R$ or $(|j\rangle_Q|A_j\rangle_R + |0\rangle_Q|0\rangle_R)/\sqrt{2}$. The returned decoy state is used to check the eavesdropping of the server or the outside party. In 2011, Olejnik [45] presented a new QPQ protocol in a similar form with Giovannetti et al.'s protocol. By subtly selecting the oracle operation and the encoding scheme, one query state can achieve two aims simultaneously, i.e., obtaining the expected information and checking Bob's potential attack, so the communication complexity is reduced. Unfortunately, it is very vulnerable to the realities of significant transmission losses.

Therefore, Jakobi et al. [39] proposed a novel QPQ protocol (J11) based on the QKD protocol, where QKD is essentially a quantum analog of SE. In this protocol, an asymmetric key can be distributed between Alice and Bob by utilizing SARG04 QKD protocol, and Bob encrypts the whole database with the QKD key. Alice only knows few bits of the key, which ensures the database privacy. Compared with the previous QPQ protocols, J11 protocol is loss-tolerant and more secure. What is more, the J11 protocol can be easily generalized to the large database. Later, Gao et al. [40] proposed a flexible generalized protocol (G12) based on the J11 protocol, which introduced a variable θ to adjust the balance between database security and client privacy. Considering a database with size N, the J11 and G12 protocols have a communication complexity of $O(N \log N)$. In order to reduce the complexity, Rao et al. [41] gave two more efficient protocols (R13), which reduced the number of exchanged qubits to $O(N)$.

Different from classical encryption schemes based on some mathematical difficult problems, these findings have shown the potential in either the improvements of efficiency or the enhancements of security in cloud computing field with large computing resources and also brought new quantum technologies to solve private database query problems. However, to the best of our knowledge, there are few studies focusing on the quantum-based privacy-preserving database query problem in cloud environment. Therefore, we combine quantum mechanics with cloud database queries and proposed a QBDQ which aims to realize the privacy preservation for the clients and cloud server.

6. Conclusion and Future Work

As far as we know, the existing QPQ schemes either belong to the qRAM-based schemes, such as Giovannetti et al.'s [44] and Olejnik's [45] schemes, or belong to the QKD-based schemes, such as Jakobi et al.'s [39], Gao et al.'s [40], and Rao et al.' s [41] schemes. These QKD-based schemes solve the problem of the server's privacy; their communication complexity needs to be further reduced. In this study, we propose an efficient quantum private query scheme based on oracle operation, modified Grover iteration, oblivious transfer strategy, and the special offset encryption mechanism rather than QKD or qRAM. Compared with those schemes, our QBDQ scheme shows higher efficiency in terms of the communication complexity, the consumption of exchanged message, and the quantum measurement.

In our QBDQ scheme, we adopt the oblivious transfer strategy to solve the problem of the client's privacy; i.e., the client will ask the server to transmit all these encrypted data items to him/her. But in a real-world cloud environment, this is not a good approach. Although it guarantees that there is no information about the query index to be leaked, but it needs to transmit too many data items from the cloud database. Even if quantum resources have an exponential high-capacity advantage, it is also a waste of resources. Maybe the "query window" strategy is a better choice. To be specific, the client can firstly choose an index window that contains the desirable query item and ask the server to transmit these encrypted data items in this window scope other than the all data items, to him/her in a quantum way. Although there is certain information leakage from the perspective of information theory, it can save quantum resources. In this strategy, the selection of the size of a query window is a key point. In order to achieve a balance of efficiency and security, perhaps some

game theory (such as, Nash Equilibrium [46, 47]) and penalty functions [48–50] can provide relevant optimized solutions.

It is worth noting that although the proposed solution involves two clients, for the sake of brevity (and for comparison with other quantum schemes), Alice and Bob do not interact. This is the most common pattern in cloud database queries. For the multiparty joint inquiry method, we will discuss it in future work. In addition, we just consider the ideal framework of the privacy-preserving database query in cloud environment; i.e., all the clients and cloud server are semihonest. But in a real cloud environment, clients and servers may be untrustworthy. How to generalize our QBDQ into such multiuser and the untrusted scenario is an interesting work.

Data Availability

The database items and the corresponding encryption keys 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.

Acknowledgments

This work is supported by Nature Science Foundation of China (Grants nos. 71461005, 61502101, 61501247, and 61672290), Natural Science Foundation of Jiangsu Province (Grant no. BK20171458), Natural Science Foundation for Colleges and Universities of Jiangsu Province (Grant no. 16KJB520030), the Six Talent Peaks Project of Jiangsu Province (Grant no. 2015-XXRJ-013), and the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References

[1] M. Armbrust, A. Fox, R. Griffith et al., "Above the clouds: a berkeley view of cloud computing," Tech. Rep. UCB/EECS-2009-28, EECS Department, University of California, Berkeley, 2009.

[2] Y. E. Gelogo and S. Lee, "Database management system as a cloud service," *International Journal of Future Generation Communication and Networking*, vol. 5, no. 2, pp. 71–76, 2012.

[3] M. Seibold and A. Kemper, "Database as a service," *Datenbank-Spektrum*, vol. 12, no. 1, pp. 59–62, 2012.

[4] H. Takabi, J. B. D. Joshi, and G.-J. Ahn, "Security and privacy challenges in cloud computing environments," *IEEE Security & Privacy*, vol. 8, no. 6, pp. 24–31, 2010.

[5] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[6] Y. Xu, L. Qi, W. Dou, and J. Yu, "Privacy-preserving and scalable service recommendation based on simhash in a distributed cloud environment," *Complexity*, vol. 2017, Article ID 3437854, 2017.

[7] C. Yan, X. Cui, L. Qi, X. Xu, and X. Zhang, "Privacy-aware data publishing and integration for collaborative service recommendation," *IEEE Access*, vol. 6, pp. 43021–43028, 2018.

[8] W. Gong, L. Qi, and Y. Xu, "Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 3075849, 2018.

[9] L. Qi, S. Meng, X. Zhang et al., "An exception handling approach for privacy-preserving service recommendation failure in a cloud environment," *Sensors*, vol. 18, no. 7, p. 2037, 2018.

[10] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of Secure Computation*, vol. 4, no. 11, pp. 169–180, 1978.

[11] X. A. Wang, F. Xhafa, J. Ma, Y. Cao, and D. Tang, "Reusable garbled gates for new fully homomorphic encryption service," *International Journal of Web and Grid Services*, vol. 13, no. 1, pp. 25–48, 2017.

[12] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06)*, pp. 89–98, November 2006.

[13] X. A. Wang, J. Ma, F. Xhafa, M. Zhang, and X. Luo, "Cost-effective secure E-health cloud system using identity based cryptographic techniques," *Future Generation Computer Systems*, vol. 67, pp. 242–254, 2017.

[14] Y. Lu and G. Tsudik, "Privacy-preserving cloud database querying," *Journal of Internet Services and Information Security*, vol. 1, no. 4, pp. 5–25, 2011.

[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*, pp. 44–55, IEEE, USA, May 2000.

[16] X. A. Wang, F. Xhafa, W. Cai, J. Ma, and F. Wei, "Efficient privacy preserving predicate encryption with fine-grained searchable capability for Cloud storage," *Computers and Electrical Engineering*, vol. 56, pp. 871–883, 2016.

[17] B. Wang, W. Song, W. Lou, and Y. T. Hou, "Inverted index based multi-keyword public-key searchable encryption with strong privacy guarantee," in *Proceedings of the 34th IEEE Annual Conference on Computer Communications and Networks, IEEE INFOCOM 2015*, pp. 2092–2100, Hong Kong, May 2015.

[18] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in *Proceedings of the 30th IEEE International Conference on Distributed Computing Systems, ICDCS 2010*, pp. 253–262, 2010.

[19] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (SFCS '94)*, pp. 124–134, IEEE, USA, 1994.

[20] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Physical Review Letters*, vol. 79, no. 2, pp. 325–328, 1997.

[21] C. H. Bennett and G. Brassard, "Quantum cryptography: public key distribution and coin tossing," in *Proceedings of the IEEE International Conference on Computers, Systems, and Signal Processing*, pp. 175–179, India, 1984.

[22] A. K. Ekert, "Quantum cryptography based on Bell's theorem," *Physical Review Letters*, vol. 67, no. 6, pp. 661–663, 1991.

[23] M. Hillery, V. Buzek, and A. Berthiaume, "Quantum secret sharing," *Physical Review A: Atomic, Molecular and Optical Physics*, vol. 59, no. 3, pp. 1829–1834, 1999.

[24] Z. Dou, G. Xu, X. Chen, and K. Yuan, "Rational non-hierarchical quantum state sharing protocol," *Computers Materials & Continua*, vol. 58, no. 2, pp. 335–347, 2018.

[25] W.-J. Liu, Z.-Y. Chen, S. Ji, H.-B. Wang, and J. Zhang, "Multi-party semi-quantum key agreement with delegating quantum computation," *International Journal of Theoretical Physics*, vol. 56, no. 10, pp. 3164–3174, 2017.

[26] W.-J. Liu, Y. Xu, C.-N. Yang, P.-P. Gao, and W.-B. Yu, "An efficient and secure arbitrary N-party quantum key agreement protocol using bell states," *International Journal of Theoretical Physics*, vol. 57, no. 1, pp. 195–207, 2018.

[27] W.-J. Liu, H.-W. Chen, T.-H. Ma, Z.-Q. Li, Z.-H. Liu, and W.-B. Hu, "An efficient deterministic secure quantum communication scheme based on cluster states and identity authentication," *Chinese Physics B*, vol. 18, no. 10, pp. 4105–4109, 2009.

[28] J. Zhong, Z. Liu, and J. Xu, "Analysis and improvement of an efficient controlled quantum secure direct communication and authentication protocol," *Computers Materials & Continua*, vol. 57, no. 3, pp. 621–633, 2018.

[29] Z. Qu, T. Zhu, J. Wang, and X. Wang, "A novel quantum stegonagraphy based on brown states," *Computers, Materials and Continua*, vol. 56, no. 1, pp. 47–59, 2018.

[30] X. Tan, X. Li, and P. Yang, "Perfect quantum teleportation via Bell states," *Computers Materials & Continua*, vol. 57, no. 3, pp. 495–503, 2018.

[31] W.-J. Liu, Z.-F. Chen, C. Liu, and Y. Zheng, "Improved deterministic N-To-one joint remote preparation of an arbitrary qubit via EPR pairs," *International Journal of Theoretical Physics*, vol. 54, no. 2, pp. 472–483, 2015.

[32] M. Wang, C. Yang, and R. Mousoli, "Controlled cyclic remote state preparation of arbitrary qubit states," *Computers, Materials and Continua*, vol. 55, no. 2, pp. 321–329, 2018.

[33] W.-J. Liu, F. Wang, S. Ji, Z.-G. Qu, and X.-J. Wang, "Attacks and improvement of quantum sealed-bid auction with EPR pairs," *Communications in Theoretical Physics*, vol. 61, no. 6, pp. 686–690, 2014.

[34] W.-J. Liu, H.-B. Wang, G.-L. Yuan et al., "Multiparty quantum sealed-bid auction using single photons as message carrier," *Quantum Information Processing*, vol. 15, no. 2, pp. 869–879, 2016.

[35] W. J. Liu, Z. Y. Chen, J. S. Liu, Z. F. Su, and L. H. Chi, "Full-blind delegating private quantum computation," *Computers Materials & Continua*, vol. 55, no. 2, pp. 321–329, 2018.

[36] W.-J. Liu, P.-P. Gao, W.-B. Yu, Z.-G. Qu, and C.-N. Yang, "Quantum relief algorithm," *Quantum Information Processing*, vol. 17, no. 10, p. 280, 2018.

[37] W. J. Liu, P. P. Gao, Y. X. Wang, W. B. Yu, and M. J. Zhang, "A unitary weights based one-iteration quantum perceptron algorithm for non-ideal training sets," *IEEE Access*, 2019.

[38] M. O. Rabin, "How to exchange secrets with oblivious transfer," *IACR Cryptology Eprint Archive*, vol. 2005, p. 187, 2005.

[39] M. Jakobi, C. Simon, N. Gisin et al., "Practical private database queries based on a quantum-key-distribution protocol," *Physical Review A: Atomic, Molecular and Optical Physics*, vol. 83, no. 2, Article ID 022301, 2011.

[40] F. Gao, B. Liu, Q.-Y. Wen, and H. Chen, "Flexible quantum private queries based on quantum key distribution," *Optics Express*, vol. 20, no. 16, pp. 17411–17420, 2012.

[41] M. V. Panduranga Rao and M. Jakobi, "Towards communication-efficient quantum oblivious key distribution," *Physical Review A: Atomic, Molecular and Optical Physics*, vol. 87, no. 1, Article ID 012331, 2013.

[42] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proceedings of the 1995 IEEE 36th Annual Symposium on Foundations of Computer Science*, pp. 41–50, 1995.

[43] D. Beckman, A. N. Chari, S. Devabhaktuni, and J. Preskill, "Efficient networks for quantum factoring," *Physical Review A: Atomic, Molecular and Optical Physics*, vol. 54, no. 2, pp. 1034–1063, 1996.

[44] V. Giovannetti, S. Lloyd, and L. Maccone, "Quantum private queries," *Physical Review Letters*, vol. 100, no. 23, Article ID 230502, 2008.

[45] L. Olejnik, "Secure quantum private information retrieval using phase-encoded queries," *Physical Review A: Atomic, Molecular and Optical Physics*, vol. 84, no. 2, Article ID 022313, 2011.

[46] J. Zhang, B. Qu, and N. Xiu, "Some projection-like methods for the generalized Nash equilibria," *Computational Optimization and Applications*, vol. 45, no. 1, pp. 89–109, 2010.

[47] B. Qu and J. Zhao, "Methods for solving generalized nash equilibrium," *Journal of Applied Mathematics*, vol. 2013, Article ID 762165, 6 pages, 2013.

[48] C. Wang, C. Ma, and J. Zhou, "A new class of exact penalty functions and penalty algorithms," *Journal of Global Optimization*, vol. 58, no. 1, pp. 51–73, 2014.

[49] S. Lian and Y. Duan, "Smoothing of the lower-order exact penalty function for inequality constrained optimization," *Journal of Inequalities and Applications*, vol. 2016, no. 1, p. 185, 2016.

[50] S. Li and Y. Zhang, "On-line scheduling on parallel machines to minimize the makespan," *Journal of Systems Science & Complexity*, vol. 29, no. 2, pp. 472–477, 2016.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 4049765, 16 pages
https://doi.org/10.1155/2019/4049765

Research Article

A Cooperative Denoising Algorithm with Interactive Dynamic Adjustment Function for Security of Stacker in Industrial Internet of Things

Darong Huang,[1] Lanyan Ke,[1] Bo Mi,[1] Guosheng Wei,[2] Jian Wang,[2] and Shaohua Wan[3]

[1]*College of Information Science and Engineering, Chongqing Jiaotong University, Chongqing 400074, China*
[2]*Beijing Nanri Technology Co. Ltd., Beijing 100082, China*
[3]*School of Information and Safety Engineering, Zhongnan University of Economics and Law, Wuhan 430073, China*

Correspondence should be addressed to Lanyan Ke; 622160070015@mails.cqjtu.edu.cn

Received 23 August 2018; Revised 4 December 2018; Accepted 12 January 2019; Published 3 February 2019

Guest Editor: Xuyun Zhang

In order to more effectively eliminate the disturbance of vibration signal to ensure the security monitoring of stacker be more accurate in Industrial Internet of Things (IIoT), a cooperative denoising algorithm with interactive dynamic adjustment function was constructed and proposed. First, some basic theories such as EMD, EEMD, LMS, and VSLMS were introduced in detail according the characteristics of stacker in IIoT. Meanwhile, the advantages and disadvantages of varieties of algorithms have been analyzed. Secondly, based on the traditional VSLMS-EEMD, an improved VSLMS-EEMD was proposed. Thirdly, to guarantee the denoising effect of security monitoring in IIoT, a cooperative denosing model and framework named as IDVSLMS-EEMD was designed and constructed based on the advantages of LMS, VSLMS, and improved VSLMS-EEMD. In addition, the assignment rules and models of the corresponding weight coefficients were also set up according to the features of the error signal of denoising process in IIoT. At the same time, we have designed a cooperative denoising algorithm with interactive dynamic adjustment function. And some evaluated indexes such as NSR and SDR were selected and introduced to evaluate the effectiveness of the different algorithms. Thirdly, some simulation examples and real experiment examples of stacker running signals under abnormal condition, which has been developed and applied in Power Grid of China, was used to verify and simulate the effectiveness of our presented algorithm. The experiment comparison results have shown that our algorithm can improve the denosing effect. Finally, some conclusions were discussed and the directions for future engineering application were also pointed out.

1. Introduction

With the development and evolution of society, the Industrial Internet of Things (IIoT) plays a significant role in guiding the process of intelligent manufacturing for global industry [1–3]. So, the IIoT has been embedded in various industry systems, especially in the ASRS system. As we all know, the main function of ASRS is to grab, move, and stack goods from one piece of equipment to another. Therefore, stacker is the most important element in ASRS. In practice scene of ASRS, one point of the stacker running which cannot reach the design requirement can be deemed as an accident. As an indispensable device of IIoT, this accident is related not only to the security of ASRS but also to the data acquisition and the data exchange of IIoT [4–6]. For example, because of the long-term wear and tear of the stacker track, the weld seam of the track is enlarged or pits appear which leads to the decrease of positioning accuracy, thus seriously affecting the data acquisition of the whole IIoT. So, how to use monitoring data to ensure the security of stacker is very important in IIoT [6–8]. Notice, in practice, as the primary data resource for security monitoring and maintenance of the systems, mechanical vibration signals are always influenced by any strong interferences of surrounding environment. However, the strong noise always conceals the abnormal characteristic information or forms false characteristics, which has greatly

affected the further abnormal detection of stackers in IIoT [9–11]. Hereby, denosing the noise of vibration data for stacker under the abnormal condition is not only the primary premise to ensure the further effective detection of the abnormity, but also the necessary measure to ensure the security of the IIoT. Moreover, along with the continuous improvement of operation speed, to find the algorithm and model to denoise has become more and more urgent for vibration signals in IIoT.

At present, how to effectively eliminate and filter the disturbance noise from measured signals is the prerequisite for health monitoring of industrial systems [12]. So, domestic and foreign researchers have made a lot of progress from various aspects. For instance, some researchers have constructed some data denoising models by combining with the Kalman filter and machine learning to separate the noise and useful stationary signals by high pass, low pass, band pass, or band stop in several ways in [13, 14]. The simulation results showed that the proposed model and algorithm have better performance of separation for mixed signals with nonoverlapping power spectrum. Unfortunately, the data resources acquired in health monitoring for IIoT always contain lots of noise. This perhaps leads to appearance of spectrum aliasing. Obviously, the filter model and algorithm based on frequency domain are not suitable. So, it is necessary to construct the reasonable algorithm to improve the denoising effect for mixed signal with nonoverlapping spectrum.

To overcome the shortage, the multipoint mean smoothing denosing method was constructed and simulated to distinguish and separate useful signals from noise by the frequency difference in [15]. However, the presented algorithm may achieve the denoising effect for stationary signals. Nevertheless, most of the mechanical vibration signals, which are measured from real IIoT, are nonstationary. Because of this, some scholars have studied and established the denoising method by combining with canonical correlation analysis and empirical model decomposition (EMD) in [16]. In their experiments, this denoising model has improved the denoising accuracy based on multipoint mean smoothing denosing method to some extent, but there is the problem of end effect on EMD. Thus, there are still many defects when the above methods were used to implement noise-elimination in the university.

Furthermore, the wavelet theory was introduced to depict the characteristics according to the different amplitude of signals and noise in [17–20]. In that case, the simulation results showed that the presented algorithms and methods were effectively for the nonstationary signals to a certain extent. But, it is difficult to determine the threshold and set up a reasonable order of the filter in real practical project. In fact, to select the perfect threshold and set up the reasonable order is the key step of wavelet denoising model for health monitoring and safety maintenance of IIoT. Similarly, the denoising model based on EMD algorithm also has the defect of selection principle of the threshold and filter order [21]. In order to solve this problem, some scholars has discussed and analyzed some solutions in [22–25]. Although these solutions have achieved certain improvement, EMD has end effect. For improving the end effect of EMD, the Ensemble Empirical Mode Decomposition (EEMD) has been introduced to solve the shortage of denoising algorithm based on EMD in [26]. However, EEMD still faces the problem of threshold selection. Therefore, the adaptive equalizing algorithms without threshold selection have been widely used for industry signal denoising in [27, 28]. Their experiment had verified and indicated that the denoising performance may be achieved to a certain extent. But the denoising performance based adaptive equalizing algorithm is not stable for wide-band signals. So, the engineers need constantly to improve and update the denoising algorithms to ensure the effectiveness of health monitoring and safety maintenance in IIoT.

Based on this, many scholars and engineers have tried to construct and establish the improved model combined with EMMD and other methods such as LMS, Gath-Geva clustering, and so on in [29–31]. And then the wide-band signals may be transformed to the narrow band signal by these improved models. The denoising effect of these improved models may be guaranteed in processing data of IIoT. Regrettably, the convergence speed is very slow while the step size of LMS algorithm cannot be adaptive adjusted. Therefore, considering that the step factor may be adaptive adjusted, some scholars had presented an improved algorithm based on LMS. The new model is named as VSLMS. Based the thesis, Yu Xiao and his coauthors had constructed the new denoising algorithm by combining with VSLMS and EEMD to solve the accuracy of the denoising performance in [32]. But, in [32], the adjustment of the step factor for VSLMS is seriously affected by error signal at the present time. To remedy this problem, another VSLMS algorithm is introduced into combining with EEMD to construct an improved VSLMS-EEMD algorithm. Notice that the practical data acquired from IIoT will be influenced by the different factors such as the fault of sensors, the performance degradation of equipment, and so on. So, the noise in the practical engineering is particularly complex. For all points of the vibration signal, the above algorithm cannot achieve the best performance for all points. Aiming at each point of the vibration signal, different algorithms have different denoising performance. Thus, based on the fact that the improved VSLMS-EEMD algorithm is proposed, to find a cooperative mechanism to maximize the denoising performance at each point based on the above denosing algorithms is very important to process the nonstationary signal in IIoT.

Based on this thesis, a cooperative denoising algorithm and model with interactive dynamic adjustment function have been analyzed and discussed in further section. The layout of this paper is arranged as follows. In Section 2, we have introduced the basic theories and methods such as EMD, EEMD, LMS, and VSLMS. In Section 3, an interactive dynamic adjusted denoising algorithm has been designed and analyzed. Meanwhile, some evaluated indexes were selected and introduced to evaluate the effectiveness of the different algorithms. In Section 4, to verify the effectiveness of the proposed algorithm, some simulative examples were implemented to compare the denoising performance of LMS, VSLMS, VSLMS-EEMD, and presented algorithm. In addition, to enlarge the applications, the practical denoising

project of stacker running signals, which have been developed and applied in Power Grid of China, was used to verify the effectiveness of our presented algorithm. Finally, some conclusions and the directions for future engineering application are discussed according to the real simulation results in health monitoring and safety maintenance of IIoT.

2. Introduction and Analysis of Basic Theory and Model in Health Monitoring of IIoT

In practical engineering of IIoT, as we all know, the measurement signals are always typical and nonstationary, and they are the direct information resource of actual sense for IIoT, including running state, fault modes, and so on. Thus, the measurement signals obtained in actual IIoT contain inevitably strong background noise, which makes the useful information submerged. Obviously, the information features of health monitoring are not obvious for IIoT. Thus, how to design and find an effective denoising algorithm with interactive dynamic adjustment function is key step to guarantee the performance of safety maintenance of IIoT. Meanwhile, the denoising model and algorithm must achieve both timeliness and stability in health monitoring of IIoT. Only then can the work be of great theoretical and practical significance for nonstationary signals of health monitoring in IIoT.

On this basis, some basic theoretical models will be introduced and discussed for designing the cooperative denoising algorithm with interactive dynamic adjustment function in the next section.

2.1. Empirical Mode Decomposition Algorithm (EMD). For simplicity of analysis in health monitoring of IIoT, the measurement signal $S(t)$ including the useful signal and noise may be simply described as follows:

$$S(t) = s(t) + v(t) \tag{1}$$

where $s(t)$ represents the useful vibration signal and $v(t)$ describes the disturbance noise.

According to the basic thesis, the detailed decomposition process of measurement mixed signals is shown as follows.

Step 1. Suppose that the symbol $S(t)$ represents the original signal of IIoT. If all local extremums can be found, the upper envelope $u(t)$ and the lower envelope $l(t)$ may be computed by using the cubic spine function. Meanwhile, the envelope line should contain all the data.

Step 2. Combined with $u(t)$ and $l(t)$, the mean value of the upper and lower envelope may be used to process the original envelope; i.e.

$$m_1 = \frac{u(t) + l(t)}{2} \tag{2}$$

To separate first component from original signal, we have introduced the computing formula as follows:

$$h_1(t) = S(t) - m_1(t) \tag{3}$$

If $h_1(t)$ meets IMF condition, it is the first component of $S(t)$. Otherwise, go on the next step.

Step 3. Let $h_1(t)$ be the new original signal and repeat Step 2 again until the IMF condition can be met. The corresponding computed formula is as follows:

$$h_{11}(t) = h_1\text{-}m_{11}, \ldots, h_{ik}(t) = h_{1(k-1)}\text{-}m_{1k} \tag{4}$$

where m_{1k} indicates the mean value of the upper and lower envelope of $h_{1(k-1)}$ and k is the number of iterations.

In that case, h_{1k} should meet the IMF conditions. Then, the first IMF component of original signal can be gotten; i.e., $c_1(t) = h_{1k}(t)$.

Meanwhile, the new signal may be separated from the original signal by the following formula:

$$r_1(t) = S(t) - c_1(t) \tag{5}$$

And go on to the next step.

Step 4. The filtering process in Step 2 is used to repeatedly execute for $r_1(t)$ until the IMF condition is met. In other words, the second IMF component and the similar new signal are denoted as c_2 and $r_2(t)$, respectively. Similarly, all IMF components and new original signals are represented as follows:

$$r_2(t) = r_1(t) - c_2(t), \ldots, r_n(t) = r_{n-1}(t-1) - c_n(t) \tag{6}$$

Step 5. It is rewriting the original signal $S(t)$ by the following mode:

$$S(t) = \sum_{i=1}^{N} c_i(t) + r_n(t) \tag{7}$$

where $r_n(t)$ is the remainder which presents the monotonous trend of $S(t)$. Obviously, the decomposition results IMF $s(c_1, \cdots, c_n)$ indicate the different IMF components which represent from high frequency to low frequency distribution of the original signal.

If we use the EMD to decompose the nonstationary signal in practice, there is one thing we have noticed: the EMD method has serious end effect and mode mixing of different time-scale IMF. Of course, the lacks caused by EMD signal decomposition will affect the denoising effect of the original signal in IIoT. So, how to improve the efficiency of noise reduction is very important in practice engineering. Next, we will introduce in depth the basic principles and related situations to establish an improvement algorithm.

2.2. Ensemble Empirical Mode Decomposition (EEMD). To overcome the influence of the end effect and mode mixing in health monitoring of IIoT, an improved denoising algorithm named as Ensemble Empirical Mode Decomposition (EEMD) has been proposed based on EMD for signal denoising. The decomposition steps of EEMD are shown as follows.

Step 1. It is adding a Gaussian random white noise $w(t)$ to original measurement signal of IIoT; i.e.,

$$S_1(t) = S(t) + w(t) \tag{8}$$

where $w(t) \sim N(\mu, \sigma^2)$.

Step 2. It is decomposing the new original signal $S_1(t)$ by using EMD algorithm. And then each IMF component $c_{ij}(t)$, $j = 1, 2, ...K$ may be obtained and acquired in time, where $c_{ij}(t)$ presents the jth IMF component when the ith white noise has been added into the original signal.

Step 3. It is repeating Steps 1–2 to decompose the renewal signal with different Gaussian white noise again. And we may obtain a set of new IMFs, which are quite different from the original ones.

Step 4. It is computing the average value of the IMFs obtained by decomposing the corresponding renewal signal with different Gaussian white noise; i.e.,

$$c_i(t) = \frac{1}{T}\sum_{i=1}^{T} c_{ij}(t) \tag{9}$$

where $c_i(t)$ is the ith IMF component.

So, the decomposition results IMF $s(c_1, \cdots, c_n)$ can be selected to represent the different IMF components from high frequency to low frequency distribution of the original signal.

In fact, the highest advantage of EEMD is that IMFs decomposed by the algorithm are independent and can prevent IMFs from mode mixing. In that case, it is vital to adaptively decompose the measurement signal of IIoT. But, as we all know, the effect of signal processing is always greatly influenced by choice of the decomposition threshold when EEMD is used to denoise for the measurement signal in IIoT.

Therefore, to further guarantee the effect and accuracy of selecting the decomposition threshold in processing the mixed signal, many engineers and researchers have tried to focus on finding out some helper methods to modify the defect of EMMD. Based on this, the typical LMS algorithm will be introduced to solve the problem of the decomposition threshold in further section.

2.3. Least Mean Square (LMS) Algorithm. In the security monitoring of IIoT, it is necessary to find an adaptive algorithm to reduce or inhibit the correlative noise. So, to get the more ideal signal, IMF $s(c_1, \cdots, c_n)$ or original signal should be used as the training specimen to further process. In that case, take the IMF $s(c_1, \cdots, c_n)$ as an example, so the initial input vector of training is described as follows:

$$\text{IMF}s(n) = [c(n), c(n-1), \ldots, c(n-M-1)]^T \tag{10}$$

where M represents the number of tap coefficients.

For the sake of simplicity, the equalized signal of the training iteration is supposed as follows:

$$y(n) = \sum_{i=0}^{M-1} w_i(n)\, c(n-i) \tag{11}$$

where $w_i(n)$ is the weight coefficient of every component of IMFs.

For the convenience of calculation, the above formula may be simplified as follows:

$$Y(n) = W^T(n) \cdot \text{IMFs}(n) \tag{12}$$

where $W(n)$ is the weight coefficient matrix; i.e.,

$$W(n) = [w_0(n), w_1(n), \cdots, w_{M-1}(n)]. \tag{13}$$

where $W(n)$ is calculated as follows:

$$W(n+1) = W(n) + 2\mu \cdot e(n) \cdot \text{IMFs}(n) \tag{14}$$

where μ is the step factor and $e(n)$ is error signal which is modeled as follows:

$$e(n) = d(n) - y(n) = d(n) - W^T(n) \cdot \text{IMFs}(n) \tag{15}$$

where $d(n)$ represents the actual value of each iteration training.

Although the algorithm may reduce the error accumulation effect in fine processing of nonstationary signal and improve the denoising accuracy, the convergence is slow. From a practical situation, one reason might be that the fixed step size cannot keep the insistency between the fast convergence speed and steady residual error [33–35]. Therefore, we need to find a method to modify the shortage according to the actual requirements.

2.4. LMS Algorithm with Variable Step Factor (VSLMS). As is well known, the denoising accuracy of nonstationary signal in IIoT is usually affected by varieties of factors, such as the testing environment, test methods, and so on. Furthermore, the training signals acquired by using LMS algorithm may still contain the strong noise because of the fixed step size. So, the amplitude of characteristic information cannot be evidently separated from the noise information. In brief, the residual noise has brought great obstacles for the denoising performance of nonstationary signal in IIoT. To overcome the problem, in this section, the variable step factor is inducted to the denoising control to balance the insistency between the fast convergence speed and steady residual error. The core of the thesis is that the step size can be dynamically adjusted according to the error signal of each training.

In formula (14), the updating of the fixed step size should to be related to the current time error $e(n)$, which results in the characteristic's confusion. So, the computing method is shown as follows.

$$\mu(n) = \left(\frac{1}{1 + \exp\left(-\alpha\, |e(n)|^m\right)} - 0.5\right) \tag{16}$$

where α is the control parameter. The value of parameter is taken according to various concrete statuses.

In practical health monitoring of IIoT, we find the abnormal phenomenon that the error signal has the cumulative effect with experimental time. Further, the phenomenon results in the serious overlapping interference of denoising signal. So, to overcome the shortage, the error values at the current time and the last time are inducted to the adjustment of the step size. In other words, the step size may be gotten by the following formula.

$$\mu(n) = \left(\frac{1}{1 + \exp\left(-\alpha\, |e(n) * e(n-1)|^m\right)} - 0.5\right) \tag{17}$$

Thus, the weight coefficients may be rewritten as follows:

$$W(n+1) = W(n) + 2\mu(n) \cdot e(n) \cdot \text{IMFs}(n) \quad (18)$$

In conclusion, the improved LMS with variable step factor can not only decrease the noise sensitivity but also improve the convergence performance. This is because of the improvement mentioned above that the improved weight coefficients can filter the influence of the cumulative effect in the training. Therefore, we can make use of the improved algorithm to denoise the nonstationary health monitoring of IIoT.

Obviously, we can see from the above analysis that each method has advantages and disadvantages in denoising process of nonstationary signal. If we may establish an integrated strategy to exert the advantages of each method and minimize the influence to disadvantages, thus the denoising effect of nonstationary signal may be vastly improved in health monitoring of IIoT. Next, the work will be in detail depicted.

3. Design and Analysis of Cooperative Denoising Algorithm and Model with Interactive Dynamic Adjustment Function (IDVSLMS-EEMD)

3.1. Analysis and Establishment of Cooperative Denoising Model with Interactive Dynamic Adjustment Function. To guarantee the denoising performance of nonstationary signal in health monitoring of IIoT, we have tried to design some cell modules to realize the task of the integration and configurable controls. With this goal, we have designed the LMS denoising module, VSLMS denoising module and proposed the improved VSLMS-EEMD denoising module based on the traditional VSLMS-EEMD, respectively. The denoising module by using LMS or VSLMS is shown as Figure 1.

In addition, to overcome the shortage of the VSLMS-EEMD proposed in [24], the step updating algorithm is redesigned as (17) to construct an improved VSLMS-EEMD denoising algorithm. The framework of the improved VSLMS-EEMD is shown as Figure 2.

In fact, all these cell modules can be used to denoise of nonstationary signal in IIoT, and then each module can be used as a single denoising processor. However, the operations staff of health monitoring always want to highlight the advantages of these cell modules as large as possible. In order to maximize the denoising performance at each point, on the basis of the improved VSLMS-EEMD algorithm, a cooperative denosing algorithm with interactive dynamic adjustment function named as IDVSLMS-EEMD has been designed and constructed by using the stackable technology as Figure 3

Obviously, the framework can allow both those cell denoising modules (i.e., conventional and complementary) to exist in a framework that embarrasses neither. From an application perspective, the IDVSLMS-EEMD algorithm is a standardization of a set of denoising patterns based on a common set of denosing algorithm. So, one of the features of the IDVSLMS-EEMD model is able to move applications from one processor environment to another. From viewpoint of practical operation, the outputs of three denoising algorithms embedded in the IDVSLMS-EEMD framework are different, the differences can make up for each other's mutual limitations. Therefore, the engineers can achieve the most optimal elimination at every point of the vibration signal for IIoT.

For the sake of analysis, relevant definition and calculation of the proposed cooperative denoising model IDVSLMS-EEMD are set as follows.

Firstly, the ith output of the IDVSLMS-EEMD algorithm may be defined as the following formula:

$$\widehat{S}_F(i) = w_i(1) \times \widehat{S}_1(i) + w_i(2) \times \widehat{S}_2(i) + w_i(3) \times \widehat{S}_3(i) \quad (19)$$

where $\widehat{S}_1(i)$ is the ith output of the LMS denosing module, $\widehat{S}_2(i)$ is the ith output of the VSLMS denosing, $\widehat{S}_3(i)$ is ith output of the improved VSLMS-EEMD denosing module, and $w_i(m)$ $(m = 1, 2, 3)$ is the weight of output in every denoising processor.

From this model, the hub of the cooperative denoising framework is to determine the weights of denoising output at different time. In fact, if the denoising module is more suitable for nonstationary some point of signal in IIoT, the weight is bigger. Otherwise, the weight is smaller. But, for error signals, the opposite is true. So, it can be inferred that the error signal is inversely related to the weight coefficient, and the weight coefficient can be obtained by the error signal.

Define the error signal set at ith point as follows:

$$e(i) = [e_1(i), e_2(i), e_3(i)] \quad (20)$$

According to the errors, the dynamical assignment rule of the weights is shown as follows.

Rule 1. The larger the error of single denoising processor is, the smaller the weight is. That is, consider the following.

(1) If $e_m(i)$ $(m = 1, 2, 3)$ is maximum value in $e(i) = [e_1(i), e_2(i), e_3(i)]$, the weight $w_i(m)$ may be assigned by the following formula:

$$\omega_i(m) = \frac{\min(e_1(i)\, e_2(i)\, e_3(i))}{e_1(i) + e_2(i) + e_3(i)} \quad (21)$$

(2) If $e_m(i)$ $(m = 1, 2, 3)$ is minimum value in $e(i) = [e_1(i), e_2(i), e_3(i)]$, the weight $w_i(m)$ may be assigned by the following formula.

$$\omega_i(m) = \frac{\max(e_1(i)\, e_2(i)\, e_3(i))}{e_1(i) + e_2(i) + e_3(i)} \quad (22)$$

(3) If $e_m(i)$ $(m = 1, 2, 3)$ is intermediate value in in $e(i) = [e_1(i), e_2(i), e_3(i)]$, the weight $w_i(m)$ may be assigned by the following formula.

$$\omega_i(m) = \frac{e_m(i)}{e_1(i) + e_2(i) + e_3(i)} \quad (23)$$

Through the assignment rule, the weight of every denoising module may be determined on each point according

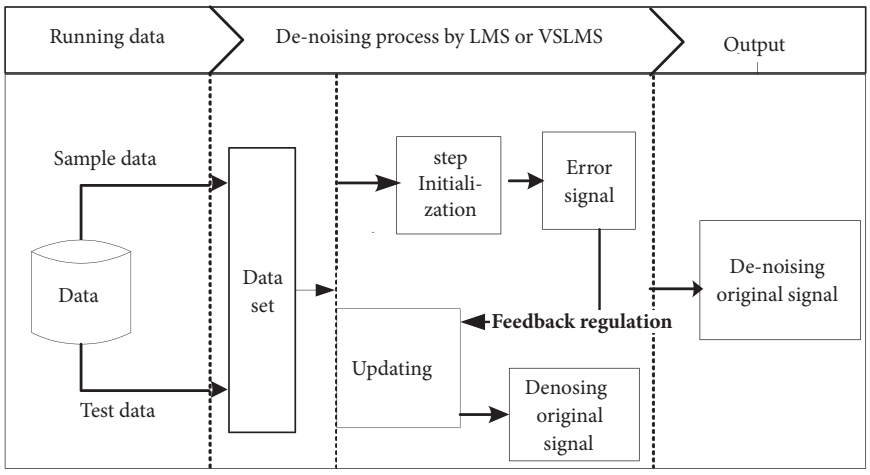


Figure 1: Denoising module of LMS or VSLMS.

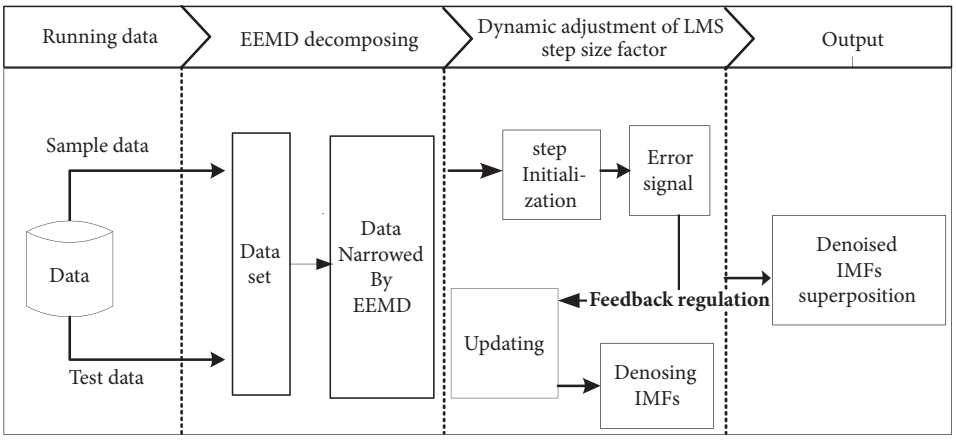


Figure 2: Improved VSLMS-EEMD denoising module.

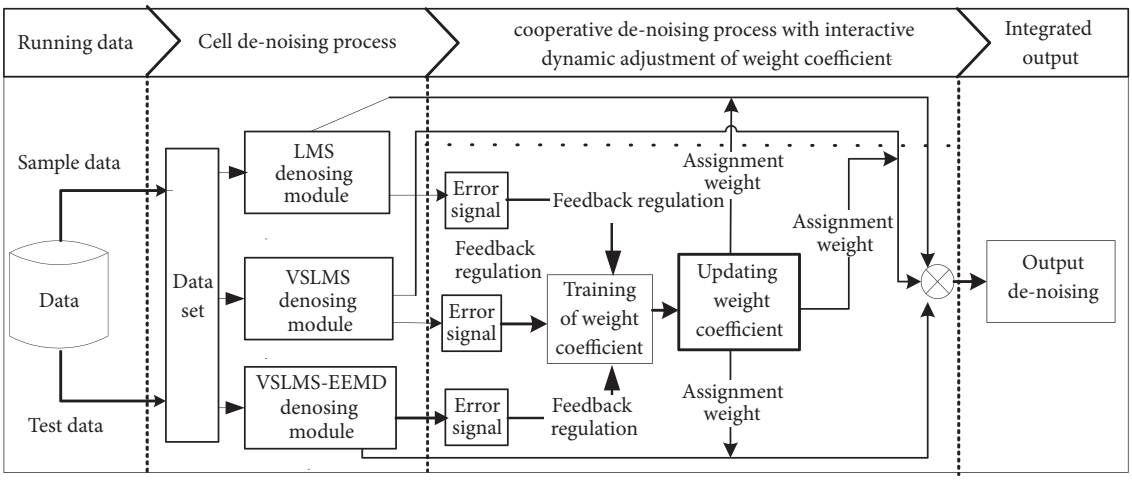


Figure 3: Integrated cooperative denoising framework of IDVSLMS-EEMD.

to the effect of denoising in IIoT. Obviously, the output of the single denoising module is ensured when the weight coefficient is dynamically adjusted in time. Of course, the denoising performance of the integrated system may be improved because each other makes use of mutual advantage and make up own shortage.

3.2. Evaluation Indexes of Integrated Cooperative Denoising Model. In the actual operation of integrated cooperative denoising framework, the success of achieving the performance goals depends on how well we develop the denoising strategy in health monitoring of IIoT. So, it is necessary to establish some scientific, systematic evaluation indexes of the cooperative denoising algorithm as feedback [36, 37].

To evaluate the effectiveness of presented model, we have constructed two indexes according to the actual situation of health monitoring in IIoT. These evaluating indexes and rules are set as follows.

(1) Absolute Value Error is

$$C = mean\left|\hat{S} - s\right| \tag{24}$$

where s is the original signal and $\hat{S}$ is the denoising output.

By formula (24), the evaluation rule is defined as follows.

Rule 2. The bigger the C is, the worse the denoising effect is and vice versa.

(2) Normalized Cross Correlation (NCC) is

$$NCC = \frac{\sum_{i=1}^{N} \hat{S}(n)\, s(n)}{\sqrt{\left(\sum_{i=1}^{N} \hat{S}^2(n)\right)\left(\sum_{i=1}^{N} s^2(n)\right)}} \tag{25}$$

where $\hat{S}(n)$ and $s(n)$ are the denosing output and the real value of the presented algorithm and n indicates the testing time. NCC represents the curve similarity between the denoising signal and the initial signal.

Similarity, the corresponding evaluation rule is designed as follows.

Rule 3. The larger the value of NCC is, the better the denoising effect is and vice versa.

So, the effect of the cooperative denoising model with interactive dynamic adjustment function may be evaluated by the above evaluation indexes.

3.3. Construction and Analysis of the Cooperative Denoising Algorithm with Interactive Dynamic Adjustment Function. Based on the above discussion, combining with the cooperative denoising framework, the cooperative denoising algorithm for nonstationary signal in IIoT may be designed in detail as below.

Step 1. It is initialization of system. Load the original signal of IIoT and determine the states of the algorithm switches to be off or on.

Step 2. Calculate the number of the switches that are on. If the number is equal to 3, step 3 is performed; otherwise Step 5 is performed.

Step 3. Obtain three denoised signals by using LMS, VSLMS, and VSLMS-EEMD denoising algorithms, respectively. The denoising process is divided into training stage and equaling stage.

(1) Training stage: for LMS denoising algorithm, the optimal weight coefficient W can be obtained by using (14)-(15); for VSLMS and VSLMS-EEMD denoising algorithm, the optimal weight coefficient W can be obtained by using (17)-(18).

(2) Equalizing stage: the optimum weight coefficient W obtained by the training stage is used to carry out equalization and noise elimination for original signals of IIoT by using (12) to obtain the denoised signals named $\hat{S}_1(i)$, $\hat{S}_2(i)$, and $\hat{S}_3(i)$, respectively.

Step 4. Obtain the dynamic adjustments of weight coefficients w.

(1) Obtain the error signal $e(i)$ shown as (20) by using (14).

(2) Obtain weight coefficients ω based on error signal obtained from a) by using Rule 1 that is shown as (21)-(23).

Step 5. Interactively denoising the IIoT signal by using (19).

Step 6. Repeat steps 1–5 until the number of processed signals is equal to length of the original signal.

Step 7. Evaluate denoising algorithms by using Rules 2 and 3 that are shown as (24)-(25) based on the denoised signals obtained from step 5.

The cooperative denoising flow chart is shown as Figure 4.

4. Simulation Examples

To verify the effectiveness and rationality of the presented algorithm, the simulation examples were first used to test the denoising ability to the network data packet of health monitoring in IIoT. In general, the simulation original signal $S(t)$ was described as follows.

$$S(t) = s(t) + v \tag{26}$$

where $s(t)$ represents the useful signal and $v(t)$ indicates the random noise.

In our simulation experiments, $s(t)$ and $v(t)$ were set up as below.

$$\begin{aligned} s(t) &= 0.13\cos(2\pi \times 20 \times t) + 0.08\sin(2\pi \times 10 \times t) \\ &\quad + 0.02\sin(2\pi \times 40 \times t) \\ v &= 0.18\,\mathrm{wgn}(L, 0) \end{aligned} \tag{27}$$

where L represents signal length.

In our simulation examples, L is set up as 2000. The comparison results between the original signal and compounded signal with noise are shown as Figure 5.

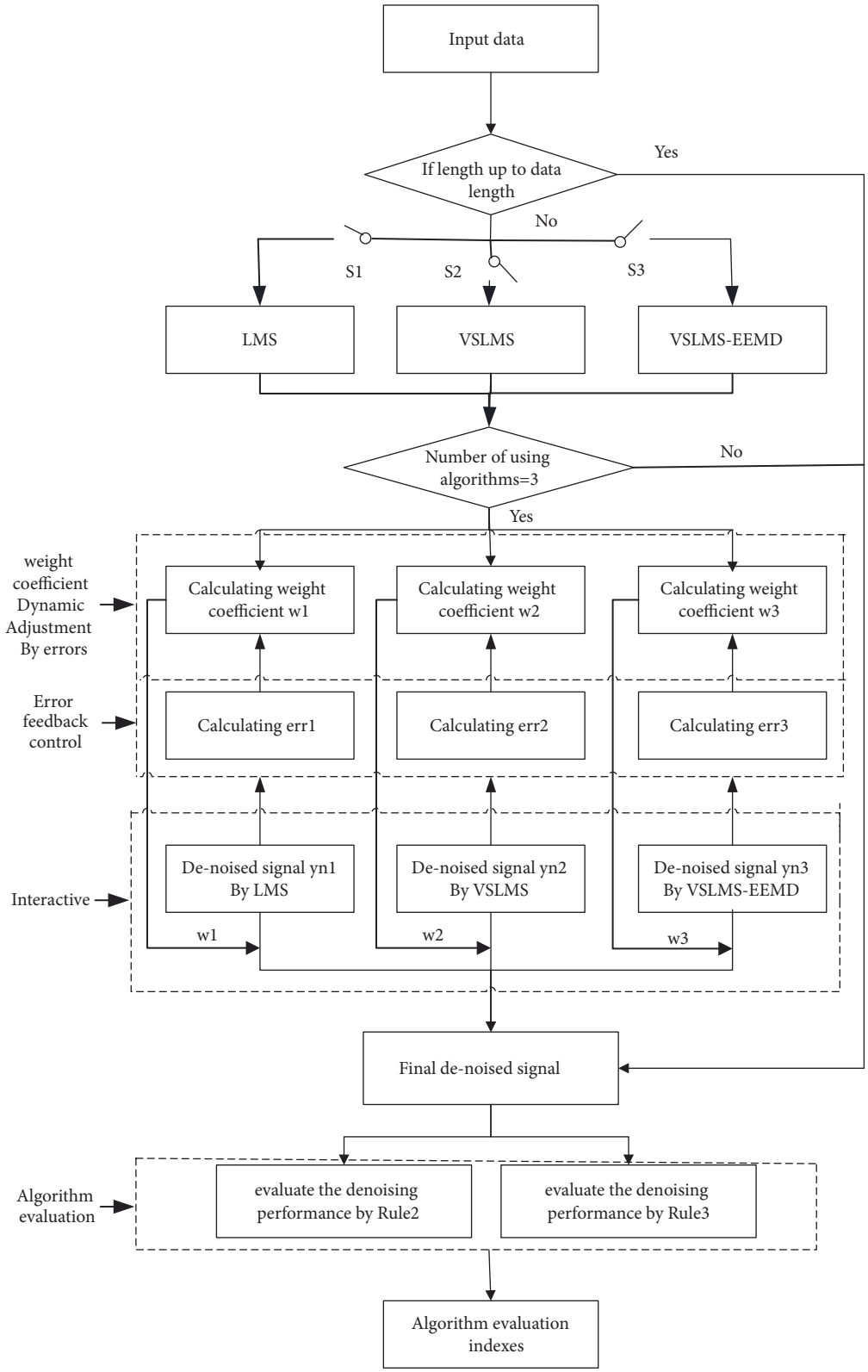
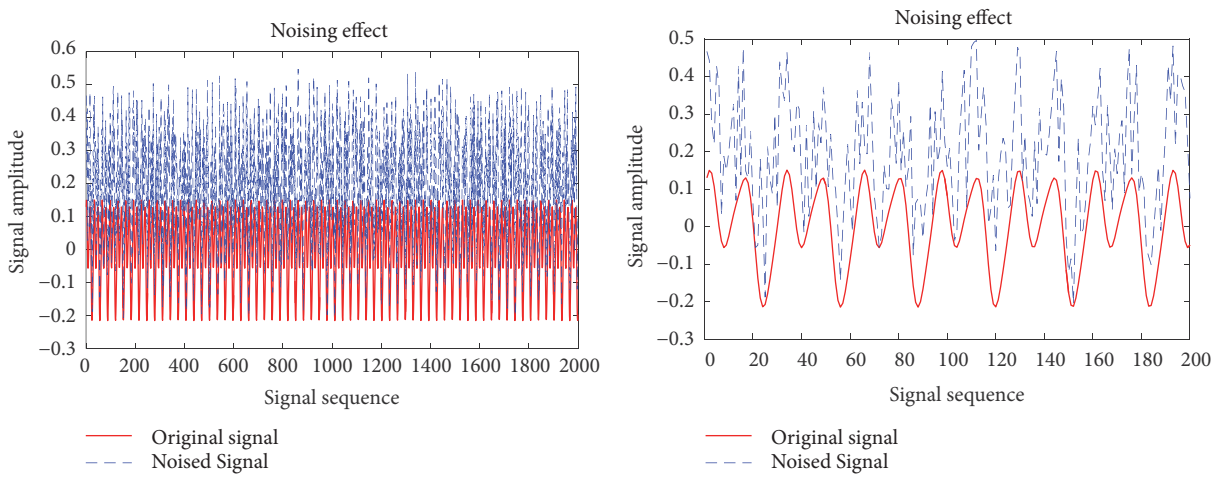


Figure 4: The cooperative denoising flowchart with interactive dynamic adjustment function.



(a) Comparison result of the entire data sets (1-2000)

(b) Refined comparison result of the data sets (1-200)

Figure 5: Comparison result between the original signal and noised signal.

Further, to prove the efficiency and superiority of the improved VSLMS-EEMD and the proposed IDVSLMS-EEMD algorithm, some comparative simulations were done, including LMS, VSLMS, and wavelet with soft threshold combined with EEMD (WTS-EEMD) denoising model in [38, 39]. The corresponding denoising results were shown as Figure 6.

Where, Figures 6(a), 6(c), 6(e), 6(g), and 6(i) illustrate the whole effectiveness of these denoising algorithms. In addition, Figures 6(b), 6(d), 6(f), 6(h), and 6(j) have shown more clear and specific effectiveness by selecting anterior 200 signals. In that case, the effectiveness of the presented algorithm may be better depicted.

To compare the effect of varieties of denoising algorithms, we have selected the Noise Suppression Ratio (NSR) and Signal Distortion Rate (SDR) to evaluate denoising effect, which are defined as follows:

$$\mathrm{NSR} = 1 - \frac{\left[\sum_{i=1}^{N}\left(\hat{S}(n) - s(n)\right)^2\right]^{1/2}}{\left[\sum_{i=1}^{N}\left(\hat{S}(n) - s(n)\right)^2\right]^{1/2}} \tag{28}$$

$$\mathrm{SDR} = \frac{\left[\sum_{i=1}^{N}\left(\hat{S}(n) - s(n)\right)^2\right]^{1/2}}{\left[\sum_{i=1}^{N} s(n)^2\right]^{1/2}} \tag{29}$$

where $\hat{S}(n)$ and $s(n)$ indicate the original signal with noise as well as the denoised signal.

Without loss of the generality, the following rule needs to be noticed.

Rule 4. The larger the NSR is, the smaller the SDR will be. Meanwhile, this also means that the elimination effect of noise is better.

Table 1: Denosing evaluations of LMS, VSLMS, WTS-EEMD, VSLMS-EEMD, and IDVSLMS-EEMD.

Method/parameter	NSR	SDR
LMS	0.7011	0.6266
VSLMS	0.8369	0.3419
WTS-EEMD	0.8603	0.5896
VSLMS-EEMD	0.8644	0.2842
IDVSLMS-EEMD	0.8674	0.2779

Based on the rule, the comparison results are shown in Table 1.

Combining with Figures 6(b), 6(d), 6(f), 6(h), and 6(j), we can know that the denoised curve of IDVSLMS-EEMD algorithm is the smoothest and is closest to original signal. The simulation results and the denoising method parameters in Table 1 illustrated that the denoising effect of LMS, VSLMS, WTS-EEMD, and VSLMS-EEMD is inferior to proposed IDVSLMS-EEMD. Moreover, the improved cooperative denosing algorithm may be provided with the maximum NSR and the minimum SDR. Thus, the denoising effect of the improved algorithm is the best.

In addition, to illustrate the influence of noise, the comparison result of SNR between noised signal and denoised signal is also shown in Table 2.

As seen in Table 2, the effect of the cooperative denosing algorithm is very good. So, after the function testing, this integrated framework may be applied to actual project.

5. Real Experiment Examples

Denosing is the essential premise for further security analysis of stacker in IIoT. To further verify the performance of the proposed algorithm, the real-time simulation signal of stacker under abnormal condition in ASRS, which has been developed and applied in Power Grid of China, was selected to test the denoising performance of the presented algorithm.

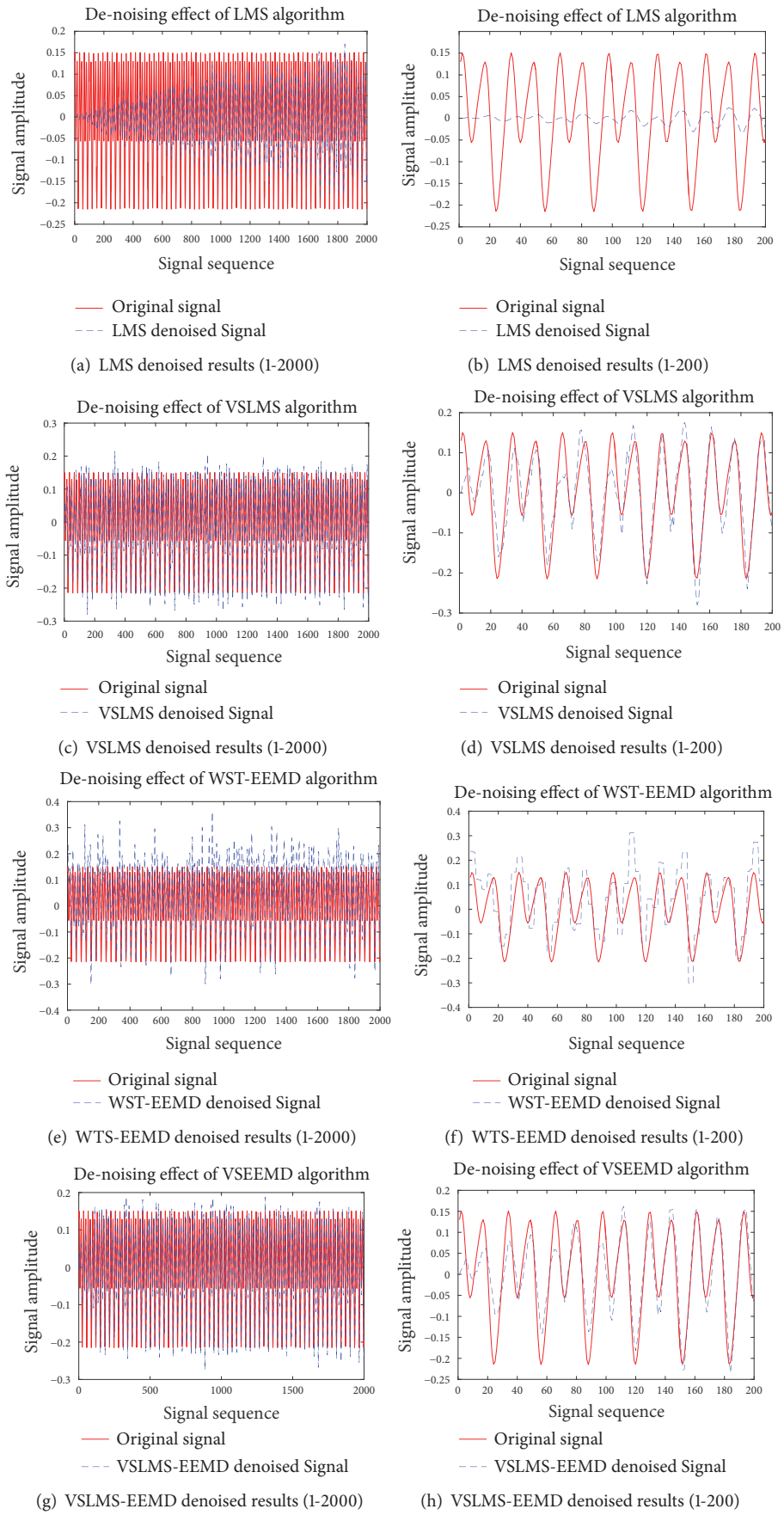
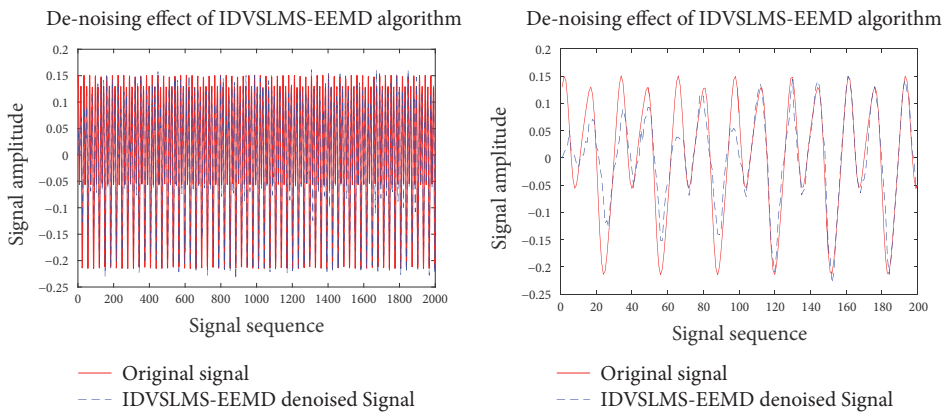


Figure 6: Continued.



(i) IDVSLMS-EEMD denoised results (1-2000) (j) IDVSLMS-EEMD denoised results (1-200)

Figure 6: Denosing simulation results of simulation signal by varieties of denoising algorithms (1-2000).

Figure 7: Simulate rigs of ASRS.

Table 2: SNR of the noised and denoised simulated signal.

Signal/parameter	SNR
Noised signal	-1.1238
Denoised signal	10.1160

The test rig of the prototype systems in IIoT is shown as Figure 7.

The simulation rig of ASRS is constructed and developed according to the real requirements of Power Grid in China. Their main function is to grab, move, and stack goods from one piece of equipment to another. As the crucial equipment of ASRS, the security and the positioning accuracy of stacker will directly affect the data acquisition and the data exchange of the whole IIoT system. In addition, the stacker is driven by motor, so the running state of stacker is directly reflected by the driving vibration signal. In real engineering, the test rig of stacker signal is shown as Figure 8.

In real application, the sampling time is from a.m. 9:03:51 to p.m. 15:04. The column of starting and stopping range is from 0 to 23. The size of the detecting signal is 2000. Then, the comparison results between the original signal and compounded signal with noise measured were simulated by the stacker's running. The results were shown as Figure 9.

Table 3: Simulation results of Rule 1.

Signal/parameter	C
Original signal	0
Noisy signal	0.1424
LMS	0.0823
VSLMS	0.0820
WST-EEMD	0.0810
VSLMS-EEMD	0.0726
IDVSLMS-EEMD	0.0589

Secondly, to further prove the efficiency and superiority of the improved VSLMS-EEMD and the proposed IDVSLMS-EEMD algorithm, some comparative experiments were done. The results of stacker's running signal were simulated by the above relevant denoising algorithms, respectively. The denoising results were shown in Figure 10.

To see more clearly the performance of denoising algorithm, we had selected the anterior 200 signals to refine the display degree of the denoising effect. The results are illustrated as Figure 11.

Figures 11(a)–11(e) highlight the refined display degree and more clearly reveal the difference between the original signal and denoising signal when the nonstationary signal was denoised using different algorithms. As can be seen from the refined illustration in Figure 11, the presented algorithm with interactive dynamic adjustment function approaches accurately high quality.

Moreover, to quantitatively illustrate and assess the difference of denoising effect, we have used the evaluation indexes to compute the evaluated results. These values are listed in Tables 3 and 4.

As measured in Table 3, the denoising absolute error value is minimal when the proposed algorithm with interactive dynamic adjustment function was used to denoise for the running signals of Stacker. Meanwhile, Table 4 shows that NCC of the proposed algorithm is maximum. By the evaluation of Rules 2-3, we know that the proposed algorithm

(a) Driven motor (b) Depression of lower track

Figure 8: Test rig of stacker' signal.

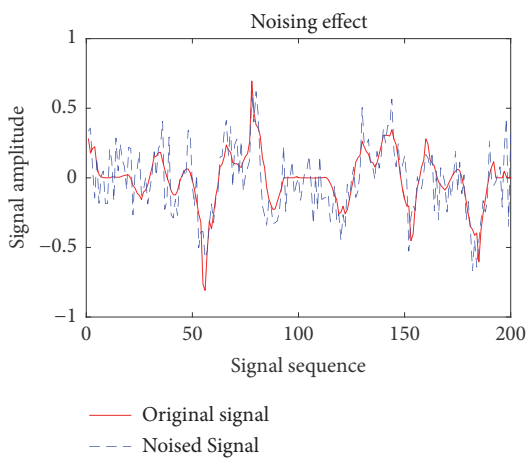


Figure 9: Comparison result between the original signal and noised signal of the stacker running.

Table 4: Simulation results of Rule 2.

Method/parameter	NCC
LMS	0.7500
VSLMS	0.7978
WST-EEMD	0.8005
VSLMS-EEMD	0.8429
IDVSLMS-EEMD	0.8658

with interactive dynamic adjustment function may achieve the desired performance in real systems.

The overall idea here is the same as what we have discussed in the previous simulation examples; the SNR between noised signal and denoised signal was also computed to illustrate the influence of noise for security analysis of stacker in IIoT. The numerical results are shown in Table 5.

Obviously, the SNR is strengthened because the proposed algorithm may obtain and integrate more abundant information compared to traditional methods for security of stacker in real health monitoring of IIoT. That means that the effect of the cooperative denosing algorithm is very good.

Table 5: SNR of the noised and denoised signal for stacker.

Signal/parameter	SNR
Noised signal	-0.7437
Denoised signal	14.6039

The analysis results on the actual examples show that the proposed denoising algorithm may improve the accuracy of denosing to provide higher reliability for security monitoring of stacker in IIoT. That means that our algorithm may be applied to monitoring the security of the devices in the real IIoT.

6. Conclusions

In this paper the cooperative denoising algorithm with interactive dynamic adjustment function was depicted and analyzed based on LMS, VSLMS, and VSLMS–EEMD via the integrated optimization strategies. Meanwhile, some basic theories and corresponding evaluated indexes were also selected and established. The simulation examples and actual examples show the validity and rationality of the proposed algorithm in monitoring the security of real IIoT devices. The main conclusions of our work are listed as follows:

(1) In IIoT system, the original signal is seriously interfered by the surroundings resulting in low SNR. Because of this phenomenon, it is difficult to obtain accurate and reliable features from the confused signals, which has seriously hindered the security analysis, health detection, and the maintenance of IIoT system. Therefore, it is necessary to denoise the nonstationary signal of IIoT.

(2) The shortcomings of traditional EMD algorithm and traditional LMS algorithm with fixed step are considered. To maximize the advantages of LMS, VSLMS, and EEMD, the VSLMS-EEMD denoising algorithm has been constructed. On this basis, a cooperative denoising algorithm with interactive dynamic adjustment function is proposed to further improve the denoising accuracy of VSLMS-EEMD. Meanwhile, the evaluated indexes and rules were designed according to the features of the information for IIoT devices.

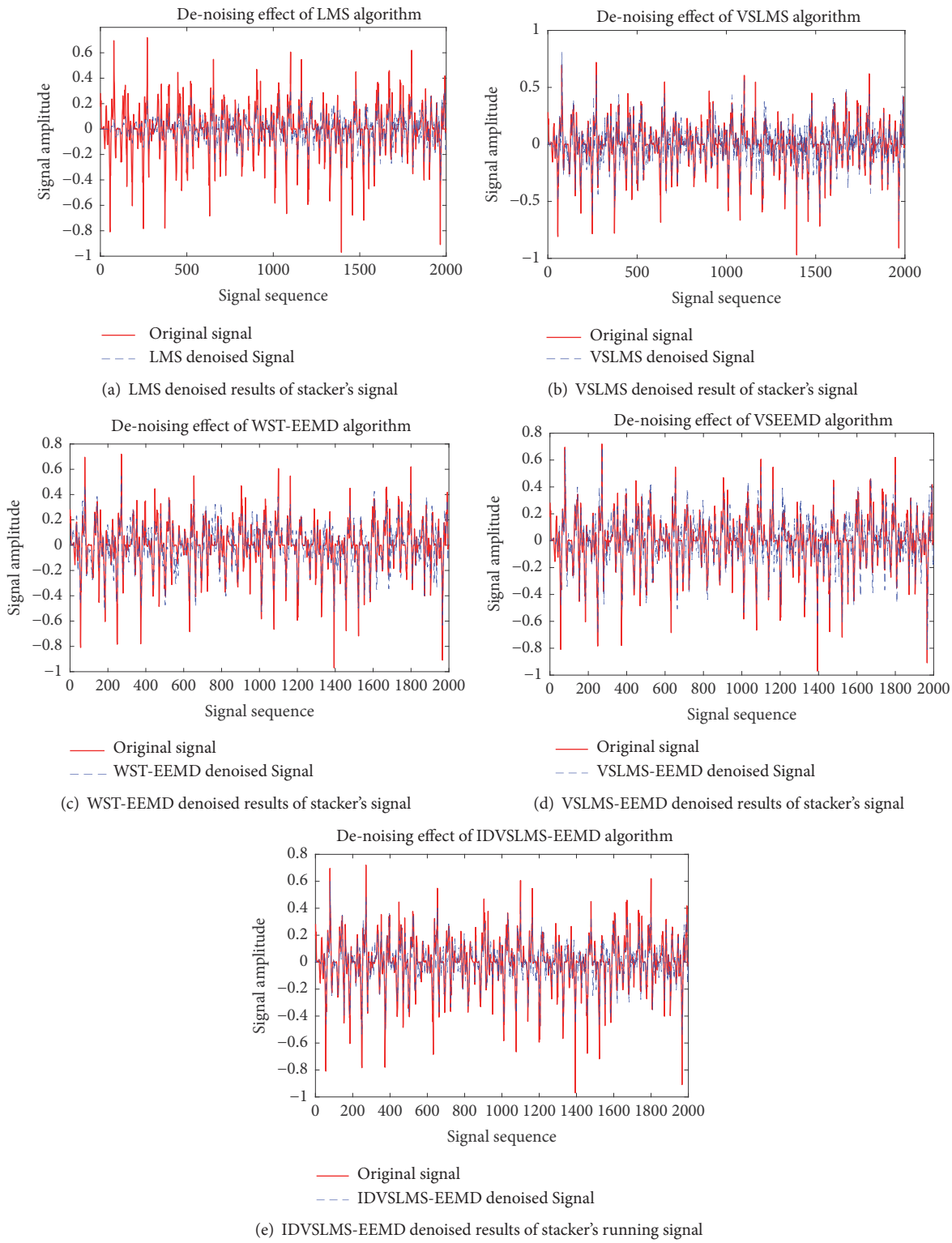
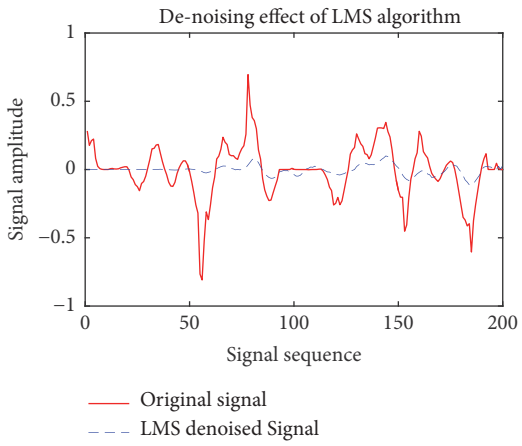
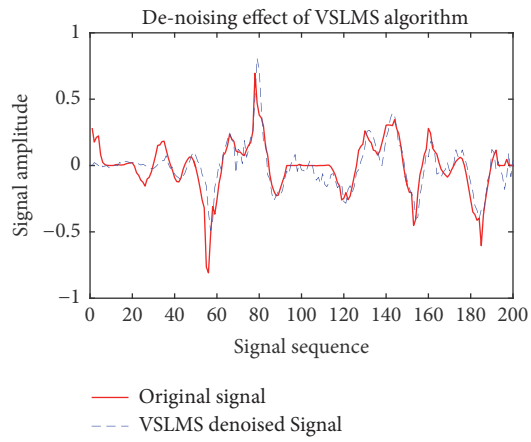


Figure 10: Denosing simulation results of stacker's running signal by using varieties of denoising algorithms (1-2000).

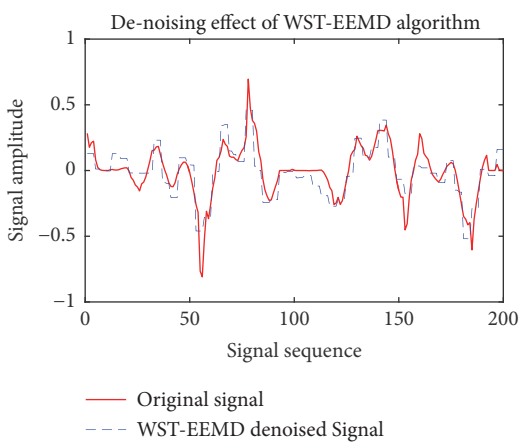
(3) Simulation examples and real data examples were used to implement and verify the efficiency of the proposed algorithm. Moreover, the comparison results were computed via the denoising evaluating indictors (i.e., model and rule). The simulation results show that the new algorithm has a better synchronous precision and security. Compared with the traditional method, the presented method can greatly reduce the noise ratio of security monitoring of IIoT devices.



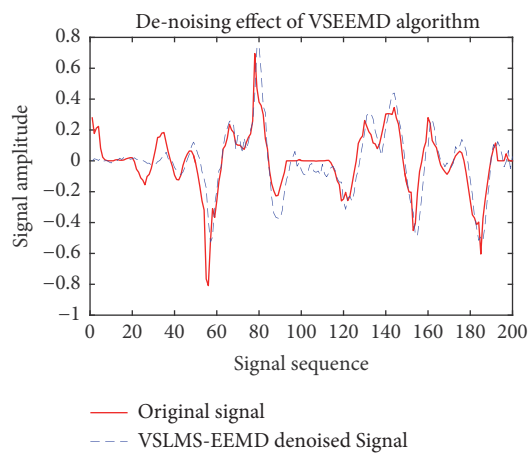
(a) LMS refined denoised effect of stacker's signal



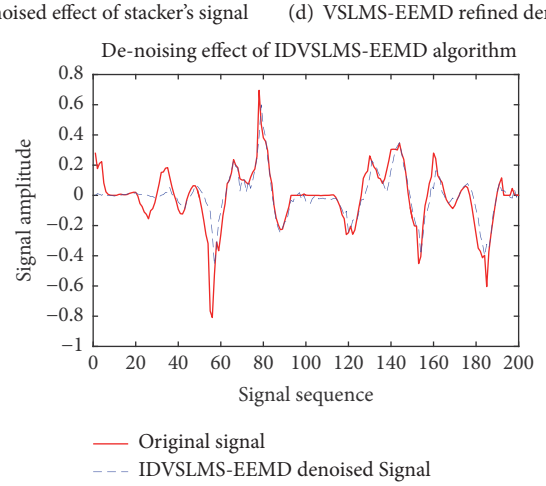
(b) VSLMS refined denoised effect of stacker's signal



(c) WTS-EEMD refined denoised effect of stacker's signal



(d) VSLMS-EEMD refined denoised effect of stacker's signal



(e) IDVSLMS-EEMD refined denoised effect of stacker's signal

Figure 11: Denosing refined simulation chart of stacker's running signal by varieties of denoising algorithms (1-200).

Unfortunately, this cooperative denoising algorithm is only for one or three kinds of denoising algorithms, and no specific design is made for the cooperation of the two algorithms; the weight coefficients ω is calculated by error signal, so its calculation can be further optimized. Due to the limited space, this work will be given in another paper.

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 National Natural Science Foundation of China under Grants 61573076, 61663008, 61703063, and 61004118; the Scientific Research Foundation for the Returned Overseas Chinese Scholars under Grant 2015-49; the Program for Excellent Talents of Chongqing Higher School of China under Grant 2014-18; Chongqing Natural Science Foundation of China under Grant CSTC2017jcyjA1665; Science and Technology Research Project of Chongqing Municipal Education Commission of China under Grants KJ1605002, KJ1705121, and KJ1705139 and KJZD-K201800701; the Program of Chongqing innovation and entrepreneurship for Returned Overseas Scholars of China under Grant cx2018110.

References

[1] L. He, C. Chen, T. Zhang, H. Zhu, and S. Wan, "Wearable depth camera: monocular depth estimation via sparse optimization under weak supervision," *IEEE Access*, vol. 6, pp. 41337–41345, 2018.

[2] S. Wan, Y. Zhang, and J. Chen, "On the construction of data aggregation tree with maximizing lifetime in large-scale wireless sensor networks," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7433–7440, 2016.

[3] P. Zhao, J. Li, F. Zeng et al., "Enabling k-anonymity-based privacy preserving against location injection attacks in continuous LBS queries," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1033–1042, 2018.

[4] L. Qi, R. Wang, S. Li et al., "Time-aware distributed service recommendation with privacy-preservation," *Information Sciences*, vol. 480, pp. 354–364, 2019.

[5] L. Qi, W. Dou, W. Wang et al., "Dynamic mobile crowdsourcing selection for electricity load forecasting," *IEEE Access*, vol. 6, pp. 46926–46937, 2018.

[6] L. Qi, X. Zhang, W. Dou et al., "A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment," *Future Generation Computer Systems*, vol. 88, pp. 636–643, 2018.

[7] W. Gong, L. Qi, and Y. Xu, "Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 3075849, 8 pages, 2018.

[8] C. Yan, X. Cui, L. Qi et al., "Privacy-aware data publishing and integration for collaborative service recommendation," *IEEE Access*, vol. 6, pp. 43021–43028, 2018.

[9] D. Huang, M. Lin, and L. Ke, "A new cooperative anomaly detection method for stacker running track of automated storage and retrieval system in industrial environment," *Journal of Control Science and Engineering*, vol. 2018, Article ID 1938490, 12 pages, 2018.

[10] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[11] Y. Xu, L. Qi, W. Dou, and J. Yu, "Privacy-preserving and scalable service recommendation based on SimHash in a distributed cloud environment," *Complexity*, vol. 2017, Article ID 3437854, 9 pages, 2017.

[12] A. Gonzalez-Moreno, S. Aurtenetxe, M.-E. Lopez-Garcia, F. del Pozo, F. Maestu, and A. Nevado, "Signal-to-noise ratio of the MEG signal after preprocessing," *Journal of Neuroscience Methods*, vol. 222, no. 1, pp. 56–61, 2014.

[13] Z. Ge, Z. Song, S. X. Ding, and B. Huang, "Data mining and analytics in the process industry: the role of machine learning," *IEEE Access*, vol. 5, pp. 20590–20616, 2017.

[14] J.-B. Qiu, Z.-G. Ying, and L. H. Yam, "New modal synthesis technique using mixed modes," *AIAA Journal*, vol. 35, no. 12, pp. 1869–1875, 2015.

[15] F. Romero, F. J. Alonso, J. Cubero, and G. Galán-Marín, "An automatic SSA-based de-noising and smoothing technique for surface electromyography signals," *Biomedical Signal Processing and Control*, vol. 18, pp. 317–324, 2015.

[16] H. Mahmoud, B. Sofiane, T. Jérémy et al., "Combination of canonical correlation analysis and empirical mode decomposition applied to denoising the labor electro hysterogram," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 9, pp. 2441–2447, 2011.

[17] C. Ruan, D. Zhao, W. Jia et al., "A new image denoising method by combining WT with ICA," *Mathematical Problems in Engineering*, vol. 2015, Article ID 582640, 10 pages, 2015.

[18] P. Xiong, H. Wang, M. Liu, S. Zhou, Z. Hou, and X. Liu, "ECG signal enhancement based on improved denoising auto-encoder," *Engineering Applications of Artificial Intelligence*, vol. 52, pp. 194–202, 2016.

[19] X. Hui and Y. Rengang, "Power quality disturbance detection method using wavelet packet transform based on de-noising scheme," *Proceedings of the CSEE*, vol. 24, no. 3, pp. 85–90, 2004.

[20] M. Bitenc, D. S. Kieffer, and K. Khoshelham, "Evaluation of wavelet and non-local mean denoising of terrestrial laser scanning data for small-scale joint roughness estimation," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLI-B3, pp. 181–186, 2016.

[21] W. Lu, L. Jianxun, and H. Ting, "Denoising algorithm of pulsar signal based on EMD with kurtosis test window," *Journal of Systems Engineering and Electronics*, vol. 39, no. 6, pp. 1208–1214, 2017.

[22] D. Safieddine, A. Kachenoura, L. Albera et al., "Removal of muscle artifact from EEG data: comparison between stochastic (ICA and CCA) and deterministic (EMD and wavelet-based) approaches," *EURASIP Journal on Advances in Signal Processing*, vol. 1, article 127, 2012.

[23] C. Guo, Y. Wen, P. Li, and J. Wen, "Adaptive noise cancellation based on EMD in water-supply pipeline leak detection," *Measurement*, vol. 79, pp. 188–197, 2016.

[24] X. Shang, X. Li, A. Morales-Esteban et al., "Enhancing microseismic P-phase arrival picking: EMD—Cosine function based denoising with application to the AIC picker," *Journal of Applied Geophysics*, vol. 9, no. 12, 2017.

[25] K. He, L. Yu, and L. Tang, "Electricity price forecasting with a BED (Bivariate EMD Denoising) methodology," *Energy*, vol. 91, pp. 601–609, 2015.

[26] Z. Erlei, Z. Lesson, and D. Su, "Power quality disturbance detection based on EEMD threshold denoising," *East China Electric Power*, vol. 41, no. 10, pp. 2090–2094, 2013.

[27] R. Xiaomin and L. Dongxin, "Adaptive filtering for speech based on multi-rate LMS algorithm," *Foreign Electronic Measurement Technology*, vol. 37, no. 9, pp. 68–73, 2018.

[28] Z. Hongjun, "Wavelet model and filter method for adaptive filtering of non-stationary signals," *Journal of Mechanical Engineering*, vol. 42, no. 8, pp. 201–204, 2006.

[29] Z. Min, D. Zhishan, and G. Baoliang, "Application of CEEMDAN Combined with LMS Algorithm in Signal De-noising of Bearings," *Noise and Vibration Control*, vol. 38, no. 2, pp. 144–149, 2018.

[30] Z. Guinan, L. Zhigang, X. Chuan et al., "Characteristic analysis and frequency estimation on voltage fluctuation of electrified railway considering multi-frequency modulation," *Power System Technology*, vol. 41, no. 1, pp. 251–257, 2017.

[31] K. Yu, T. R. Lin, and J. W. Tan, "A bearing fault diagnosis technique based on singular values of EEMD spatial condition matrix and Gath-Geva clustering," *Applied Acoustics*, vol. 121, pp. 33–45, 2017.

[32] X. Yu, L. Xu, J.-Q. Mo, and X.-Y. Lü, "Raman spectroscopy denoising based on EEMD combined with VS-LMS algorithm," *Optoelectronics Letters*, vol. 12, no. 1, pp. 16–19, 2016.

[33] Z. Lanyong, W. Bangmin, L. Sheng et al., "A novel variable step-size adaptive interference cancellation algorithm," *Chinese Journal of Electronics*, vol. 45, no. 2, pp. 321–327, 2017.

[34] F.-L. Li, Y.-H. Zhou, and F. Tong, "Two-parameter adjustable underwater acoustic channel equalization algorithm," *Acta Armamentarii*, vol. 34, no. 6, pp. 726–731, 2013.

[35] Z. Zhu, X. Gao, L. Cao, D. Pan, Y. Cai, and Y. Zhu, "Analysis on the adaptive filter based on LMS algorithm," *Optik - International Journal for Light and Electron Optics*, vol. 127, no. 11, pp. 4698–4704, 2016.

[36] J. Tang, Y.-L. Li, Y.-B. Xie et al., "A new parameter for evaluating waveform distortion of partial discharge signals after denoising," *Journal of Chongqing University*, vol. 32, no. 3, pp. 252–256, 2009 (Chinese).

[37] C. Jianhua, X. Xiao et al., "De-noising of magneto telluric signal in the ore concentration area based on combination filter," *Journal of Jilin University*, vol. 47, no. 3, pp. 874–883, 2017.

[38] Y. Hongyu, Z. Yue, and Y. Xiaolin, "Wavelet de-noising method for blasting vibration signals of tunnel after EEMD decomposition," *Railway Engineering*, vol. 7, no. 7, pp. 83–86, 2018.

[39] S. Wan, Y. Zhao, T. Wang, Z. Gu, Q. H. Abbasi, and K. R. Choo, "Multi-dimensional data indexing and range query processing via Voronoi diagram for internet of things," *Future Generation Computer Systems*, vol. 91, pp. 382–391, 2019.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 1635081, 10 pages
https://doi.org/10.1155/2018/1635081

Research Article

Application of Temperature Prediction Based on Neural Network in Intrusion Detection of IoT

Xuefei Liu,[1] Chao Zhang,[2] Pingzeng Liu,[1] Maoling Yan,[1] Baojia Wang,[1] Jianyong Zhang,[1] and Russell Higgs[3]

[1] *Shandong Agricultural University, College of Information Science and Engineering, Tai'an 271000, China*
[2] *Agricultural Big-Data Research Center of Shandong Agricultural University, Tai'an 271000, China*
[3] *School of Mathematics & Statistics, University College Dublin (UCD), Belfield, Dublin 4, Ireland*

Correspondence should be addressed to Chao Zhang; zhangch@sdau.edu.cn

Received 1 August 2018; Accepted 11 October 2018; Published 18 December 2018

Guest Editor: Xuyun Zhang

The security of network information in the Internet of Things faces enormous challenges. The traditional security defense mechanism is passive and certain loopholes. Intrusion detection can carry out network security monitoring and take corresponding measures actively. The neural network-based intrusion detection technology has specific adaptive capabilities, which can adapt to complex network environments and provide high intrusion detection rate. For the sake of solving the problem that the farmland Internet of Things is very vulnerable to invasion, we use a neural network to construct the farmland Internet of Things intrusion detection system to detect anomalous intrusion. In this study, the temperature of the IoT acquisition system is taken as the research object. It has divided which into different time granularities for feature analysis. We provide the detection standard for the data training detection module by comparing the traditional ARIMA and neural network methods. Its results show that the information on the temperature series is abundant. In addition, the neural network can predict the temperature sequence of varying time granularities better and ensure a small prediction error. It provides the testing standard for the construction of an intrusion detection system of the Internet of Things.

1. Introduction

The big data of agricultural production is based on the continuous observation of the environmental elements of the farmland. It integrates massive multisource and multiscale information [1, 2]. Relying on the perception terminal of the Internet of Things (the following are expressed in IoT) to collect farmland environmental information has been widely used [3–6]. The Internet of Things sensor terminal integrates various sensors, such as meteorology, water and salt, soil, and groundwater, and combines ground and air sensor cluster to collect and transmit all kinds of data in real time. Sensor nodes in the Internet of Things are usually distributed in an unattended environment, which is vulnerable to external malicious attacks and requires high security for nodes. The perspectives of attack mode and intrusion behavior are the two main ways to influence the normal routing forwarding of nodes and to consume node resources [7–9]. Although the existing intrusion detection technology for wireless sensor networks can resist system attacks to a great extent, there are also some shortcomings [6], such as high false alarm rate of intrusion detection system, the unstable speed of intrusion detection system, and a previous update of attack feature library. With the development of artificial intelligence, neural networks have attracted much attention because of their ability of self-learning and searching for optimal solutions at high speed. Using the principle and technology of neural network to realize intrusion detection has become a new direction in the development of intrusion detection technology in recent years. It has emulated the theory and method of the biological information processing mode to obtain the intelligent information processing function [10]. The intrusion detection system based on neural network belongs to the category of abnormal intrusion detection, including data acquisition module, data training, and detection module and a response module. The most essential and most important feature of the

neural network algorithm is the data training and detection module. In this study, the research on the data training and detection module is carried out. The prediction data is added to the data training and detection module. By using the better prediction method, the accurate prediction of the evidence is realized [11], the characteristics of the collection information are extracted, the internal association rules of the collection information are excavated, and the detection standard for the subsequent accurate intrusion detection is provided.

At present, the prediction of farmland climate mainly involves indicators such as rainfall, humidity, wind speed, and soil temperature. Among them, Ashok Mishra adopted the SWAP crop model which run for the rice and two scenarios and realized the rainfall forecasting. It was confirmed that the accurate prediction of rainfall could save rice irrigation water [12]. I. Białobrzewski used neural network modeling and STATISTICA method to predict relative air humidity and found that neural network prediction results are more accurate [13]. To realize mean hourly wind speed modeling prediction, R.E. Abdel-Aal using GMDH-based abductive networks verified abductive networks predictions have better predictive effects than neural networks [14]. Z Gao et al. using the revised force-restore method to predict the soil temperatures in naturally occurring nonuniform soil [15]. In summary, most of the temperature prediction is aimed at atmospheric temperature prediction, but the generalized climate and field microclimate have different climatic characteristics. Agricultural microclimate research is of great significance to the development of agricultural production, and farmland temperature is critical to crop production. Therefore, the temperature of agricultural microclimate is taken as the research variable [16]. Therefore, the temperature in the agricultural microclimate is used as the research variable. Moreover, the suitable forecast model has been chosen to predict the farmland temperature to provide some data guidance for agricultural production.

Although there are many studies on the prediction of atmospheric temperature, most of the research is based on the projection of the temperature according to the average annual temperature, the monthly average temperature, or the average daily temperature. Time has a significant influence on the prediction results, so the time factor should be taken into account in the prediction [17]. Regarding atmospheric temperature prediction, Changjun used the Winters method to predict the average monthly temperature from June to August in summer [18]. Zhang Yingchun used the artificial neural network learning algorithm to predict monthly average temperature data in the Karamay Desert [19]. B. Ustaoglu used the three artificial neural network algorithms (RBF, FFBP, and GRNN) to predict the daily average, maximum, and minimum temperature series [20]. For the prediction of greenhouse temperature, Zuo Zhiyu and others established the ARMA 1-step prediction model using time series analysis method and realized the prediction of greenhouse temperature during the next period with the acquisition time unit of 30 minutes [21]. Zhang Xiaodan using parameter optimization support vector machine to predict and model the daytime temperature sequence in the greenhouse, the time interval of data is one hour [22]. HuihuiYu et al. used the improved PSO to optimize the LSSVM to predict the temperature series collected in the solar greenhouse. A temperature sequence with a time granularity of 6 hours is predicted by contrasting different methods [23]. It can be perceived that most of the forecast of the climate temperature is based on the monthly and daily forecasting units, and the time granularity of the greenhouse temperature forecast is mostly in groups of hours. However, the greenhouse temperature is controlled more manually than in the farmland microclimate. Therefore, the temperature of farmland microclimate is taken as the research object, the temperature time series data is organized, different time granularities are divided, and the trend characteristics are analyzed. The traditional time series analysis method and the neural network prediction method are used to predict the different time granularity, respectively. Based on the feature extraction and prediction of the collected data, we construct the association rule base of intrusion detection and update the rule base of the detected intrusion information and achieve the goal of dynamic learning.

2. Method

The Internet of Things can make all kinds of integrated embedded sensors work together, monitor, perceive, and collect the information of various environment or monitoring objects in real time by using all sorts of sensors to work together. The embedded system analyzes the data, and through adaptive wireless network communication, the collection and perception of various signals in the physical world are realized. However, because many sensors are distributed in relatively open and unsupervised places, it is easy to be attacked from outside. Therefore, the security of the Internet of Things will become an important research direction. The power supply of the Internet of Things is limited, the communication ability is limited, and the calculation and storage are also finite. In this case, how to establish an effective security system, detect all kinds of intrusion and malicious attacks, and ensure the reliability of the Internet of Things is particularly important [24]. From the point of view of security technology, the technologies for the security of the Internet of Things include authentication technology to ensure its own security, key establishment and distribution mechanism to ensure secure transmission, and data encryption to ensure the security of the data itself [25]. These technologies are passive precautions and cannot detect intrusions actively. Intrusion detection based on Internet of Things security technology is a proactive defense technology. By monitoring the state, behavior, and usage of the whole network and system, the intrusion detection system detects the primary use of the system users and the attempt by the external invaders to invade the network or system. It can not only identify the intrusion from the outside but also monitor the illegal behavior of the internal users [26]. Zhang Jianfeng et al. have carried out a series of discussions on the intrusion detection technology of WSN and introduced the application of neural network to intrusion detection technology in the Internet of Things [27]. By dividing the intrusion detection system into different modules, the neural

network is applied to each module to realize the intelligent and dynamic detection of the intrusion detection system. Through the feature extraction and prediction of the collected data, the predefined dataset rules and attack data set rules are trained to train the neural network module to provide a dynamic rule base for the intrusion detection system.

Before analyzing and forecasting meteorological data, it is necessary to examine the characteristics of the sequence and grasp the changing rule of the data. The main characteristics of meteorological data are seasonal analysis and periodic analysis. Among them, the seasonal study is the analysis of the climate differences between different seasons, the amplitude of commonly used climatic elements, and the large magnitude indicating strong season. Through seasonal analysis, we can understand the seasonal changes of data and help people to conduct seasonal decomposition according to needs. Moreover, the periodic report is to explore whether a variable shows an inevitable trend of change with time [28]. The relatively long periodic patterns of timescale include annual cyclical trend, seasonal trend, cyclical trend, relatively slight quarterly periodic trend, weekly cyclical trend, even shorter day and hour cycle trend. The object of this study is the temperature sequence. The feature analysis of the series is helpful to understand the variation of the chain and can also be used to distinguish the different prediction time granularity.

Temperature series are time series data, and the commonly used methods of analyzing time series data were divided into traditional time series prediction model and data-driven time series forecasting model [29]. The conventional time series prediction models mainly include ARMA (AR, MA), ARIMA, improved time series model Threshold Auto-Regressive (TAR), Vector Auto-Regression (VAR), Auto-Regressive Conditional Heteroscedasticity (ARCH), and Generalized Auto-Regressive Conditional Heteroscedasticity (GARCH). The use of ARMA model must satisfy the self-correlation of the parameters, and the autocorrelation coefficient must be higher than 0.5, and the model can only be used to predict the economic phenomena related to its early stage. The main problem that the TAR model has for meteorological time series prediction is that it requires a lot of complicated optimization work in the modeling process [30]. The VAR model can be viewed as a multivariate extension of the AR model. Using the VAR model must eliminate the periodic nonstationary nature of the variables [31]. Both ARCH and GARCH processes are new stochastic processes that show the variation of the variance of random variables over time, but not all-time series data exhibit heteroskedasticity [32, 33]. The ARIMA model only requires endogenous variables without resorting to other exogenous variables. Data-driven time series prediction methods include chaotic time series forecasting, gray time series forecasting, fuzzy logic time series forecasting, neural network time series forecasting, and SVM time series forecasting and so on. When selecting the chaotic time series forecasting model, the specific characteristics of the time series should be analyzed to grasp the nature of the chaotic precursors. Gray prediction still needs improvement regarding grey measure, sequence operator, correlation measure, residual error correction, etc. Fuzzy time series has the problems of the quantitative level of fuzzy inference, prediction accuracy, and prior knowledge dependent on specific issues. SVM is challenging to implement for large-scale training samples, and the speed of operation needs to be improved. Moreover, the neural network has better nonlinear mapping ability, generalization ability, and fault tolerance. Based on the above analysis, this experiment selected the ARIMA model in traditional time series analysis and the data-driven neural network model to predict the farmland temperature.

ARIMA. ARIMA model only needs endogenous variables and does not need to use other exogenous variables. The use of ARIMA model needs to satisfy that time series data must be stable. Moreover, the model can capture the linear relationship in essence and cannot capture the nonlinear relationship.

Step 1. To test the stability of the original sequence, if the p-value of the nonstationary test is more than 0.05, the different treatment should be continued at this time, and then the stability test after the difference processing is carried out, if the sequence is stationary, the first order difference is stable. If nonstationary, the most two-order difference stationary test is carried out. If the two-order difference post is nonstationary, the sequence is a nonstationary sequence, and it is not suitable for the next step prediction.

Step 2. According to the recognition rule of time series model, the corresponding model is established. If the partial correlation function of stationary sequence is truncated and the autocorrelation function is tailed, it can be concluded that the sequence is suitable for AR model; if the partial correlation function of stationary sequence is tailed and the autocorrelation function is truncated, it can be concluded that the sequence is suitable for MA model; if the partial correlation function and autocorrelation function of stationary sequence are tailed, then the sequence is suitable for MA model. Sequences are suitable for ARMA models. (Truncation is the property that the autocorrelation function (ACF) or partial autocorrelation function (PACF) of a time series is zero after a certain order (such as the PACF of AR); trailing is the property that ACF or PACF is not zero after a certain order (such as the ACF of AR).)

Step 3. Carry out parameter estimation and test whether it has statistical significance.

Step 4. The hypothesis test is used to diagnose whether the residual sequence is white noise.

Step 5. Predictive analysis was performed using the tested models.

Levenberg-Marquardt Algorithm. The Levenberg-Marquardt algorithm is the most widely used nonlinear least squares algorithm [34]. It is the use of gradient to find the maximum (small) values of the algorithm. The goal of the algorithm is to the function relation $y = f(p)$, given $f(\cdot)$ and Noise-containing observation victory, estimates. Calculation steps are as follows.

Step 1. Take the initial point p_0, terminate the control constant ε_0, and calculate $\varepsilon_0 = \left|\left|y - f(p_0)\right|\right| k := 0, \lambda_0 = 10^{-3}, v = 10 (v > 1$ is OK).

Step 2. Calculate the Jacobi matrix J_k, calculate $\overline{N}_k = J_k^T J_k + \lambda_k I$, and construct an incremental normal equation $\overline{N}_k \cdot \delta_k = J_k^T \varepsilon_k$.

Step 3. Solve delta normal equation to obtain δ_k.

(1) If $\left|\left|y - f(p_k + \delta_k)\right|\right| < \varepsilon_k$, make $p_{k+1} = p_k + \delta_k$, and if $\|\delta_k\| < \varepsilon$, then stop the iteration, output the result, otherwise make $\lambda_{k+1} = \lambda_k / v$, and go to Step 2.

(2) If $\left|\left|y - f(p_k + \delta_k)\right|\right| \geq \varepsilon_k$, make $\lambda_{k+1} = \lambda_k \cdot v$, resolve the normal equation to obtain δ_k, and return to 1.

3. Materials

The primary data sources and temperature data collected by the automatic acquisition equipment of the Internet of Things are introduced, and the data are pretreated and analyzed.

Monitoring Data. The real-time sensing system of dynamic farmland information based on the Internet of Things broke through significant real-time problems, such as real-time dynamic detection of salt, alkali, water, rapid self-diagnosis of equipment faults, and online automatic real-time warning. The information database realizes data receiving, cleaning, storage, integration, and sharing, effectively improves the authenticity and reliability of the collected data, and provides a useful data service foundation for subsequent data mining and precision agriculture [35]. The data are divided into two groups. One group is of Dongying city meteorological station air temperature data, which is every 3 hours for a sampling frequency. We selected the data from 2014-2017, a total of 11680. The second part was based on IoT equipment acquisition in Dongying, which is collected one hour at a time. We choose the data for the whole year of 2016, with a total of 8,784 data.

Data Feature Analysis. Before data analysis, two groups of data are preprocessed to fill in missing values and smooth noise data, identify and delete outliers and resolve inconsistencies, and eliminate duplicate data. Data is transformed into data mining form by smoothing and normalizing.

Annual Statistical Variation. Four-year overall air temperature changes are obtained by plotting the farmland air temperature for 2014-2017, as shown in Figure 1.

It can be seen from Figure 1 that the curve of farmland air temperature changes in the region is similar to the function curve of $|\sin(x)|$ $(x \in (0, \pi))$. The changing trend of air temperature in a year is firstly increased and then decreased. The months with the highest temperatures occur every year in the three months from June to August, the lowest temperatures in January, February, and December.

In order to compare and analyze the difference in air temperature spacing, we selected daily maximum temperature and daily minimum temperature of the most substantial temperature change in September (which is 0.528), the smallest change in July (which is 0.270) and the general temperature changes in April (0.421) and December (0.393) as the object of study, their changing trends are shown in Figure 2.

Table 1: Stability test for 2014-2017 years.

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-17.9471	0
Test critical values	1% level	-3.98354	—
	5% level	-3.42225	—
	10% level	-3.13398	—
R-squared	0.525219	Mean dependent	0.007844
Adjusted R-squared	0.521252	S.D. dependent	3.172735

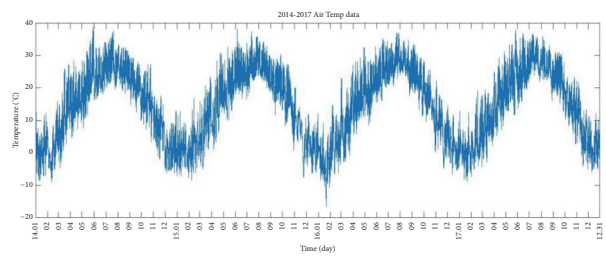


Figure 1: The changing trend of air temperature in 2014-2017.

The daily maximum and minimum temperature gaps in July are smaller than those in September which are evenly distributed. The highest and lowest temperature curves in December are distributed at 0°C, and the temperature gap is minimal on the 22nd day.

Diurnal Temperature Change. The daily variation data for each month were obtained by the statistical mean of 24 hours of daily temperature in each month.

By analyzing Figure 3, we can find that the daily temperature trends are similar, and the temperature is higher at 11:00-16:00 daily, at 22 o'clock the next day 4 o'clock to achieve the lowest. The mean daily temperature in January was the lowest and the highest in July. In September, the temperature difference between day and night was tremendous, while that of July was the smallest.

Stability Test of Air Temperature Time Series. Before ARIMA predictive modeling, we need to test the stability of the data. Therefore, the stationary test of time series of air temperature in 2014-2017 is obtained by Table 1. Time series is first-order differential stationary, that is, the temperature time series is stationary.

Fitting Results and Analysis. The fitting of the average daily temperature in 2015 and 2016 can be obtained as follows.

The equation of the fitting curve for two years is $f(x) = a1 * \sin(b1 * x + c1) + a2 * \sin(b2 * x + c2)$. Coefficients (with 95% confidence bounds) are as shown in Table 2 and Figure 4.

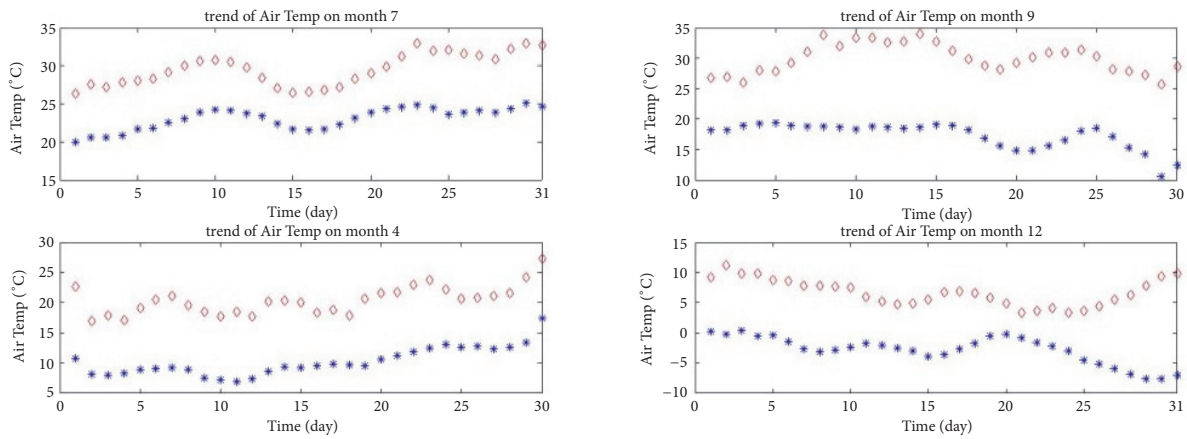


Figure 2: The maximum and minimum temperature change of months.

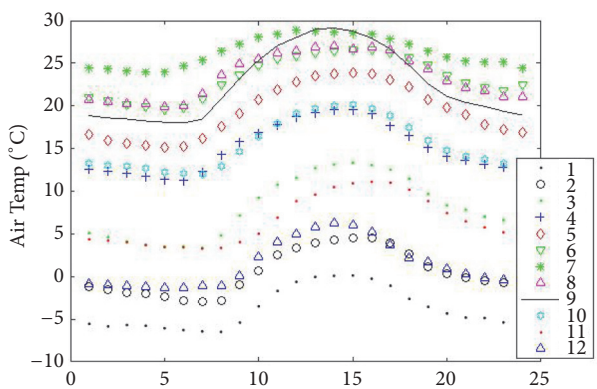


Figure 3: The daily temperature change curve of every month in 2016.

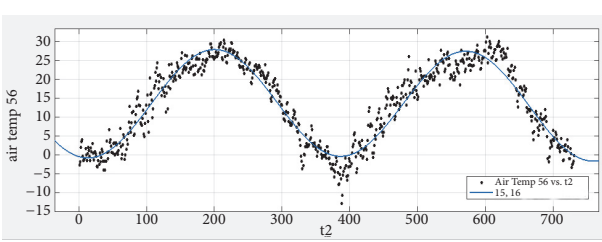


Figure 4: Air temperature fitting.

Because the fitting curve is the sine function, the mean squared error of the fitting is 3.201 and R^2 is 0.9067. There is no discernible trend in the short term, but a certain periodicity. If using traditional time series forecasting may not get the ideal effect, but the neural network has good learning ability, in the sequence some advantages can be predicted.

4. Modeling

Using ARIMA model and the L-M algorithm model, respectively, for modeling, we obtain the experimental results.

Table 2: Parameters of the fitting equation.

Parameter	a1	b1	c1	a2	b2	c2
Value	13.78	0.2007	1.642	14.11	3.572	-1.906

4.1. Data Group. The temperature data of two groups with different time granularity were divided into a training set, a verification set, and a test set, and the proportion was 0.7:0.15:0.15. First, the average monthly temperature of 2014-2017 years is modeled and predicted, and the effect of temperature prediction [36] with time granularity is analyzed. Then the temperature data of the time granularity of 3 hours are modeled and predicted. We selected the temperature data in 2017 to verify the model and choose the best model. Finally, the data for the last week of 2016 was forecasted by the time granularity of 1 hour.

4.2. Model Construction. Two forecasting methods were used in the experiment to predict the farmland temperature, in which the prediction experiment of ARIMA should first carry out the stability test. When the time series is stable, the order forecast is carried out. Before using the neural network for prediction, clean the data, format the input variable and output variable, and divide the data set into proportions. Through the network training, the optimization is continuously optimized until the best state is reached. The execution steps of the two prediction methods were shown in Figures 5 and 6.

Figures 5 and 6 show the prediction process of the two modeling methods. It can be seen that different forecasting methods require different data, the ARIMA model needs the higher stability of data, and the neural network does not need data stability. The critical step of the ARIMA model is to solve the stationarity of data and determine the order of the model, and the key to the neural network lies in the optimization model.

4.3. Results. The model training is realized by the Levenberg-Marquardt algorithm. Besides, the network was trained

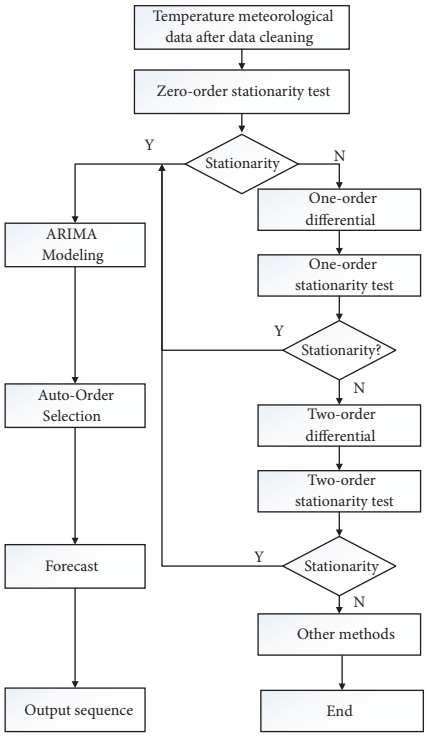


Figure 5: ARIMA forecast construction.

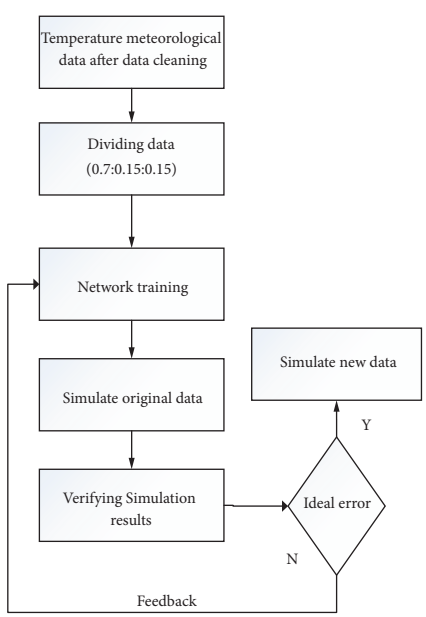


Figure 6: The neural network forecast construction.

according to the sample input vector, target vector, and hidden layer nodes and delay number parameters of the preset training network. The error autocorrelation was used to judge the training network whether it is optimal. Moreover, the model was continuously optimized until the autocorrelation coefficient of the error reaches the optimal range.

By setting the time granularity for months, we forecast the monthly average temperature. The results are as shown in Figure 7.

The MSE diagram shows the variation of the mean square error of training data, validation data, and test data in different training periods. The overall trend of the three curves is similar. The best state is at sixth times, at which the mean square error of the test data is minimized. In the training state graph, MU first dropped and then rose, then fell to 10^{-2}, and remained stationary, which indicates that the model had reached its optimal state. The regression diagram describes the regression of the three datasets. Most of the data are in the vicinity of the diagonal, indicating that the regression works well.

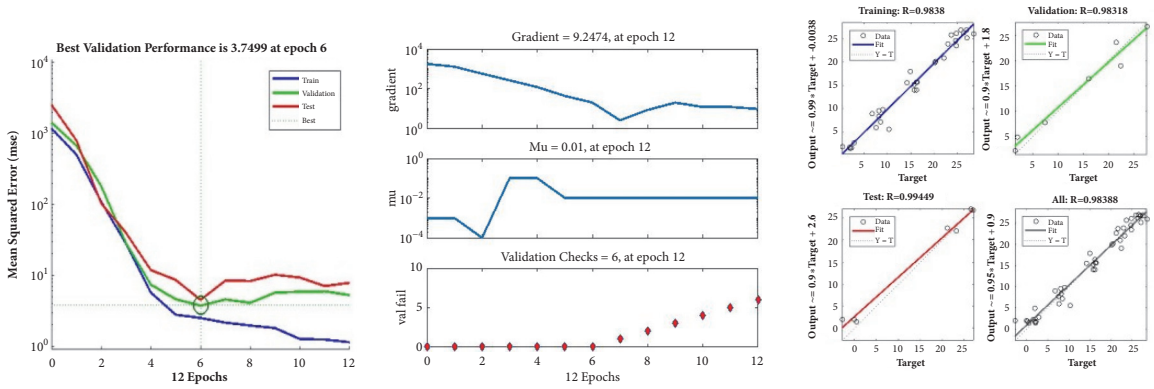
The upper half of the graph (Figure 8(a)) is the response of the output element to the time series, and the lower part is the output error, whose range is (-5, 5), which indicates that the error is small. It can be seen in the chart (Figure 8(b)) that except for the 0 order autocorrelation, the correlation coefficients should not exceed the upper and lower confidence intervals. Some of the charts in the confidence interval indicate that the prediction results are not very ideal, and the reason is that the amount of data is relatively small.

Figure 9, because of the small amount of data, shows the effect of model learning in general. The results of the L-M prediction and the real value have a little gap, but the trend is the same, which is consistent with the effects of the error and error autocorrelation of Figure 8. The data quantity has a particular relationship with the accuracy of the prediction of the model. The black line shows the data predicted by the traditional time series forecasting method ARIMA. The RMSE, MAE, MPE, and MASE of the detected ARIMA are 1.588801, 1.051737, -86.78105, and 0.3993068. It can be seen that the trend of the ARIMA model is the same as that of the real value, but the numerical difference of the data is significant. The average difference is 6.537237°C, which of L-M neural network is 0.548778°C.

Daily Sequence Prediction. By setting the time granularity of the obtained data to the day which is collected every three hours. The result of the prediction is as shown in Figure 10.

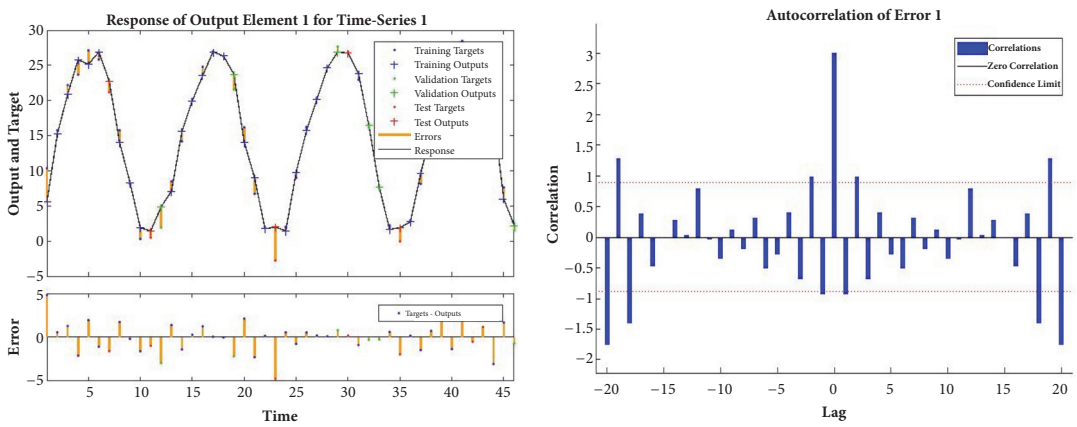
Figure 10(a) shows the MSE of the three datasets trained 15 times is displayed, and the MSE becomes best when the number of training times is close to 9 times. The first curve of Figure 10(b) shows that the gradient of training shows a decreasing trend. When the value of MU does not change, it means that the model training reaches the best state, and stop the practice. Otherwise, it will cause overfitting and affect the prediction effect. The third figure is the verification of neural networks, whose main impact is to look at the effects of network evolution.

The target and output sequence of three data sets is all distributed in a sinusoidal style. From the error diagram, the time series error is small, and the distribution is about 0. Figure 11(b) shows that the time series has a high 0 order autocorrelation, and the other self-correlation values are small, and most of them are distributed in the upper and lower confidence intervals.



(a) **MSE** (b) **Training state** (c) **Regression status**

FIGURE 7



(a) **Output element response** (b) **Error autocorrelation**

FIGURE 8

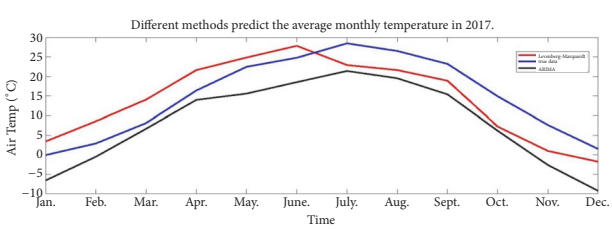


FIGURE 9: Average monthly temperature forecast and true value curve in 2017.

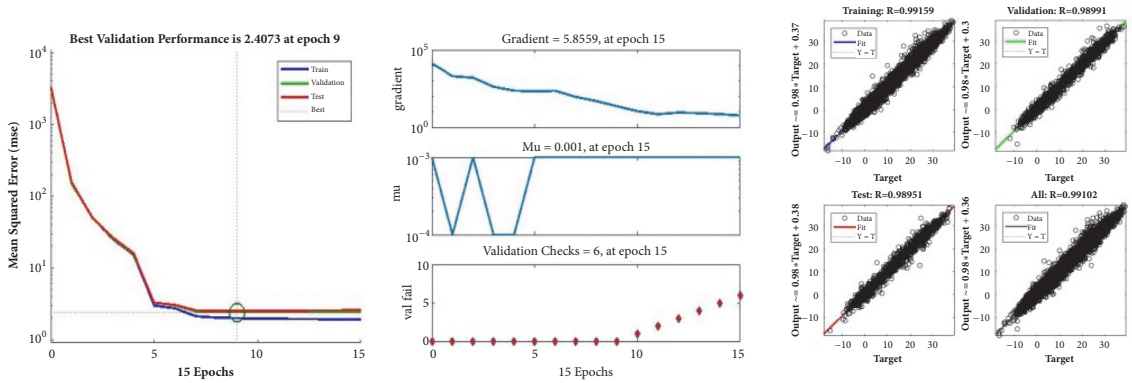
By comparing the predicted values with the real costs of different methods as shown in Figure 12, the sequences predicted by the ARIMA method are not accurate, indicating that ARIMA cannot achieve good results in predicting temperature time series.

Performance Evaluation. Two evaluation indexes are used to measure the accuracy of the model: mean square error (MSE) and R^2. The accuracy of the ARIMA model is measured by comparing the average absolute standard error (MASE). The MSE is the expectation of the square of the difference between the parameter estimate and the true value of the parameter, which can evaluate the degree of change of data, and the smaller MSE value shows the prediction model has better accuracy in describing experimental data. R^2 is similar to MSE, but the difference is that R^2 compares the trend of the predicted value with the actual value. R^2 close to 1 indicates that the linear relationship between y and $\hat{y}$ is very close. The corresponding calculation formulas are (1) and (2).

$$MSE\left(y, \hat{y}\right) = \frac{1}{n_m} \sum_{i=0}^{n_m-1} \left(y_i - \hat{y}_i\right)^2 \tag{1}$$

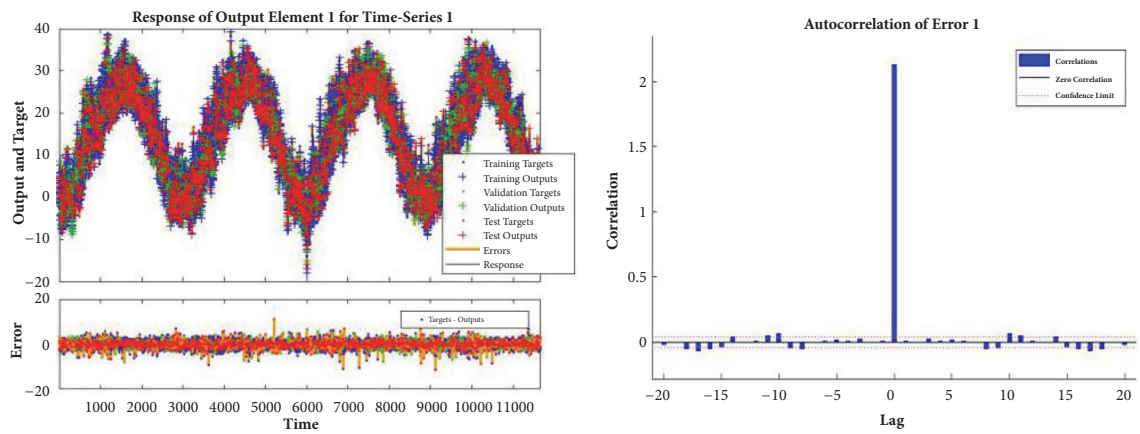
$$R^2\left(y, \hat{y}\right) = 1 - \frac{\sum_{i=0}^{n_m-1} \left(y_i - \hat{y}_i\right)^2}{\sum_{i=0}^{n_m-1} \left(y_i - \overline{y}\right)^2} \tag{2}$$

In formula (1) and formula (2), $\hat{y}$ is the predictive value, y is the real value, n_m is the sample capacity, by calculating MSE, and R^2 can evaluate the performance of the model. Besides, in formula (1), the numerator is the sum of squares of errors, and n_m is the degree of freedom. The evaluation index of the model is calculated as shown in Table 3.



(a) **MSE** (b) **Training state** (c) **Regression status**

Figure 10



(a) **Output element response** (b) **Error autocorrelation**

Figure 11

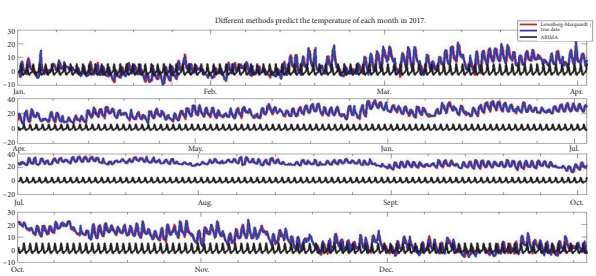


Figure 12: Comparison and analysis of results with a time granularity of 3 hours.

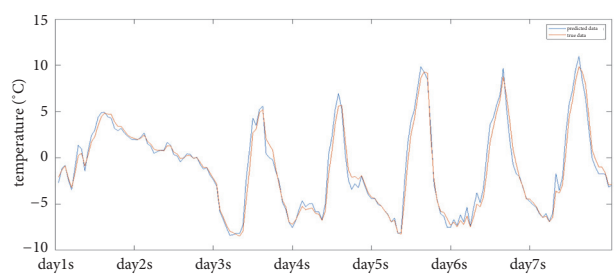


Figure 13: Comparison and analysis of results with a time granularity of 1 hour.

In Table 3, the monthly mean temperature measured MSE values larger than regular daily temperature MSE. It is due to the fewer data on average monthly temperature. The value R^2 is measured by the two experiments indicating that the fitting result is good, and the prediction error is smaller than the real value.

Model Application. To verify the accuracy of the temperature prediction model, we selected the air temperature data collected in the project area Dongying in 2016.

According to Figure 13, it can be seen that the neural network algorithm has high accuracy and consistency.

By calculating the MSE and R^2 of the model again, we got Table 4. The values of MSE in the table are all less than 1.8. The value of R^2 is above 94%, which indicates that the error of model prediction data is small and the fitting degree of verification data is high.

4.4. Summary. It can be found that when the data quantity is few, the difference between the two models' predicted results

Table 3: MSE and R^2 of air temperature test results.

	MSE (LM)		R^2 (LM)		MASE (ARIMA)	
	Average monthly temperature	Daily timing temperature	Average monthly temperature	Daily timing temperature	Average monthly temperature	Daily timing temperature
Training	0.0336	1.99	0.999	0.992	0.3993	0.86212
Validation	6.76	2.41	0.958	0.990		
Testing	1.06e	2.50	0.996	0.990		

Table 4: The MSE and R^2 of the temperature in the last week of 2016.

	MSE	R^2
	Daily timing temperature	Daily timing temperature
Training	1.17	0.995
Validation	1.33	0.994
Testing	1.30	0.995

and the real value is excellent, but the L-M is better than ARIMA. The prediction effect of ARIMA model is weak when the data volume is large and has no long-term trend.

5. Summary and Prospect

The security of the wireless sensor network has been the focus of attention all the time. The contradiction between the security protection measures and the attack mode of the Internet of Things will emerge. Therefore, the application of intelligent new intrusion detection model to intrusion detection is one of the key points in its security research.

In this study, we study the application of neural network in intrusion detection system. By modeling and analyzing the real-time data collected by the Internet of Things terminal, we constructed the intrusion detection rule base. The main research work has the following two points.

(1) The neural network is applied to the intrusion detection system, which makes full use of the self-organization and self-learning ability of the neural network. Moreover, we make up for the shortcomings of the lack of active protection for the security technology of the Internet of Things.

(2) Taking temperature data as an example, we study the accuracy and efficiency of the neural network and the traditional ARIMA model in predicting the type of data. The study provides a reference for introducing prediction research into intrusion detection.

Through the research and discussion of the intrusion detection system, we propose a network intrusion detection system based on a neural network. On the premise of guaranteeing the security and reliability of the system, the system fully considers the intelligent characteristics of data acquisition and intrusion detection nodes in the Internet of Things. Using the intelligent perception ability of intrusion detection nodes in the Internet of Things (IoT), we synthesize intrusion detection and data prediction and provide a new scheme for the construction of the IoT security 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 they have no conflicts of interest.

Acknowledgments

This work has been funded by Shandong's independent innovation achievements transformation project (2014ZZCX07106).

References

[1] M.-M. Zhao, S.-C. Zhao, L.-Y. Zhang et al., "Applications of eco-environmental big data: Progress and prospect," *Chinese Journal of Applied Ecology*, vol. 28, no. 5, pp. 1727–1734, 2017.

[2] D. Wanchun et al., "An energy-aware virtual machine scheduling method for service QoS enhancement in clouds over big data," *Concurrency & Computation Practice Experience*, vol. 29, 2016.

[3] Z. Jie, R. Huaijun, F. W. Xu, and X. Shiwei, "Research progress on the architecture and application field of agricultural IoT," *Agricultural Science in China*, vol. 50, no. 04, pp. 657–668, 2017.

[4] G. Wenjie and C. Zhao, "Research and application status and Development Countermeasures of Agricultural Internet of things," *Journal of Agricultural Machinery*, vol. 45, no. 07, pp. 222–230, 2014.

[5] T. Miao, "Prediction of winter wheat yield based on conditional vegetation temperature index," *Journal of Agricultural Machinery*, vol. 45, no. 02, pp. 239–245, 2014.

[6] G. Xing, X. Xu, H. Xiang, S. Xue, S. Ji, and J. Yang, "Fair energy-efficient virtual machine scheduling for Internet of Things applications in cloud environment," *International Journal of Distributed Sensor Networks*, vol. 13, no. 2, 2017.

[7] R. H. Jhaveri, N. M. Patel, Y. Zhong, and A. K. Sangaiah, "Sensitivity Analysis of an Attack-Pattern Discovery Based Trusted Routing Scheme for Mobile Ad-Hoc Networks in Industrial IoT," *IEEE Access*, vol. 6, pp. 20085–20103, 2018.

[8] W. Chen, "A novel security scheme based on instant encrypted transmission for internet of things," *Security & Communication Networks*, pp. 1–7, 2018.

[9] J. Li, X. Y. Huang, J. W. Li, X. F. Chen, and Y. Xiang, "Securely outsourcing attribute-based encryption with checkability," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2201–2210, 2014.

[10] X. Xu, W. Dou, X. Zhang, C. Hu, and J. Chen, "A traffic hotline discovery method over cloud of things using big taxi GPS data," *Software: Practice and Experience*, vol. 47, no. 3, pp. 361–377, 2017.

[11] Z. Pan, J. Lei, Y. Zhang, and F. L. Wang, "Adaptive fractional-Pixel motion estimation skipped algorithm for efficient HEVC motion estimation," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 1, pp. 1–19, 2018.

[12] A. Mishra, C. Siderius, K. Aberson, M. van der Ploeg, and J. Froebrich, "Short-term rainfall forecasts as a soft adaptation to climate change in irrigation management in North-East India," *Agricultural Water Management*, vol. 127, pp. 97–106, 2013.

[13] I. Białobrzewski, "Neural modeling of relative air humidity," *Computers and Electronics in Agriculture*, vol. 60, no. 1, pp. 1–7, 2008.

[14] R. E. Abdel-Aal, M. A. Elhadidy, and S. M. Shaahid, "Modeling and forecasting the mean hourly wind speed time series using GMDH-based abductive networks," *Journal of Renewable Energy*, vol. 34, no. 7, pp. 1686–1699, 2009.

[15] Z. Gao, R. Horton, L. Wang, H. Liu, and J. Wen, "An improved force-restore method for soil temperature prediction," *European Journal of Soil Science*, vol. 59, no. 5, pp. 972–981, 2008.

[16] H. Shoubo, *Agricultural Microclimate*, Zhejiang University Press, Zhejiang Province, 1 edition, 2001.

[17] L. Qi, X. Xu, X. Zhang et al., "Structural Balance Theory-based E-commerce recommendation over big rating data," in *Proceedings of the IEEE Transactions on Big Data*, 2016.

[18] J. Chang, L. Zhen, and L. Suping, "The application of Winters method in the prediction of temperature in summer," *Meteorological Science and Technology*, vol. S1, no. 3, pp. 107–109, 2005.

[19] Z. Yingchun, X. Dongrong, and Z. Yuandong, "Study on the time-series wind speed forecasting of the wind farm based on time series," *Electric Power Technology and Environmental Protection*, vol. 27, no. 2, pp. 237–240, 2003.

[20] B. Ustaoglu, H. K. Cigizoglu, and M. Karaca, "Forecast of daily mean, maximum and minimum temperature time series by three artificial neural network methods," *Meteorological Applications*, vol. 15, no. 4, pp. 431–445, 2008.

[21] Zuo. Zhiyu et al., "Forecast Model of Greenhouse Temperature Based on Time Series Method," *Transactions of the Chinese Society of Agricultural Machinery*, vol. 41, no. 11, pp. 173–177, 2010.

[22] Z. Xiaodan, "Prediction Model on Agricultural Greenhouse Temperature Based on Support Vector Machine with Parameter Optimization," *Journal of Beihua University (Natural Science)*, vol. 18, no. 4, pp. 557–560, 2017.

[23] H. H. Yu, Y. Y. Chen, S. G. Hassan, and D. L. Li, "Prediction of the temperature in a Chinese solar greenhouse based on LSSVM optimized by improved PSO," *Computers and Electronics in Agriculture*, vol. 122, pp. 94–102, 2016.

[24] L. Qiang et al., "Key Technologies and Applications of the Internet of Things," *Computer Science*, vol. 37, no. 6, pp. 1–4, 2010.

[25] Yi. Xu, *The principle and method of wireless sensor network*, Tsinghua University press, 2012.

[26] Luo. Shoushan, "Intrusion Detection [M]," in *Intrusion Detection [M]*, pp. 13–26, Beijing University of Posts and Telecommunications, 2004.

[27] Z. Jianfeng, "Research on Intrusion Detection Technology of WSN," *Digital Technology and Application*, vol. 11, pp. 193-194, 2014.

[28] C. Wu, "Time Optimization of Multiple Knowledge Transfers in the Big Data Environment," *Computers Materials & Continua*, vol. 54, no. 3, pp. 269–285, 2018.

[29] Z. Wei and Z. Feng, "Overview of data-driven time series forecasting methods," *Journal of Shaanxi University of Science & Technology*, vol. 28, no. 03, pp. 22–27, 2010.

[30] J. Juliang et al., "Application of genetic threshold auto-regressive model to forecasting meteorological time series," 17(04), 415-422, 2001.

[31] C.-S. Sun, Y.-N. Wang, and X.-R. Li, "Vector autoregression model of hourly wind speed and its applications in hourly wind speed forecasting," *Zhongguo Dianji Gongcheng Xuebao/Proceedings of the Chinese Society of Electrical Engineering*, vol. 28, no. 14, pp. 112–117, 2008.

[32] W. Qiming, "Autoregressive Conditional Heteroscedasticity (ARCH) Model and Its Application," *Forecasting*, vol. 4, no. 1, p. 47, 1998.

[33] H. Chen, "New method of load forecasting based on generalized autoregressive conditional heteroscedasticity model," *Dianli Xitong Zidonghua/Automation of Electric Power Systems*, vol. 31, no. 15, pp. 51–105, 2007.

[34] J. Qu, B. Xu, and Q. Jin, "Parameter identification method of large macro-micro coupled constitutive models based on identifiability analysis," *Computers, Materials and Continua*, vol. 20, no. 2, pp. 119–157, 2010.

[35] Z. Cai, H. Yan, P. Li, Z.-A. Huang, and C. Gao, "Towards secure and flexible EHR sharing in mobile health cloud under static assumptions," *Cluster Computing*, vol. 20, no. 3, pp. 2415–2422, 2017.

[36] R. Cheng, R. Xu, and X. Tang, "An abnormal network flow feature sequence prediction approach for DDoS attacks detection in the big data environment," *Computers Materials & Continua*, vol. 55, no. 1, pp. 095-095, 2018.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 6124160, 15 pages
https://doi.org/10.1155/2018/6124160

Research Article

Fingerprinting Network Entities Based on Traffic Analysis in High-Speed Network Environment

Xiaodan Gu, Ming Yang, Yiting Zhang, Peilong Pan, and Zhen Ling

School of Computer Science and Engineering, Southeast University, Nanjing, China

Correspondence should be addressed to Ming Yang; yangming2002@seu.edu.cn

Received 24 August 2018; Accepted 28 October 2018; Published 16 December 2018

Guest Editor: Yuan Yuan

For intrusion detection, it is increasingly important to detect the suspicious entities and potential threats. In this paper, we introduce the identification technologies of network entities to detect the potential intruders. However, traditional entities identification technologies based on the MAC address, IP address, or other explicit identifiers can be deactivated if the identifier is hidden or tampered. Meanwhile, the existing fingerprinting technology is also restricted by its limited performance and excessive time lapse. In order to realize entities identification in high-speed network environment, PFQ kernel module and Storm are used for high-speed packet capture and online traffic analysis, respectively. On this basis, a novel device fingerprinting technology based on runtime environment analysis is proposed, which employs logistic regression to implement online identification with a sliding window mechanism, reaching a recognition accuracy of 77.03% over a 60-minute period. In order to realize cross-device user identification, Web access records, domain names in DNS responses, and HTTP User-Agent information are extracted to constitute user behavioral fingerprints for online identification with Multinomial Naive Bayes model. When the minimum effective feature dimension is set to 9, it takes only 5 minutes to reach an accuracy of 79.51%. Performance test results show that the proposed methods can support over 10Gbps traffic capture and online analysis, and the system architecture is justified in practice because of its practicability and extensibility.

1. Introduction

With the rapid development and widespread application of computer networks, mobile communications, smart devices, and the Internet of Things technology, cyberspace is becoming more and more integrated into people's social life. People can access services through various devices anytime and anywhere, thereby realizing the interconnection between people and people, people and things, and even things and things. However, while cyberspace brings a lot of convenience to people, network attacks such as DDoS attacks, worm attacks, information theft, and cyber fraud have become increasingly severe. Therefore, it is imperative to effectively protect against cyber threats.

The intrusion detection system is used to monitor a network or system, which can identify malicious activities or policy violations from both inside and outside intruders. As an important and dynamic research area, the network intrusion detection technology can identify malicious activities by monitoring and analyzing inbound and outbound traffic [1, 2]. But there is less work that can effectively identify potential threats if an intruder has no abnormal activity. To address this issue, we introduce the identification technologies of network entities to detect the intruder with no abnormal activity, which mainly consist of device identification and user identification. The basic idea is that if we detect unauthorized devices or unauthorized users using authorized devices, we can indicate that a network intrusion may be taking place.

However, the conventional identification technologies of network entities are usually based on explicit identifiers. For example, user devices are always identified by the MAC addresses or browser cookies, and the user is identified by intercepting and verifying his account information in the network traffic through the Deep Packet Inspection (DPI). These explicit identifiers can be easily hidden or tampered, causing an identification failure. In response to this problem, the researchers have demonstrated [3] that the absence of explicit identifiers brings no harm as long as well-chosen implicit

identifiers are reasonably combined. The potential selections are instantiated as the SSID information in the 802.11 active probe frame, the plug-in installed in the browser, the font library in the system, etc. Although these implicit identifiers cannot uniquely identify a network entity individually, they are hard to be concealed because they usually reflect the user's personalized configuration, historical behavior records, or subtle differences between entities. Therefore, in practice, a combination of the implicit identifiers is usually utilized to generate a fingerprint for device identification and user tracking. The fingerprinting technology based on implicit identifiers is essentially a method of traffic analysis and side-channel attack. Although its effectiveness has been initially verified by existing work, there are still many problems to be solved in practical application, including the selection of efficient features, realization of real-time processing of network traffic in a high-speed network environment, and quick identification of devices and users in a short period of time.

In view of the problems mentioned above, we use PFQ kernel module to realize high-speed capture of network packets and use Storm, a well-known distributed real-time streaming data processing technology, to realize online analysis of network traffic. Based on these, a device identification method based on runtime environment analysis and a network user identification method based on behavioral fingerprinting are proposed separately. The main contributions of the paper are as follows:

(i) A distributed traffic analysis framework for high-speed network environments is designed. The framework uses the PFQ kernel module to implement packet capture, Kafka for packet distribution, and Storm for packet content analysis and information extraction of applications, operating systems, HTTP User-Agents, domain names, and Web access records.

(ii) An online device identification technology based on the analysis of user device operating environment is proposed. This technology selects 961 features such as applications, operating systems, and HTTP User-Agent fields to constitute the fingerprints of the devices, while a variety of offline classification models are trained and verified. Finally we select the logistic regression algorithm to identify the user device in a sliding window manner.

(iii) In order to realize network user identification, Web access records, domain names, and HTTP User-Agent fields are selected to constitute user behavioral fingerprints according to the user's network behavior habits. These fingerprints, containing a total of 57593 feature columns, are trained and verified by the two offline classification models using machine learning, which are Naive Bayes and random forest models. By comparison, the Multinomial Naive Bayes model is found to outperform the random forest model, so it is chosen as the classification algorithm for identifying online users in a sliding window manner.

(iv) In order to achieve high-efficiency identification, the impacts of different time window sizes on the recognition rate are tested. Specifically, the accuracy rate of device identification reaches 77.03% within 60 minutes, and 79.51% of user identification accuracy can be achieved within 5 minutes. The packet capture rate, distributed processing speed, and online recognition response speed are also evaluated to verify the practicability of the proposed identification technology.

An early version of the network entities identification is presented in [4]. In the early version, we design a distributed high-speed traffic analysis framework to recognize devices based on runtime traffic. In this journal version, to deal with the cross-device scenario, we further analyze the user's network behavior habits and generate fingerprints to identify the network users. We also evaluate the identification performance difference between the Boolean and numerical type fingerprints in this extended version.

The rest of this paper is organized as follows. In Section 2 we overview the related work. In Section 3 we describe the overall design of our network entities fingerprinting technology. Section 4 introduces a distributed traffic analysis framework for high-speed network environments. In Sections 5 and 6, we present the details of the identification technologies of network device and user, respectively. Section 7 tests the performance of the identification technologies. Finally, the paper is concluded in Section 8.

2. Related Work

The identification of network users and that of devices are two differentiated research directions but are closely related. Earlier device identification technologies mainly obtain the information of the hardware, operating systems, network protocols, and other parameters by collecting and analyzing the physical signals or traffic generated by the device. For example, in the physical layer [5], the TCP packet time stamps are analyzed to obtain the clock skew [6], and the Ethernet frames are analyzed to obtain the differences among the analog signals of different devices [7]. While in the operating system layer, the active scanning algorithm used by the wireless device driver might be inferred by analyzing the interval time of 802.11 probe request frames [8]. In the application layer, the User-Agent field, IP address, browser cookie, user login ID, and other identity information are extracted through the traffic analysis in clear text [9]. The interval time, number, direction, and other attributes of the encrypted wireless packets are analyzed for the distinction of different terminal applications [10]. Other researches have applied different threat models to achieve the identification of devices, e.g., the device recognition based on browsers [11–17] and that based on mobile applications [18, 19].

The above-mentioned identification technologies are merely, in essence, the identification of a single browser [20] or a single terminal device. They are far from being capable enough to identify the user's cross-device activities. For example, in the scenario of intrusion detection, if some

intruder occupies an authorized device, we cannot detect the intrusion by using the device identification technologies. So it is necessary to carry out the research on the user identification technology based on behavioral fingerprints.

Essentially, user identification technologies based on behavioral fingerprinting are biometric, which use the inherent physiological characteristics or behavioral characteristics of the human body for identification. They can be categoried into two types. The former one has been widely used by employing the characteristics of human body parts, such as fingerprint identification, face identification, and iris identification.

The behavior-based identification technology [21] extracts the features for identification with the information of the user's operation skills, knowledge, styles, preferences, and strategies revealed in behaviors. For example, researchers have found that different users differentiate from each other in moving, clicking, dragging, and releasing the mouse [22]. Some may be different in key stroking when keyboarding [23]. All of these differences can help to extract fingerprints for effective identification. In the network area, users have different behavioral patterns for network access due to different preferences, habits, etc. Different behavior patterns lead to different traffic flows. Therefore, researchers believe that the network traffic generated by the user can be regarded as biometric for user identification [24].

In order to identify users based on network traffic, early solutions are implemented by extracting explicit identifiers such as IP addresses or MAC addresses [3]. However, such method based on explicit identifiers is not reliable for that it will fail once the user changes the IP address or devices. So far, the dynamic address allocation scheme adopted by ISP makes users change the IP more frequently. To address this issue, the researchers apply the behavior fingerprinting technology to user identification based on network traffic. Padmanabhan et al. [25] find that different users may have different behaviors when browsing the same website. By analyzing the real data, they extract the users' clickprints to generate behavioral fingerprints. Pang et al. [3] propose to explore the destination address, network name, 802.11 option configuration, and broadcast frame length so as to identify the user from the perspective of protocol and user preferences. This is actually a comprehensive application of user-related and device-related implicit identifiers.

Yang [26] uses data mining techniques on the Web browsing dataset in order to mine association rules for each user's behavior, proposes three strength evaluation criteria based on support and lift to generate fingerprints, and finally calculates the distance between fingerprints for identification. Kumpošt et al. [27] believe that the websites visited by the user and the corresponding frequencies, which reflect individual preferences, can be identified as a behavioral fingerprint. They store the source IP, destination IP, and frequency in a two-dimensional matrix and perform the inverse document frequency and cosine similarity algorithm to identify users. Similarly, Herrmann et al. [28] extract user's destination domain names and the corresponding visiting frequency to derive the behavioral fingerprints and use Multinomial Naive Bayes classifier to classify them. Experiments are conducted on a dataset containing HTTP traffic generated by 28 volunteers, and a 73% accuracy rate is obtained.

Since the data set used in the experiments [28] is not big enough to prove the feasibility of the method, the author conducts a larger-scale experiment in the later work [29]. He tests a dataset containing more than 2,100 users' DNS requests, uses the cosine similarity algorithm to filter the noise data, and finally obtains an accuracy of 88%. In addition, Herrmann et al. [30] also compare and evaluate three classification methods through a large number of experiments, including the Multinomial Naive Bayes classifier, the nearest neighbor algorithm, and association rules mining technology. Kim et al. [31] get the user's behavioral fingerprints based on DNS traffic by analyzing the domain name, the sequence of domain names, and the requested periods. Gu et al. [32] infer the users' preferences through the semantic analyses of search records and achieve an accuracy of 93.79% on the dataset of 509 network users.

Overall, the current device and user identification researches have some defects. For example, only a few features are explored and the identification effect is prone to jitter. In addition, the existing studies are not time-efficient enough as it needs to aggregate a whole day's traffic as a fingerprint.

To avoid the aforementioned problems, we adopt the distributed processing technology to extract information such as applications, operating systems, HTTP User-Agent, domain names, and Web access records in real time from high-speed network traffic. Then we propose two online identification methods based on the runtime environment and the behavioral fingerprints, respectively, which are made possible in the pattern of sliding windows. In addition, we also focus on testing the impact of different traffic window sizes on the identification rate and thereby prove the high efficiency and practicality of the proposed technologies.

3. Overall Design of Fingerprinting

The overall design of our network entities fingerprinting technology is shown in Figure 1. The initial step leverages the PFQ-based high-speed packet capture module to capture high-speed network traffic and then forwards the packets to distributed high-speed network traffic processing modules through the Kafka message queue. Then the processing module parses the message content, extracts the relevant feature data, and stores it in the HBase. Finally, the Spark-based online identification module periodically reads the feature data from the distributed database and realizes the online identification of network devices and users by employing the machine learning algorithms in a sliding window mechanism. The working mechanism and functions of each module are specified as follows:

(i) **PFQ-based high-speed packet capture module.** Configure a mirror port on the switch or router, or use an optical splitter to mirror the traffic to a data distribution server. The high-speed packet capture module on this server achieves highly efficient packet capture by adopting the memory mapping mechanism, zero

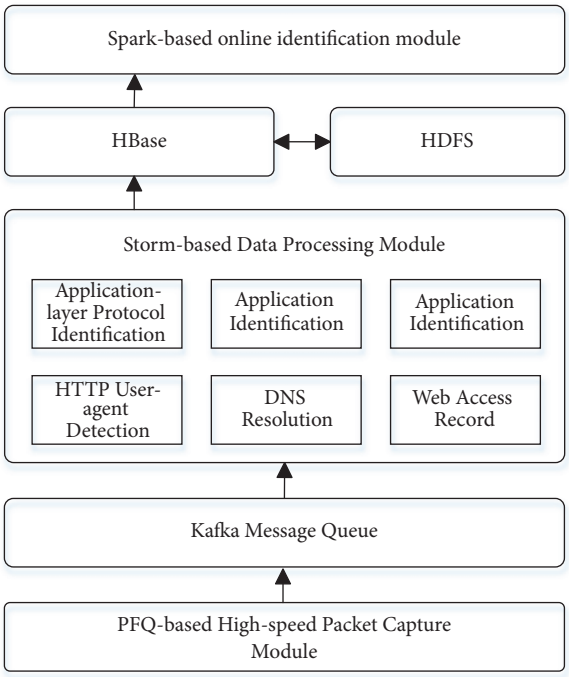


Figure 1: Overall design of network entities identification.

copy technology, and double-buffering mechanism based on the PFQ Linux kernel module.

(ii) **Kafka message queue.** The distributed message queue is a data transmission channel between the packet capture module and the distributed processing module. More specifically, it is a buffer zone for coordinating the producer and consumer. We use Kafka to implement distributed message publish, which has a high linear extensibility in adapting to the high-speed data transmission scenario.

(iii) **Storm-based data processing module.** The distributed data processing module, as a core functional module, undertakes all processing and analysis tasks for network traffic, including parsing of network messages, identification of application protocols, identification of applications, and finally extracts and stores data of applications, operating systems, Web access records, domain names, and HTTP User-Agent fields. We realize distributed streaming data processing based on the Storm platform, which can achieve high-speed data reading combined with Kafka queues and can achieve high-speed data writing combined with HBase. Meanwhile, data transmission performance between the internal components of Storm is also very efficient.

(iv) **HBase.** The online identification module is proposed to read and analyze the data periodically. Thus, as a distributed column-oriented database, the HBase functions as a data medium between the distributed data processing module and the online identification module.

(v) **Spark-based online identification module.** Our identification module is based on the Spark platform and designed to fit into two scenarios. For the device identification scenario, the module extracts device features about runtime environment to generate the fingerprints. For the cross-device identification scenario, the user behavior data are collected to implement fingerprinting for network users. The relevant feature data are read from the HBase distributed database, and the machine learning algorithms in Spark MLlib along with the sliding window mechanism are employed to identify the network devices and users online.

4. Collection and Distributed Processing of High-Speed Network Traffic

4.1. Collection of High-Speed Network Traffic. Compared to Pcap, which is a traditional packet sniffing toolkit, PFQ is a better designed network packet capture framework customized for the optimization of multicore CPUs and multihardware queue network interfaces. It is primarily used for efficient packet capture and transmission on Linux. In its internal implementation, PFQ eliminates the cost of copying packets from kernel space to user space by adopting a memory mapping mechanism, and performs concurrency operations of user-space applications and PFQ kernel packet grabbing programs on the buffers by means of double-buffering technology. The core components of PFQ fall into three parts: packet extracting program, packet forwarding module, and socket queue. First, the packet extraction program directly obtains the packets from the network interface card (NIC) driver and transfers them to the batch queue. Then, the packet forwarding module selects the socket and sends packets to the user-space applications.

After the packets are captured, librdkafka is used to write them into the Kafka message queue. In this paper, we decouple the high-speed packet capture module and the Kafka message queue through the open source project Blockmon [33].

4.2. Distributed Processing of Network Traffic. When the captured packets are written into the Kafka message queue, we implement distributed analysis and processing of packets based on the Storm platform. The following are the key concepts related to the Storm:

(i) Topologies: the logic of the application which defines various components and the ways of communication between them.

(ii) Streams: data flows consisting of message tuples transmitted between Storm components. All Streams are parallel transferred in a distributed way.

(iii) Spouts: data sources. Usually, a Spout reads messages from an external data source and transfers them into the Topology in the form of tuples.

(iv) Bolts: Storm processing units. Each Bolt completes one or more processing tasks and is responsible for

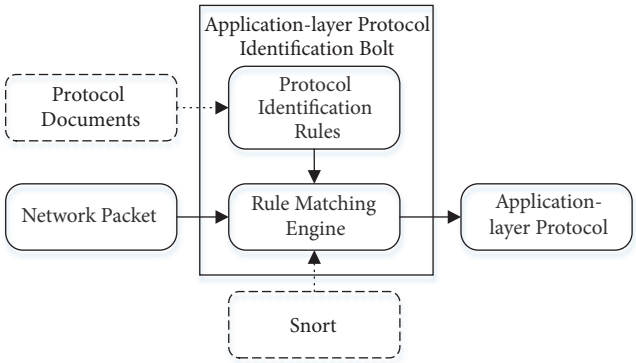


Figure 2: Application layer protocol identification bolt.

transmitting the processing results to the external system for storage or display.

4.2.1. Input of Flow Data. Data transmission between Spouts and Bolts, as well as that among Bolts, is in the form of Streams, while Spouts obtain data from external sources in different ways. In this paper, we use KafkaSpout to read packets from the Kafka message queue and transmit tuples to the packets parsing Bolts.

In the distributed processing environment, KafkaSpouts retrieve data from Kafka partitions in parallel on all slave nodes. And the degree of parallelism has a crucial influence on the throughput of the system. Meanwhile, it is affected by the number of Kafka partitions, because each Kafka partition can only be consumed by one KafkaSpout. Namely, the key to improving Storm throughput is to increase the number of Kafka partitions.

4.2.2. Packets Analyzing and Filtering. The inputting packet tuples are analyzed and filtered to obtain the content of messages, and the header information in each layer of the network protocol is extracted, including PFQ pkthdr header, Ethernet frame header, IP header, TCP header, and UDP header. In the process of packet analysis, several rules are set to filter packets unrelated to network device identification, such as the network control protocol packets and routing protocol packets to improve the processing efficiency of the system.

4.2.3. Application Protocol Identification. Traditionally, in order to identify application protocols, the port numbers of the captured packets are always matched with the well-known ones. Such method is deficient due to a high false positive rate. Therefore, we aim to improve the identification accuracy by employing DPI technology to analyze the packet payloads. Although such method is less efficient, its accuracy rate is significantly higher than that of the former. The module of application protocol identification is developed in two phases: (a) designing the rule matching engine and (b) writing protocol identification rules. The first step extracts the rule matching engine of the Snort core components and introduces multithreading. Then, based on referencing protocol documents, we summarize the characteristics of a typical protocol and write an appropriate rule according to the writing specification of Snort rules.

```
alert udp $EXTERNAL_NET any -> $HOME_NET any
(msg: "snmp"; content: "|30|"; offset:0; depth:1;
byte_test:1,<,0x80,1; content: "|02|"; offset:2; depth:1;
sid:70; rev:1;)
```

Box 1

The process of identifying application protocols is shown in Figure 2. The rule matching engine generates a rule tree according to the specified protocol identification rules and performs rule matching for network traffic. When a match occurs, it indicates that the packet is identified as a specific type of protocol.

Box 1 shows an identification rule for the SNMP protocol and corresponding explanation ensues.

This statement indicates that the rule is released as the first version, with the id of 70. All the UDP packets sent from any port or IP address are detected without exception. According to the identification rule, the first byte value of the application layer payload is 0x30. The second byte value must be less than 0x80, and the third is 0x02. When the match is successful, the alert action is triggered and the protocol type value SNMP is returned to the caller.

After a review of various typical protocol documents, we have completed the writing of recognition rules for 25 typical application layer protocols such as BITTORRENT, DNS, DROPBOX, HTTP, SMTP, and SSH.

4.2.4. Application Identification. After the identification of the application protocol, we further figure out the type of application that generates the packets. As we all know, apart from the traffic generated by the user interaction, the background process of the application communicates with the server periodically, thereby generating more traffic. We analyze the traffic and extract various features to identify the applications.

The data transmission between an application and the server is generally divided into two cases. First, the application uses a custom data transmission protocol, such as the OICQ protocol, which is designed by Tencent and is merely used as the data transmission protocol of QQ. In this case, as long as the application protocol has been identified, the application is sure to be identified. Second, multiple kinds of applications share an application layer data transmission protocol, such as the HTTP protocol, to encapsulate data transmitted between the client and the server. For this situation, we distinguish different applications by extracting multiple field values from the traffic. For example, in the HTTP protocol, the HOST field represents the combination of the domain name and the port number of the server address. Usually, the addresses and corresponding domain names of applications devised by different companies are not identical. Even though different applications provided by the same company share the same server, the addresses are still distinguished from each other with different HTTP request parameters. Therefore, the combination of the HOST field

of HTTP protocol and request parameters can be used to identify different applications.

This paper observes and analyzes the application traffic in the experimental network and summarizes a collection of 116 commonly used application identification rules under 21 categories, such as browser, e-mail, remote management, online game, instant messaging, social networking, Web disk, input tool, online video, P2P video, and stock software. These identification rules cover traffic generated from users' clicks, login activities, automatic updates, and background process communications.

4.2.5. HTTP User-Agent Detection. As the name suggests, the User-Agent field in HTTP traffic contains the user agent information. Generally, the User-Agent field generated by a browser contains the information like the types and versions of the browser and the operating system. As an important piece of information in the User-Agent field, a specific operating system has its own structure-mapping rules, diversifying the types of all the existing operating systems. For instance, the prefixes of the Windows operating systems are normally Windows NT, and suffixes represent specific operating system versions. The identification rules of the operating systems presented in this paper cover the mainstream operating systems such as Windows, Mac OS, OpenBSD, and Ubuntu.

4.2.6. DNS Resolution and Web Access Records. The user generates a large number of HTTP requests and DNS requests when he manipulates the applications or accesses websites using the browser. The destination IP address in an HTTP request together with the corresponding time information can reflect the behavioral characteristics of the user to some extent. And the DNS responses can help us to associate multiple IP addresses of the same domain name together. The resolution of DNS packets is mainly for the IPv4 protocol. From the response packets, we can extract the <domain name, address> pairs. Specifically, the Questions field indicates the number of requested domain names, and the number of corresponding addresses is contained in the Answer RRs field. There are many kinds of DNS response types, which are distinguished by the Type field. For example, the A record maps a domain name to the corresponding IP address while the CNAME record maps an alias name to a canonical domain name.

4.3. Distributed Storage of Network Traffic Data. Based on the processing and extraction of the packets, the information of the extracted applications, operating systems, HTTP User-Agent, DNS, and Web access records is stored in the corresponding columns of the distributed database HBase. By reading data from HBase, the device identification and user identification are implemented.

5. Device Identification Based on Runtime Environment Analysis

5.1. Basic Ideas. For the first device identification scenario, we propose a novel identification method based on runtime environment analysis. Its basic idea is to realize the unique identification of devices based on the combination of the device operating system, the HTTP User-Agent information, and especially the installed applications.

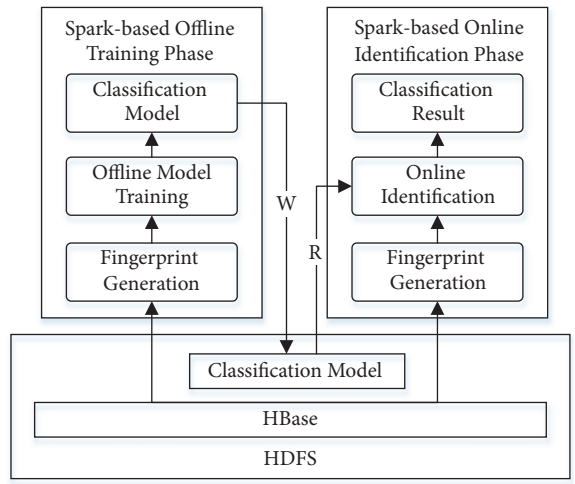


Figure 3: Two-stage fingerprint recognition.

As shown in Figure 3, the identification process can be divided into two phases, namely, the Spark-based offline training phase and the Spark-based online training phase. In the offline training phase, Spark distributedly reads the relevant features from the HBase, generates the corresponding device fingerprint denoted as a vector, and accordingly learns an appropriate classification model by offline training. The dataset needed for offline training and verification can be labeled with IP addresses (MAC addresses can be utilized for labeling in LANs instead). In the online identification stage, distributed analysis and feature extraction are performed on the real-time traffic. And the generated fingerprint vector is classified by the offline training model. Finally, the classification results indicate the identity of the device.

5.2. Feature Selection and Fingerprint Generation. This section focuses on the operating environment of user devices and derives a device identification technology based on its characteristics. The operating environment mainly includes two types of features, which are the operating system type and application type (version information is included). To generate the device fingerprints for identification of the user device, we collect the type and version information of applications from the results of application identification, and extract the attributes such as the browser type and version and operating system type from the results of HTTP User-Agent detection.

Specifically, the device fingerprints are generated by analyzing the traffic per unit time. If the traffic of an application is detected within the time period, the corresponding feature attribute of the application is set to 1 or the corresponding frequency. The dimension of the device fingerprint feature vector is 961. According to the value types of the extracted feature attributes, we design two types of device fingerprints: boolean type device fingerprint and numerical type

device fingerprint, where the Boolean type device fingerprint indicates whether features such as applications or operating systems appear in the network traffic, and the numeric device fingerprint indicates how often these features appear.

Note that all the identifiable application type sets are $S = \{S_1, S_2, \cdots, S_N\}$ and the ith application's version set is $V_i = \{V_i^1, V_i^2, \cdots, V_i^{C_i}\}$. C_i refers to the total number of recognizable versions of the ith application and OS represents the operating system type. The feature vector of the device fingerprint can be presented by formula (1).

$$\overrightarrow{FP_{dev}} = \left\{ \underbrace{S_1, S_1V_1^1, \cdots, S_1V_1^{C_1}}_{S_1}, \underbrace{S_2, S_2V_2^1, \cdots, S_2V_2^{C_2}}_{S_2}, \cdots, \underbrace{S_N, S_NV_N^1, \cdots, S_NV_N^{C_N}}_{S_N}, OS \right\} \tag{1}$$

When the value of the attribute in fingerprint $\overrightarrow{FP_{dev}}$ is numerical, it is a numerical type device fingerprint. When the value of the attribute is only 0 or 1, it is a Boolean type one. Then the device identification problem can be modeled as a multiclassification problem in machine learning.

5.3. Offline Model Training and Verification. Since the efficiency of identification depends greatly on the classification algorithm and the dimension of the device fingerprint vector is relatively small, the multiclassification algorithm can generally be used to train the identification model. We compare the classification effects of Multinomial Naive Bayes algorithm, random forest algorithm, and the logistic regression algorithm. After that, the best performed algorithm is selected for online identification.

We collect network traffic of 118 user devices for 53 days from June 1st to July 23rd, 2016. The network traffic produced on each device is examined per hour. Based on the examination, the features are extracted to form a fingerprint (all zero-vector fingerprints are discarded). Then we get 50,305 valid device fingerprints in total. The data collected in the first 30 days is used to train and verify the offline model, including 30,148 records, while the remaining 20,157 records gathered in the following days are used to evaluate the accuracy of the classification model.

During the offline training process, the device fingerprints in the data set are randomly divided into two subsets, one being the training set containing 70% fingerprints and the other being the verification set containing the remaining 30%. Above all, the classification model of Boolean type device fingerprinting is trained and verified as follows.

5.3.1. Training and Verification of the Classification Model of Boolean Type Device Fingerprinting. First, the random forest classification model is trained. Different from Multinomial Naive Bayesian and logistic regression, the random forest classification model has two parameters that need to be tuned, i.e., the number of decision trees (*nums*) and the maximal depth of the decision tree (*depth*). The parameter of *nums* affects the accuracy of the overall classification, while *depth* affects the classification accuracy of each decision tree. Training and testing are performed under different values of *nums* and *depth*, and the obtained classification accuracy is shown in Figure 4(a).

As can be seen from Figure 4(a), *depth* generally has a greater impact on the classification accuracy. With the increase of *depth*, the accuracy is significantly improved. When the value of *depth* is 30, the classification turns to be optimal. The effect of *nums* on the classification accuracy correlates positively to *depth*: when *depth* is small, the accuracy rate rises with the increase of *nums*; when the *depth* is larger, the classification accuracy rate first goes up with the increase of *nums*, and then remains stable when *nums* is greater than 20. When the value of *nums* is 150, the classification accuracy is the highest. Therefore, we set the value of *nums* as 150 and the value of *depth* as 30, respectively, for optimization.

Then, the Multinomial Naive Bayesian and logistic regression classification models are trained separately, and the classification accuracy of the models is evaluated by the verification set and test set, respectively. Figure 5(a) shows the classification accuracy of the three models, where MNB refers to Multinomial Naive Bayes, RF denotes random forest, and LR represents logistic regression. From Figure 5(a), it can be seen that the classification accuracy of the verification set by performing the logistic regression algorithm is considerably higher than that of other algorithms. For the same algorithm, the classification effect of the test set is significantly lower than that of the verification set. This is because the data in the training and verification set is randomly segmented, and data in the test and training set has a time-series relationship. Moreover, the operating environment of the device is likely to change, so the accuracy of the device identification may gradually decrease over time.

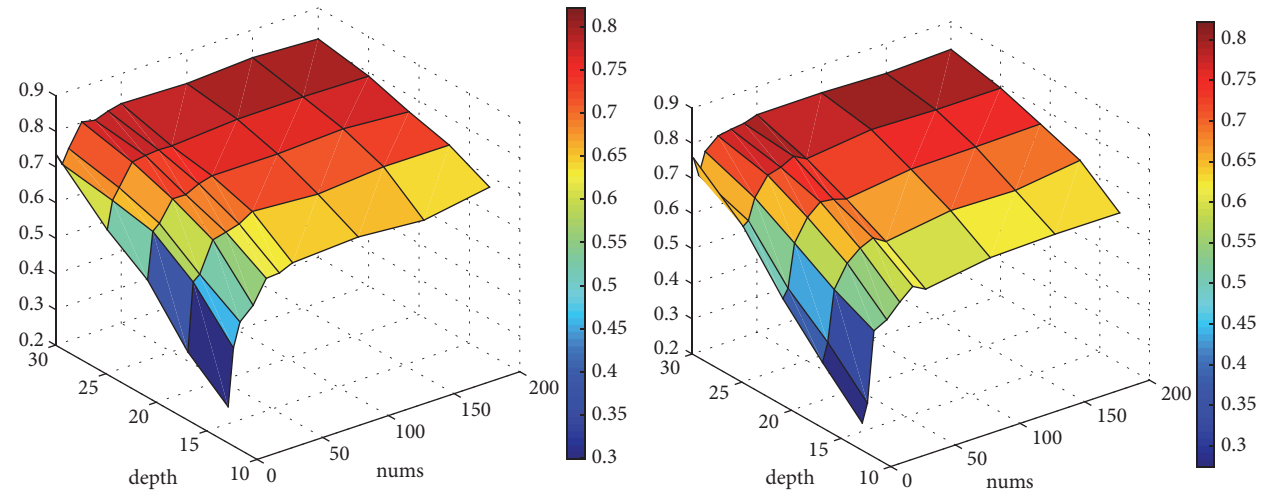
Further analysis of the data reveals that a portion of records in the device fingerprint vectors stay close to the full value of 0. This is due to the fact that not all device traffic is generated by the identifiable applications involved in this paper. Such traffic cannot be identified as valid device information. To deal with it, the following definitions are given.

Definition 1 (effective dimension). Given a fingerprint vector, denote the number of feature columns with a nonzero value as effective dimension.

Definition 2 (the minimal effective dimension). For the set of fingerprint vectors to be identified, the minimal effective dimension is defined as the threshold, below which the fingerprints are deemed as invalid ones and filtered out due to a lack of information.

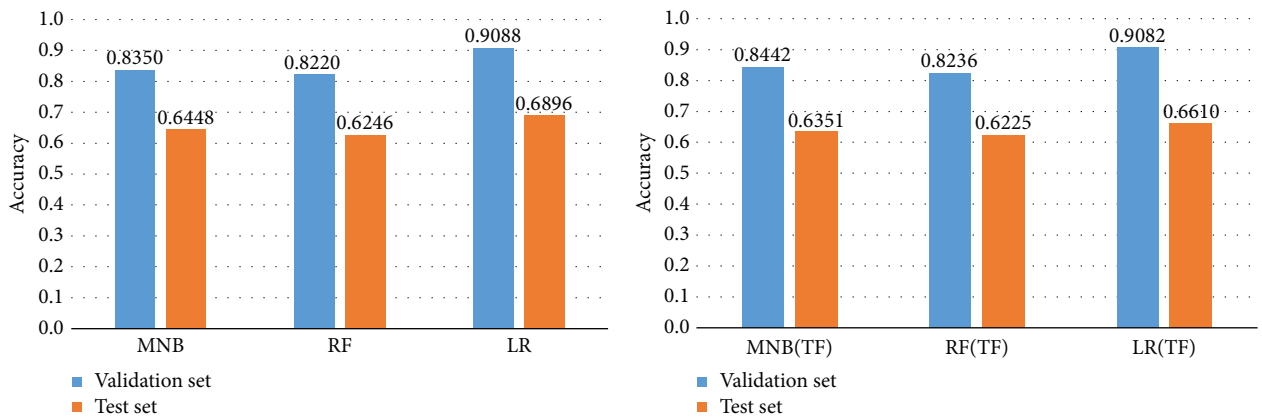
Figure 6(a) shows the impact of this threshold on the classification accuracy. And Table 1 shows the ratio of valid device fingerprints to the total number with different values of the minimal effective dimension, which shows the traffic coverage rate of device identification.

From Figure 6(a) and Table 1, we can see that the classification accuracy of the validation set and the test set increases gradually as the minimal effective dimension



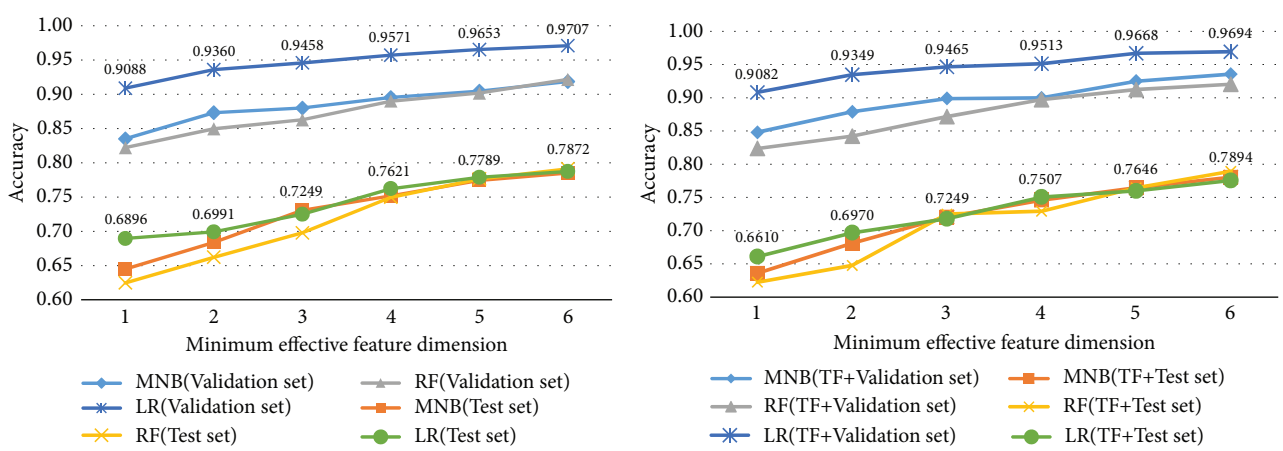
(a) Boolean type fingerprints (b) Numerical type fingerprints

Figure 4: The classification accuracy of device fingerprints using random forest model.



(a) Boolean type fingerprints (b) Numerical type fingerprints

Figure 5: The accuracy of device recognition using MNB, RF, and LR.



(a) Boolean type fingerprints (b) Numerical type fingerprints

Figure 6: The effect of the minimum effective feature dimension on the device classification accuracy.

Table 1: The effect of the minimum effective feature dimension on the number of device fingerprints.

Minimum Effective Feature Dimension	Validation Set	Test Set
1	100.00%	100.00%
2	89.58%	90.43%
3	82.98%	81.66%
4	75.94%	74.85%
5	70.57%	68.99%
6	63.73%	62.08%

climbs. The classification accuracy values of three models all plateau close to 80% for the test set when the minimal effective dimension is 6. However, despite the top accuracy, only 62.08% of device fingerprints in the test set are retained. In comparison, when the minimal effective dimension is lowered to 4, the classification accuracy of Multinomial Naive Bayes and logistic regression for the test set is higher than 75%, and 74.85% of device fingerprints in the test set are retained. Considering the effect of the minimal effective dimension on fingerprint classification accuracy and traffic coverage, the minimal effective dimension 4 is determined as the threshold for filtering fingerprint data. Since the logistic regression model performs comparatively better for both the validation set and the test set, the logistic regression model is chosen as the online identification model for the Boolean type device fingerprinting.

5.3.2. Training and Verification of Classification Model for Numerical Type Device Fingerprinting. For numerical type device fingerprinting, the same random forest parameters are first trained. The results are shown in Figure 4(b). When *nums* is 100 and depth is 30, the random forest model has the best classification effect. Since the specific value of each feature has an important influence on the classification result of the Multinomial Naive Bayesian classification model, we need to perform the term frequency (TF) transform as shown in formula (2) for each feature value.

$$f_x^{tf} = 1 + \log\left(f_x\right) \tag{2}$$

To further implement numerical type device fingerprinting, we train the Multinomial Naive Bayes classification model and the logistic regression classification model, respectively, and calculate the classification accuracy on the verification set and the test set. The results are shown in Figure 5(b). We also verify the impact of the minimum effective dimension on the classification accuracy, as shown in Figure 6(b). By comparing Figures 5(a) and 5(b), Figure 6(a) and Figure 6(b), respectively, we can find that the performances of Boolean and numerical type device fingerprinting are basically the same and can both achieve a comparatively high device identification accuracy. However, because the numerical type fingerprints may fluctuate on feature values, we only leverage the Boolean type device fingerprinting to test the online identification accuracy of devices.

5.4. Online Identification of User Devices. The online identification of user devices employs the Boolean type device fingerprinting, and a logistic regression model is taken as the classification model. The experiment is based on the sliding window mechanism, which simulates the online identification process by replaying network traffic in the test set for 23 days. The sliding window has two important parameters: the windows slide and the windows size.

A prediction is made iteratively when the sliding window slides backward over a distance of the window slide. The windows size is the range that fully covers flow data. When we want to make a prediction, we need to read feature data within the time range of the previous windows size from the current moment. The online identification accuracy rate of the user devices is counted as the ratio of the total number of device fingerprints correctly classified to the total number of device fingerprints in all the windows.

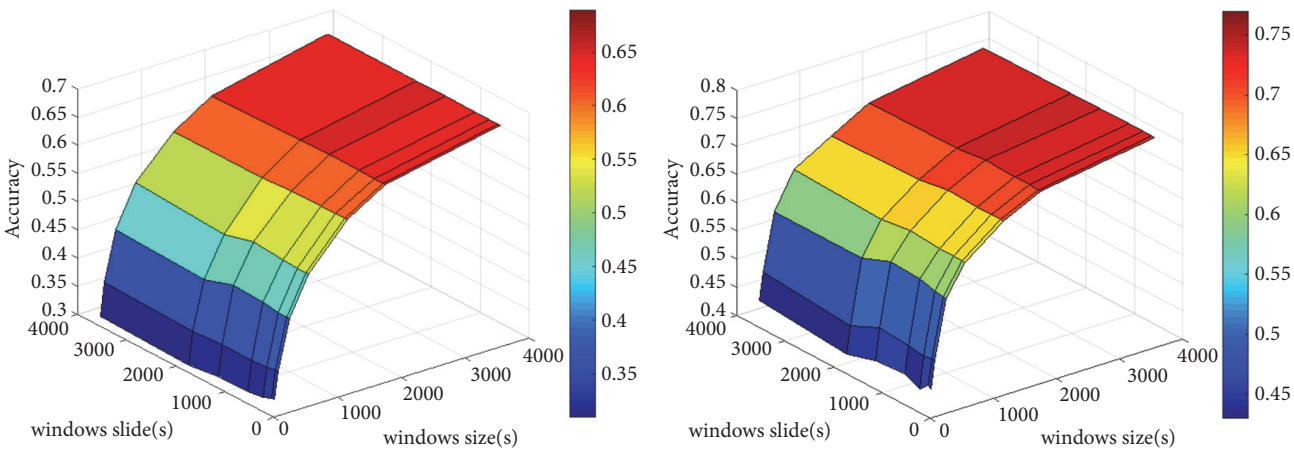
In this paper, the values of the windows slide and the windows size are set to be 1 minute, 2 minutes, 5 minutes, 10 minutes, 20 minutes, 30 minutes, and 60 minutes, respectively. By adjusting the values, we analyze how the two parameters influence online identification accuracy of user devices. Figure 7 shows the results when the minimum effective dimensions are 1 and 4, respectively.

As can be seen from the figure, the online identification accuracy rate is barely influenced by the change of windows slide, while it is in a positive correlated response to the increase of windows size. That is, the bigger the windows size, the more accurate the identification. When the windows size is 60 minutes, the identification accuracy rate reaches a maximum value of 68.93%. If we filter the data with low information content by setting the minimal effective dimension to 4, the maximal online identification accuracy will increase to 77.03%.

6. User Identification Based on Network Behavioral Fingerprints

6.1. Basic Ideas. In the scenario of intrusion detection, if some intruder occupies an authorized device, we cannot detect the intrusion by using the device identification technologies. So it is of great practical importance to identify user across multiple devices. To this end, we try to analyze the user's behavior habits and generate fingerprints, which are constituted by the device-independent Web access records, DNS domain name information, and HTTP User-Agent field. Except for the feature selection, other steps are similar to those of device identification. The specific identification procedures and verification steps go as follows.

6.2. Feature Selection and Fingerprint Generation. The user's network behavior habits are mainly reflected by his Web access records, and the attributes such as operating system and browser in HTTP User-Agent can also reflect the user's preferences to some extent. Therefore, we use the Web access records extracted from the application protocol identification unit, the mapping relationship between IP addresses and domain names obtained from the DNS analysis results, and



(a) Setting the minimum effective feature dimension to 1

(b) Setting the minimum effective feature dimension to 4

Figure 7: The Effect of sliding window on the accuracy of online device identification.

the information about the types of browsers, versions, and operating system types achieved through HTTP User-Agent detection, to generate user's online behavioral fingerprints for the identification of network users.

The behavioral fingerprint vector is generated by extracting features from captured traffic in a unit time. For the target IP address in the Web access record, we associate it with the domain name based on the DNS response records and treat all IP addresses and subdomains under the same domain name as one attribute of the vector.

After the features of domain names are determined, combined with the information contained in the HTTP User-Agent field, a network user's behavioral fingerprint is generated, which constitutes the fingerprint vector of the user behavior in a unit time.

The dimension of the user behavioral fingerprint vector is 57,593. According to the value type of feature attributes, behavioral fingerprints can also be divided into two types: boolean and numerical type. However, we can see from the fingerprint classification results of the device that there is no obvious difference between the classification accuracy of the two types of fingerprints, and the classification accuracy of Boolean type device fingerprinting is slightly better than that of numerical type fingerprinting. Therefore, we will test the identification accuracy of network user with Boolean type behavioral fingerprinting.

6.3. Training and Verification of Offline Model. Since the overall dimension of the behavioral fingerprint vector is large, we select Multinomial Naive Bayes and random forest to perform and compare their performance to select the better one for online identification of network users.

The network traffic collected in this paper contains data of 118 users. The data collection procedure lasts for 53 days. Each fingerprint is generated through extraction of each user's network data per hour. Note that the fingerprints with a full list of zero feature values are discarded. Altogether, we get a total of 54107 fingerprints. The overall fingerprints are categorized into two groups. One group contains 32,217

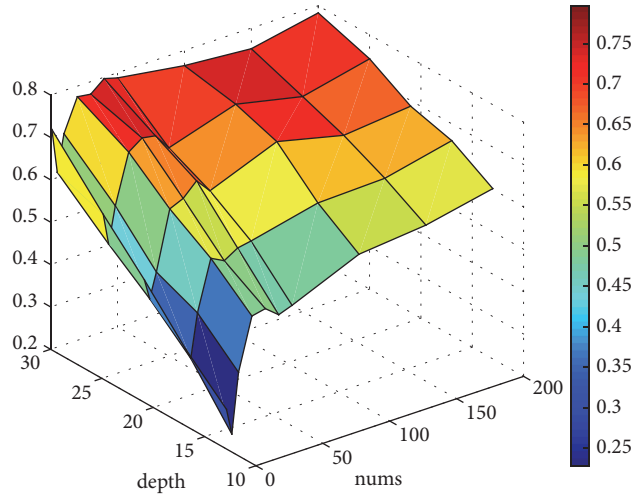


Figure 8: The classification accuracy of behavioral fingerprints using random forest model.

behavioral fingerprints collected in the first 30 days, used for the training and verification of the offline models. The other group includes the remaining 21,890 behavioral fingerprints collected in the following 23 days, used for testing the identification accuracy of network users.

Moreover, when training an offline model, the first group of the behavioral fingerprints are randomly divided into the training and verification sets, of which 70% of the behavioral fingerprints are used as a training set for model training, and the remaining 30% are used for verifying. The rest of the behavioral fingerprints are gathered as a test set for evaluating the classification accuracy. Two offline models, Naive Bayes and Random Forest, are trained with the same allocation of data sets.

First, the random forest model is trained. Its *nums* and *depth* parameters are taken into account. The obtained classification accuracy from training and testing under different values of *nums* and *depth* is shown in Figure 8. As can be seen

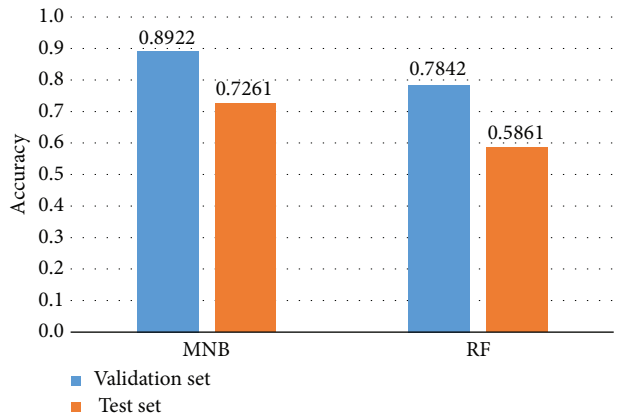


Figure 9: Behavioral fingerprints classification accuracy using MNB and RF.

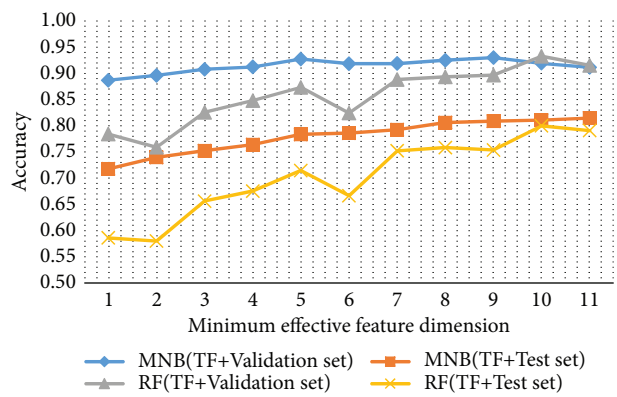


Figure 10: The effect of the minimum effective feature dimension on the user classification accuracy.

from the figure, when the *nums* is 40 and the *depth* is 30, the random forest model achieves the best classification accuracy.

Then the Multinomial Naive Bayes classification model is trained, and the classification accuracy is evaluated by the validation set and the test set, respectively. Figure 9 shows the classification accuracy of the validation set and the test set by performing the two models. These results show that the random forest model is far worse than the Multinomial Naive Bayes model for both data sets in terms of classification accuracy.

Finally, the effect of the minimum effective dimension on the classification effect and the ratio of valid behavioral fingerprints are tested. The results are shown in Figure 10 and Table 2, respectively.

As can be seen from Figure 10, the positive effect of the Multinomial Naive Bayes model on the test set is gradually enlarged as the minimum effective dimension increases. When the minimum effective dimension is set to 3, the classification accuracy rate of the test set is already higher than 75%. And when the minimum effective dimension is 9, the classification accuracy of the test set is the highest, reaching 80.70%, which can cover 74.87% of behavioral fingerprints.

Table 2: The effect of the minimum effective feature dimension on the number of behavioral fingerprints.

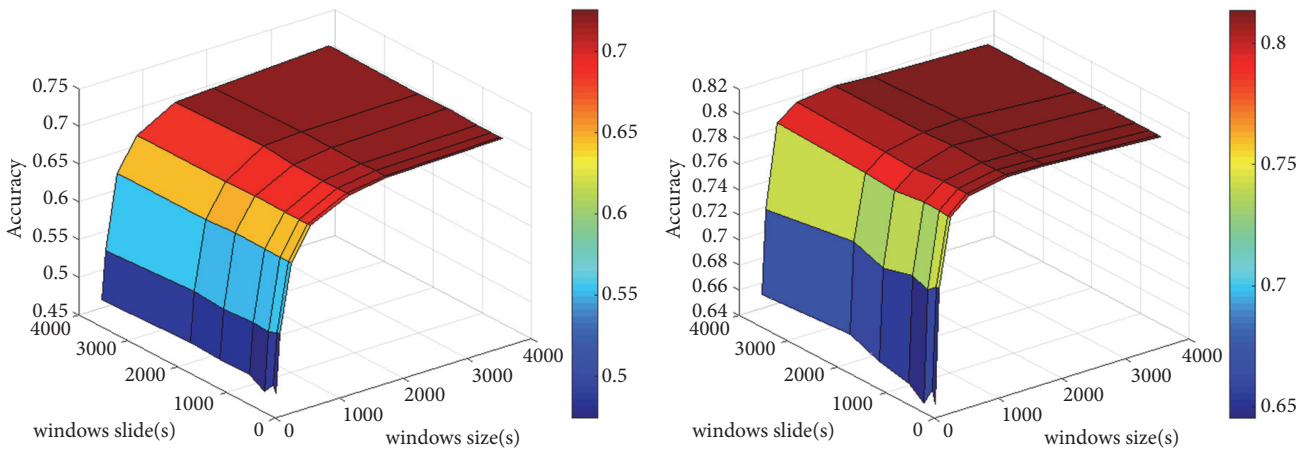
Minimum Effective Feature Dimension	Validation Set	Test Set
1	100.00%	100.00%
2	92.39%	93.70%
3	91.77%	90.54%
4	89.34%	86.72%
5	87.98%	84.22%
6	86.53%	81.03%
7	83.18%	78.36%
8	80.45%	76.42%
9	79.03%	74.87%
10	77.07%	73.43%
11	73.67%	72.16%

In comparison, the classification effect of the random forest model is not only relatively poorer, but also not stable enough. Therefore, we use the Multinomial Naive Bayes classification algorithm to implement the online user identification. Moreover, after comprehensively considering the effect of the minimum effective dimension on the classification accuracy and the coverage of the behavioral fingerprints, the value of 9 is selected as a condition to filter the invalid behavioral fingerprints.

6.4. Online User Identification. The online identification of network users is also performed in the sliding window manner, and the online process is simulated by replaying the real network traffic in the test set for 23 days. The step size and the window size of the sliding window are varied to figure out their effects on the user identification accuracy. In this paper, the values of step size are set to be 1 minute, 2 minutes, 5 minutes, 10 minutes, 20 minutes, 30 minutes, and 60 minutes and so are the values of window size. Figure 11 shows the results when the minimum effective dimensions are 1 and 9, respectively.

As can be seen from the figure, the step size of the sliding window has very little effect on the accuracy of online user identification. When the minimum effective dimension is 1, the accuracy goes up as the size of sliding window increases. When the window size is 60 minutes, the identification accuracy rate reaches a maximum of 72.58%. And the experimental results also show that when the window size of the sliding window is 20 minutes, the identification accuracy rate has already reached 71.42%. Thus, the time window size of the online user identification can be controlled within 20 minutes.

When the minimum effective dimension is 9, the identification accuracy rate firstly increases with the increase of the sliding window. When the window size increases to 5 minutes, the accuracy remains basically unchanged at 79.51%. When the window size is 20 minutes, the accuracy reaches 81.37%. And when the window size is 60 minutes, the rate



(a) Setting the minimum effective feature dimension to 1

(b) Setting the minimum effective feature dimension to 9

Figure 11: The effect of sliding window on the accuracy of online user identification.

Table 3: The results of high-speed packets capture (pps).

Packet length (Byte)	Number of physical cores				
	1	2	4	8	12
100	1329465	1566371	3206383	5936376	6272835
200	1130613	1734582	2917599	5352853	5586074
500	922524	1155344	2384589	2385555	2402127
1000	852867	1219714	1219109	1230260	1229943

is 80.74%. Therefore, the time window size of online user identification can be further shortened to 5 minutes.

7. Performance Tests and Results

7.1. Test Environment. In the above we have evaluated and proved the effectiveness of device identification and user identification with different algorithms and parameters, respectively. This section mainly tests the performance of the identification methods. The test environment is as follows:

7.1.1. Hardware

(i) 1 master node: Dell PowerEdge R730 (CPU: 2 6-core E5-2620V2, 2.1GHz, memory: 96 GB, external storage: 3.6TB).

(ii) 14 slave nodes: Dell PowerEdge C6220 II (CPU: 2 6-core E5-2620V2, 2.1GHz, memory: 64 GB, external: 8TB).

(iii) NIC: Intel 82599ES 10-Gigabit, supports up to 64 hardware queues.

7.1.2. Operating System

(i) Operating system: Red Hat Enterprise Linux 7.

(ii) Kernel version: 3.10.0-123.20.1.el7.x86_64.

7.2. Test Results. This section tests the performance of three modules: packets capture, distributed packets processing, and online identification modules. The test results are illustrated as follows:

7.2.1. Packets Capture Rate. First, network traffic is generated by the tcpreplay tool [34] and the packets capture rate is tested on the Intel 82599ES 10-Gigabit NIC. The NIC supports a maximum of 64 hardware queues and the number of hardware queues can be configured freely as required. However, due to the fact the CPU in our experimental computing node only has 12 physical cores and that the packet capture speed cannot be greatly enhanced if multiple packet capture threads are located in the same physical core, at most 12 hardware queues are enabled in this experiment.

In this paper, the speed of traffic capture is tested for the packets with different lengths and number of NIC hardware queues, separately. The results obtained are shown in Table 3. The results are acquired by calculating the average total numbers of packets that multiple NIC hardware queues capture within 10 seconds. The corresponding packet capture rate is shown in Table 4. From Tables 3 and 4 we can conclude that the packet length has a great influence on the capture rate. When the packet length is 1000 bytes, the ultimate speed of the NIC (9.76Gbps) can be reached by just consuming two hardware queues. If the packet length is reduced to 100 bytes, the packet capture rate rises as the number of NIC hardware queues increases till reaching a peak speed

Table 4: The results of high-speed packets capture (Gbps).

Packet length (Byte)	Number of physical cores				
	1	2	4	8	12
100	1.06	1.25	2.57	4.75	5.02
200	1.81	2.78	4.67	8.56	8.94
500	3.69	4.62	9.54	9.54	9.61
1000	6.82	9.76	9.75	9.84	9.84

Table 5: The speed test results of distributed processing framework.

The Number of Kafka Partitions	Maximum Processing Speed
1	3.76Gbps
2	6.68Gbps
3	10.01Gbps
4	13.35Gbps

Table 6: The time consuming of online identification.

#	Time consuming of online identification (ms)
1	9074
2	8256
3	7480
4	8188
5	6566
6	6780
7	6643
8	6503
9	8047
10	6080

of 5.02Gbps. When the length is changed to 200 bytes, the maximum capture rate is 8.94Gbps. When the packet length is 500 bytes, the maximum packet capture rate is 9.61Gbps.

The experimental results above show that the use of Linux PFQ kernel module can capture packets with a high speed and has robust systematic extensibility.

7.2.2. Speed of Distributed Processing of Packets. When testing the speed of the distributed processing framework, this paper uses KafkaProducer to write the network traffic captured by the packet capture module into each Kafka partition and then calculate the framework's processing speed of reading and analyzing data from Kafka partitions.

The number of KafkaSpouts is consistent with the number of Kafka partitions, which has a crucial influence on the speed of the distributed processing framework. Table 5 shows the speed of the distributed processing framework under different Kafka partition numbers. As we can see, the maximum processing speed is basically proportional to the number of Kafka partitions. It is noteworthy that it exceeds 10Gbps when the Kafka partition number is 3.

7.2.3. Response Speed of Online Application Identification. This paper uses the maximum window size of 60 minutes to test the response speed of the online identification module. This module contains two parts: online device identification based on the runtime environment and online user identification based on network behavioral fingerprints. Through the statistics of the time consumption of online identification modules for 10 trials, the average value is calculated as the response speed of the online identification module. The time consumption of the 10 online identifications is collected in Table 6. By averaging them, the response speed is derived as 7362 ms. This value is much smaller than the minimum step size of the recognition window (1 minute), so it can be considered that the online identification processing speed can meet the performance requirement.

8. Conclusion

In the intrusion detection area, it is increasingly important to detect the suspicious entities and potential threats. In this paper, we introduce the identification technologies of network entities to detect the potential intruders. In order to achieve network entities identification in high-speed environment, we use PFQ kernel module to capture high-speed network packets and employ Storm distributed real-time streaming data processing technology to realize online analysis of network traffic.

For the unauthorized devices in the monitored network, we design an online device identification technology based on runtime environment analysis. 961 features, such as application program, operating system, and HTTP User-Agent field, are selected to constitute the device fingerprints. And then the logistic regression algorithm is applied in a sliding window manner. For the case that the intruder occupies an authorized device and disguises himself as an authorized user, we extract the Web access record, DNS domain name, and HTTP User-Agent field to constitute user behavioral fingerprints. And then users are identified online in a sliding window manner using the Multinomial Naive Bayes model. The experimental results show that the traffic analysis framework and identification methods proposed in this paper have a high practicality as they can achieve satisfying identification accuracy rates in a short time. For future research, we intend to design an automated application identification tool in order to identify a large scale of applications and enhance the identification accuracy.

Data Availability

The network traffic data used to support the findings of this study have not been made available because they contain a lot of privacy information.

Disclosure

Any opinions, findings, conclusions, and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by National Key R&D Program of China 2018YFB0803400 and 2017YFB1003000, National Natural Science Foundation of China under grants 61572130, 61502100, 61532013, and 61632008, by Jiangsu Provincial Scientific and Technological Achievements Transfer Fund BA2016052, by Jiangsu Provincial Key Laboratory of Network and Information Security under grants BM2003201, by Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under grants 93K-9, and by Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[1] G. Pedro T, D. Jes·sE V, M. Gabriel F, and V. Enrique, "Anomaly-based network intrusion detection: Techniques, systems and challenges," in *Computers Security*, pp. 18–28, 18–28, 28(1–2, 2009.

[2] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 303–336, 2014.

[3] J. Pang, B. Greenstein, R. Gummadi, S. Seshan, and D. Wetherall, "802.11 User fingerprinting," in *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*, pp. 99–110, ACM, September 2007.

[4] Y. Zhang, M. Yang, X. Gu, P. Pan, and Z. Ling, *Proceedings of the 2018 International Conference on Advanced Cloud and Big Data*, LanZhou, China, 2018.

[5] B. Danev, D. Zanetti, and S. Capkun, "On physical-layer identification of wireless devices," *ACM Computing Surveys*, vol. 45, no. 1, article 6, 2012.

[6] T. Kohno, A. Broido, and C. Claffy K, "Remote physical device fingerprinting," in *Proceedings of the 26th IEEE Symposium on Security and Privacy (SP'05)*, Berkeley, CA, USA, 2005.

[7] R. Gerdes, T. Daniels, M. Mina, and S. Russell, "Device Identification via Analog Signal Fingerprinting: A Matched Filter Approach," in *Proceedings of the 13th Annual Network and Distributed System Security Symposium Conference (NDSS'06)*, San Diego, CA, USA, 2006.

[8] E. D. Thomas, J. A. Van Randwyk, E. J. Lee et al., "Passive data link layer 802.11 wireless device driver fingerprinting," in *Proceedings of the 15th conference on USENIX Security Symposium*, Vancouver, B.C., Canada.

[9] T. Yen, Y. Xie, F. Yu, R. Yu, and M. Abadi, "Host Fingerprinting and Tracking on the Web: Privacy and Security Implications," in *Proceedings of the 19th Annual Network and Distributed System Security Symposium (NDSS'12)*, 2012.

[10] T. Stöber, M. Frank, J. Schmitt, and I. Martinovic, "Who do you sync you are?" in *Proceedings of the the sixth ACM conference*, p. 7, Budapest, Hungary, April 2013.

[11] P. Eckersley, "How unique is your web browser?" in *Privacy Enhancing Technologies: 10th International Symposium, PETS 2010, Berlin, Germany, July 21–23, 2010. Proceedings*, vol. 6205 of *Lecture Notes in Computer Science*, pp. 1–18, Springer, Berlin, Germany, 2010.

[12] K. Mowery, D. Bogenreif, S. Yilek, and H. Shacham, "Fingerprinting information in JavaScript implementations," in *Proceedings of 2011 Web 2.0 Security and Privacy (W2SP'11*, Oakland, California, 2011.

[13] J. Mayer and J. Mitchell, "Third-party web tracking: Policy and technology," in *Proceedings of the 33rd IEEE Symposium on Security and Privacy (SP'12)*, San Francisco, CA, USA, 2012.

[14] G. Acar, M. Juarez, N. Nikiforakis et al., "FPDetective: Dusting the web for fingerprinters," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security, CCS 2013*, pp. 1129–1140, Germany, November 2013.

[15] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "Cookieless monster: Exploring the ecosystem of web-based device fingerprinting," in *Proceedings of the 34th IEEE Symposium on Security and Privacy, SP 2013*, pp. 541–555, USA, May 2013.

[16] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna, "On the workings and current practices of web-based device fingerprinting," *IEEE Security & Privacy*, vol. 12, no. 3, pp. 28–36, 2014.

[17] P. Laperdrix, W. Rudametkin, and B. Baudry, "Beauty and the Beast: Diverting Modern Web Browsers to Build Unique Browser Fingerprints," in *Proceedings of the 2016 IEEE Symposium on Security and Privacy, SP 2016*, pp. 878–894, USA, May 2016.

[18] A. Kurtz, H. Gascon, T. Becker, K. Rieck, and F. Freiling, "Fingerprinting Mobile Devices Using Personalized Configurations," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 1, pp. 4–19, 2016.

[19] W. Wu, J. Wu, Y. Wang, Z. Ling, and M. Yang, "Efficient Fingerprinting-Based Android Device Identification with Zero-Permission Identifiers," *IEEE Access*, vol. 4, pp. 8073–8083, 2016.

[20] Y. Cao, S. Li, and E. Wijmans, "(cross-)browser fingerprinting via os and hardware level features," in *Proceedings of the 24th Annual Network and Distributed System Security Symposium, NDSS*, San Diego, CA.

[21] R. V. Yampolskiy and V. Govindaraju, "Behavioural biometrics: a survey and classification," *International Journal of Biometrics*, vol. 1, no. 1, pp. 81–113, 2008.

[22] N. Zheng, A. Paloski, and H. Wang, "An efficient user verification system via mouse movements," in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, pp. 139–150, ACM, October 2011.

[23] S. Douhou and J. R. Magnus, "The reliability of user authentication through keystroke dynamics," *Statistica Neerlandica. Journal of the Netherlands Society for Statistics and Operations Research*, vol. 63, no. 4, pp. 432–449, 2009.

[24] N. V. Verde, G. Ateniese, E. Gabrielli, L. V. Mancini, and A. Spognardi, "No NAT'd user left behind: Fingerprinting users behind NAT from NetFlow records alone," in *Proceedings of the 2014 IEEE 34th International Conference on Distributed Computing Systems, ICDCS 2014*, pp. 218–227, Spain, July 2014.

[25] B. Padmanabhan and Y. Yang, *Clickprints on the web: Are there signatures in web browsing data*, 2006, http://knowledge.wharton.upenn.edu/papers/1323.pdf.

[26] Y. Yang, "Web user behavioral profiling for user identification," *Decision Support Systems*, vol. 49, no. 3, pp. 261–271, 2010.

[27] M. Kumpošt and V. Matyáš, "User Profiling and Re-identification: Case of University-Wide Network Analysis," in *Trust, Privacy and Security in Digital Business*, vol. 5695 of *Lecture Notes in Computer Science*, pp. 1–10, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[28] D. Herrmann, C. Gerber, C. Banse, and H. Federrath, "Analyzing Characteristic Host Access Patterns for Re-identification of Web User Sessions," in *Information Security Technology for Applications*, vol. 7127 of *Lecture Notes in Computer Science*, pp. 136–154, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[29] C. Banse, D. Herrmann, and H. Federrath, "Tracking Users on the Internet with Behavioral Patterns: Evaluation of Its Practical Feasibility," in *Information Security and Privacy Research*, vol. 376 of *IFIP Advances in Information and Communication Technology*, pp. 235–248, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[30] D. Herrmann, C. Banse, and H. Federrath, "Behavior-based tracking: Exploiting characteristic patterns in DNS traffic," *Computers & Security*, vol. 39, pp. 17–33, 2013.

[31] D. W. Kim and J. Zhang, "You Are How You Query: Deriving Behavioral Fingerprints from DNS Traffic," in *Security and Privacy in Communication Networks*, vol. 164 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pp. 348–366, Springer International Publishing, Cham, 2015.

[32] X. Gu, M. Yang, C. Shi, Z. Ling, and J. Luo, "A novel attack to track users based on behavior patterns," *Concurrency and Computation: Practice and Experience*, 2016.

[33] Blockmon, "cnplab/blockmon," https://github.com/cnplab/blockmon.

[34] Tcpreplay, "Tcpreplay development is now being done by AppNeta," URL http, http://tcpreplay.synfin.net.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 1420930, 10 pages
https://doi.org/10.1155/2018/1420930

Research Article

Semantic Contextual Search Based on Conceptual Graphs over Encrypted Cloud

Zhenghong Wang,[1] **Zhangjie Fu**,[1,2] **and Xingming Sun**[1]

[1]*School of Computer and Software, Nanjing University of Information Science & Technology, Nanjing, China*
[2]*College of Information Science and Technology, Jinan University, Guangzhou 510632, China*

Correspondence should be addressed to Zhangjie Fu; wwwfzj@126.com

Received 15 July 2018; Accepted 8 November 2018; Published 2 December 2018

Guest Editor: Xuyun Zhang

Currently, searchable encryption becomes the focus topic with the emerging cloud computing paradigm. The existing research schemes are mainly semantic extensions of multiple keywords. However, the semantic information carried by the keywords is limited and does not respond well to the content of the document. And when the original scheme constructs the conceptual graph, it ignores the context information of the topic sentence, which leads to errors in the semantic extension. In this paper, we define and construct semantic search encryption scheme for context-based conceptual graph (ESSEC). We make contextual contact with the central key attributes in the topic sentence and extend its semantic information, so as to improve the accuracy of the retrieval and semantic relevance. Finally, experiments based on real data show that the scheme is effective and feasible.

1. Introduction

1.1. Background. In 2000, digital storage accounted for only 1/4 of the world's data, and another 3/4 of the information was stored in newspapers, books, and other mediums. But by 2020, digital information will account for 4/5 of global data and will reach 40ZB, which is equivalent to 5200GB of data generated by each person. The consumption of the local storage of the users is too expensive. So in order to save storage costs of data, users usually choose to upload data to the cloud. However, public clouds are not always trusted, so the data always is encrypted before uploading to the cloud servers, which also makes the traditional plaintext search scheme invalidated. Thus, how to better protect and utilize user privacy in cloud computing has become a major research issue in mobile cloud computing.

Searchable encryption of the cloud server has become an important field of investigation in recent years. One of the most popular methods of traditional schemes is keywords-based search. The data owner first extracts the corresponding keywords for the data documents and builds the corresponding index and then outsources the encrypted documents and index to the cloud server. When searching for the encrypted data, the cloud server can match the trapdoor with the encrypted index; then the corresponding data documents are returned to the data user. But, as we know, there are some deficiencies with the above keywords-based schemes, which cannot reflect the user's search intention and the semantic information of the document.

In the keyword-based encryption search schemes, the data owner summarizes a document's content into some keywords, which can make search matching efficient and simple. However, the keyword cannot represent the contents of the data document well; it ignores the semantic information of the document. Thus, the returned search results from cloud server are not always matching with the requirement of the user's query. Although the keywords-based schemes [1, 2] have a semantic extension of the keywords, they still cannot overcome the limitations of the keywords. Thus, we research content-based searchable encryption scheme. The scheme [3] takes into account the central content of the text, which expresses the document content with a topic sentence, then establishes the conceptual graph for the topic sentence, and builds the corresponding encrypted index structure. Unfortunately, the scheme does not consider

context-sensitive semantics. Thus, the scheme still has a lot of defects.

Therefore, under protecting the security of user privacy in the cloud environment, in order to improve the relevance of documents information obtained by encrypted search, we proposed a searchable encryption scheme which combined the local features with the context similarity.

1.2. Main Contribution. In the paper, we propose a semantic search encrypted scheme based on conceptual graphs of context (ESSEC). We still extract the central content of the whole document as the index rather than keywords and then construct the corresponding weighted conceptual graph [4] for the topic sentence:

(i) We extend the context-based semantics of the center concept attribute, so that the generated conceptual graph can contain the content information of the document and constructs the semantic network of the conceptual graph, which helps to make search results satisfy the needs of users' retrieval as much as possible.

(ii) The experiments based on real datasets have been implemented, and the experimental contrast diagrams make clear that the two schemes put forward in this paper are effective and feasible.

2. Related Work

Searchable encryption [2, 3, 5] is cryptographic primitives developed for data's encrypted search. The symmetric key searchable encryption scheme was first proposed by Song et al. [6]. Subsequently, the early researchers Golle and Ballard et al. [7–9] have proposed the schemes to support multikeyword search in different application scenarios. Returns related documents based on whether the keywords are contained in the document. However, the earlier proposed schemes are only applicable to small-scale specific types of applications and ignore the semantic information of documents.

Cao et al. [10] first defined and solved the problem of multikeyword classification retrieval on encrypted cloud data (MRSE). In the scheme, Cao creatively uses intrinsic product similarity and coordinate matching to compute the correlation between keywords and files and put forward the two different threat models. The first model is a known ciphertext model and other is the known background model. Then, [11–13] have the further study on the basis of MRSE.

Then, the scholars have put forward many excellent schemes based on semantic searchable encryption [14–18]. Li et al [14] first use the wildcard technology and editing distance to construct a fuzzy semantic keyword set. Fu et al. [15] construct wordnet tree to expand its semantics for keywords. Then, [16] was based on the NLP analysis of the input multiple keywords to obtain the weight of each keyword to represent the interest of the user and expand the semantics by extending the central keyword to improve the efficiency and accuracy. However, taking into account the deficiencies of the keywords, literature [19] constructs a content-based semantic searchable encryption scheme, which uses the semantic representation tool of the conceptual graph to store the content information of the document, thereby implementing semantic retrieval. [18] proposes a verifiable diversity ranking search scheme over encrypted outsourced data. In our scheme, we still use the conceptual graph as our semantic expression tool, but we take into account the contextual semantic content when constructing conceptual graph, in order to construct a semantic network, increasing the retrieval accuracy.

3. Problem Formulation

3.1. System Model and Threat Models. The system model considered in this paper is shown in Figure 1: the data owner, data user, and the cloud server are 3 entities of the system. To keep the data private, before uploading the documents $F = \{F_1, F_2, ..., F_n\}$ to the cloud server, the data owner would encrypt the data $C = \{C_1, C_2, ..., C_n\}$. Meanwhile, to retrieve encrypted data, the data owner needs to generate a searchable encryption index I. In our scheme, we generate a conceptual graph index for encrypted documents. Finally, both index and documents will be encrypted and upload to the cloud.

The data users need to obtain the authorization from the data owner. Then they need to generate request trapdoor (conceptual graph) η for query sentence, which will be encrypted upload to the cloud server. And the cloud server matches the encrypted index I with the encrypted trapdoor η. Finally, the cloud will return the related encrypted documents to the data user. The data user would decrypt the encrypted documents.

In our scheme, we think the cloud server is "honest but curious." In other words, the cloud server can comply with the protocols, but it still hopes to obtain more sensitive information through learning and guessing. In this paper, we only focus on how the cloud can deal with the similarity search over the encrypted data, which is the same as the model adopted by previous literature [10].

3.2. Notations and Preliminaries

3.2.1. Notations

(i) F: the plaintext document dataset, $F = \{F_1, F_2, ..., F_n\}$, and each F_i can be summarized as a CG.

(ii) C: the ciphertext document dataset, $C = \{C_1, C_2, ..., C_n\}$.

(iii) CG: conceptual graph

(iv) Q: the query represented by two vectors and a hash table, defined as a collection $Q = \{Q_1, Q_2, QM_3\}$.

(v) QM: the hash structure in query.

(vi) $F(Q_1)$: the encrypted set of documents in F whose D_1 is similar with Q_1.

(vii) D_{ij}: the index composed of vectors and hash table, which is defined as $D_i = \{D_{i1}, D_{i2}, D_{i3}, M_4\}$.

3.2.2. Preliminaries. **Conceptual Graph:** Sowa first proposed the conceptual graph scheme [20], which is the model of

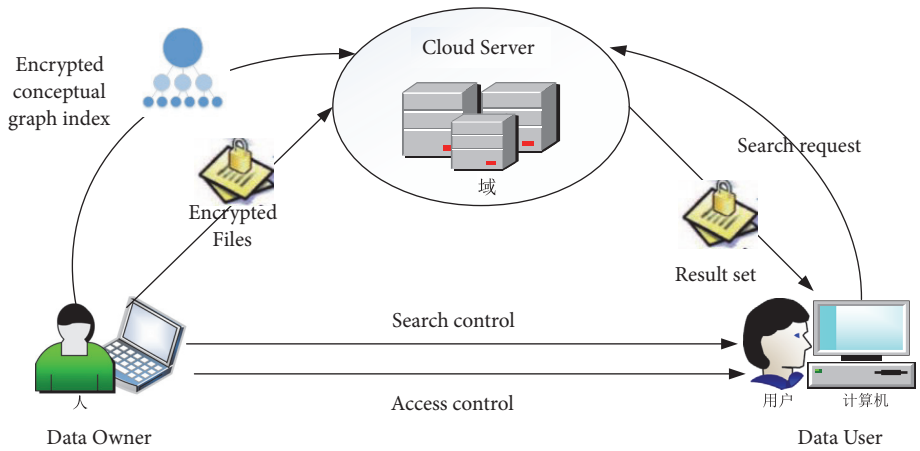


Figure 1: System model.

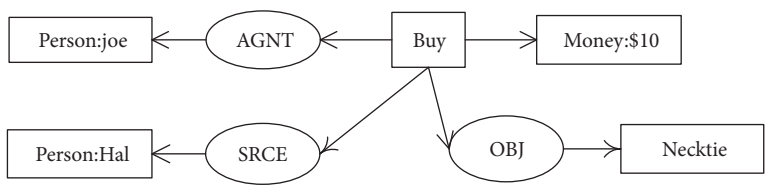


Figure 2: Conceptual graph: joe buys a necktie from Hal for $10.

semantic knowledge representation, similar to a knowledge graph. It is the structure of knowledge representation based on first-order logic [21]. As a logical model, conceptual graph can be used to describe any content that can be implemented on the digital computer. It usually has two types of nodes: concepts (rectangles) and conceptual relationships (also known as semantic roles) (Ellipse) (Figure 2). At the same time, for each concept, there are two parts: the left is a type label, which represents the type of entity, and on the right is a concept attribute value, but its concept type does not necessarily exist. Each concept is associated with other concepts. And there are about 30 relationships and 6 tenses.

TF-IDF(term frequency–inverse document frequency): is a statistical method used to reflect how important a word is to a document or corpus [1]. The TF-IDF value increases proportionally to the number of times a word appears in the document and is offset by the frequency of the word in the corpus. Term frequency (TF) refers to the frequency of a given word in the file. Inverse document frequency (IDF) is a measure of the universal importance of a word. And the TF–IDF is the product of two statistics, term frequency and inverse document frequency.

$$\frac{TF}{IDF} = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{|D|}{\left|\left\{j : t_i \in d_j\right\}\right|} \quad (1)$$

Text summarization: Text summarization always tries to determine the central content of documents. And the methods of automatic text summarization are mainly divided into two categories: extractive and abstractive. The extractive summarization is based on the assumption that the core idea of a document can be summarized in one sentence or a few sentences in the document. In this paper, we first preprocess the document and make it a clause. Then the words and sentences are expressed as vectors (word2vec) that the computer can understand. And the sentences are sorted by the following models.

(1) Bag Of Words [22]: The bag of words model defines a word as a dimension, and a sentence is represented as a high-dimensional sparse vector in the space where all words are formed.

(2) Word Embedding [23]: Through word2vec and other techniques, get the low-dimensional semantic vectors of words, sentences, and documents.

3.3. Design Goals. Taking into account the above system model and to solve the problem of neglecting context semantics in the model, the following design goals will be achieved.

(i) Data privacy: our privacy goal is to prevent the cloud learn private information from the outsourced data, the corresponding index, the user queries, and search results.

(ii) Concept attribute access privacy: the cloud cannot know which concept attribute is focused queried and extended.

(iii) Context semantic search: the goal of our scheme is to take context semantic information into consideration in building conceptual graph to achieve more accurate search.

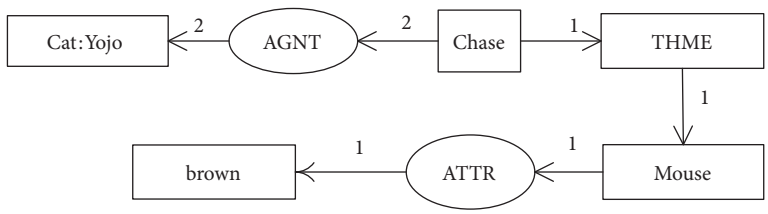


Figure 3: Weighted Conceptual graph.

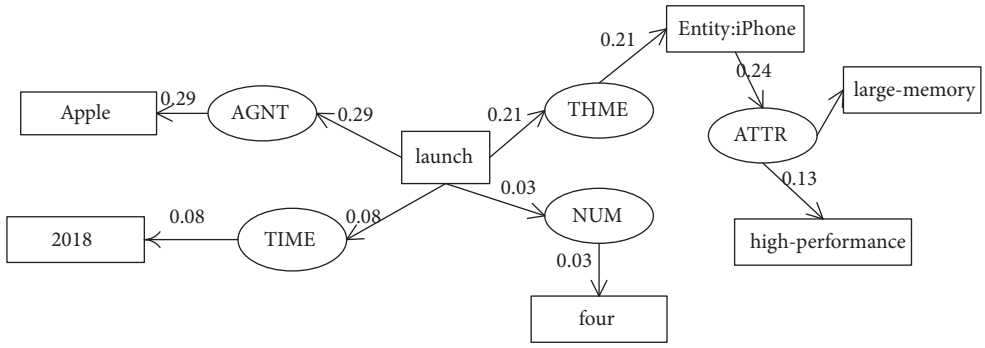


Figure 4: Weighted Conceptual graph for the subject sentence.

4. The Proposed Schemes

The searchable encryption scheme [10] ignores the semantic information of the context during the construction of the index; thus the accuracy of the search matching can be lost. We reconstruct the context-sensitive searchable encryption scheme based on conceptual graphs. First of all, we need to summarize the content of the document. According to the scheme in Section 3.2.2, we can get a topic sentence from the document abstract, and then we establish the corresponding conceptual graph [11].

In our scheme, considering the efficiency of the contextual semantic extension, we only extend the semantic information of the most important topics and construct a semantic network based on conceptual graph of document and then establish a corresponding encrypted index.

4.1. Our Basic Idea. In this section, we will detail our index construction scheme.

4.1.1. Weighted Conceptual Graph. We first introduce the weighted conceptual graph [24], which can help us analyze the importance of each concept attribute in the topic sentence, and reflect the theme of the document. In our scheme, both edges and nodes have weights. And edges' weights are assigned according to the relevance of the semantic flow in the concept relationship as shown in Figure 3.

In our idea, the initial importance of each concept should be the equal. Then we define it as follows.

Definition 1. The more times a concept type appears in a document or more grammatical relations between its conceptual type and other key attributes, the more important it is.

So after we have extracted the central sentence and constructed corresponding conceptual graph, we get all its concept attribute values (rectangular) and we calculate the term frequency (each sentence is considered as a document) and the document frequency of the concept attribute value in its sentence. We use the algorithm to get our weight. We represent the concept value in the concept map as its corresponding weight.

$$q_i = \frac{TF}{IDF(ca_i)} = DN \times \left(\frac{\log(1+tf)}{\log(df)} \right) \tag{2}$$

Thus, we can effectively obtain topic attributes by statistically weighting concepts, which helps us to extend its context-sensitive semantics. Suppose we obtain the subject sentence of the document: "Apple will launch four high-performance and large-memory iPhones in 2018." Our weighted conceptual graph for the sentence is shown in Figure 4.

4.1.2. Context-Sensitive Expansion of Central Attributes. For the topic sentence of Figure 4, we obtain the theme concept attribute which is "apple," but computer cannot know whether it represents the name of the fruit or the name of the enterprise, which leads to error easily when it is extended semantics and synonyms. So we need to have context-based semantic extensions for the central key attributes.

Our context-sensitive semantic expansion scheme is based on the assumption that the frequent-common attribute in the document has statistical relevance for the same topic. Therefore, we can reflect the connection relation of attributes by statistically analyzing the contextual relationships from the document collection.

We have the following definitions.

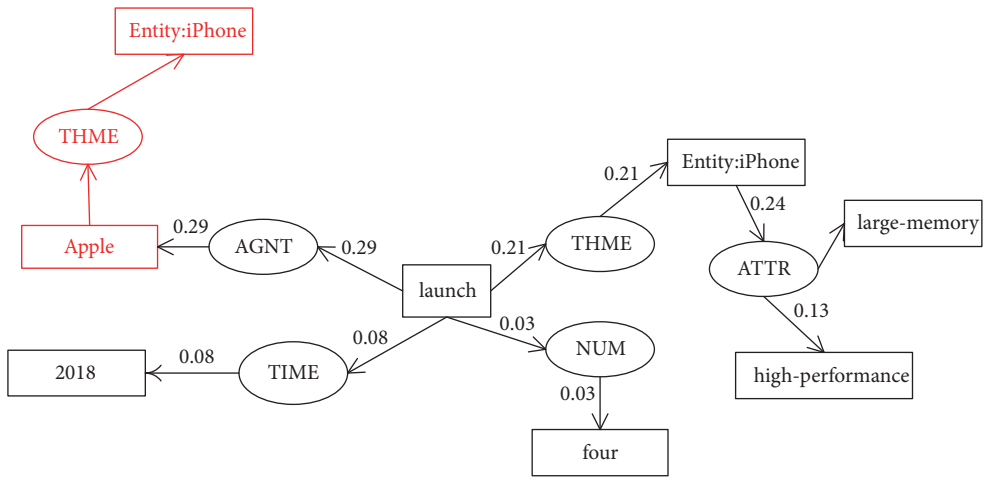


FIGURE 5: Context-based extension conceptual graph.

Definition 2. The vector of the concept attribute $attr_i$ is represented by

$$attr_i = \langle w_{1j}, w_{2j}, \cdots, w_{nj} \rangle . \quad (3)$$

n indicates the number of words which are cooccurring with $attr_i$ at least once in the document, and w_{kj} represents the word-to-word weight of word t_k to word $attr_j$.

In our scheme, we define that extended words and key attributes must belong to the same sentence. And it is generally believed that the closer the word to the key attribute in the document, the more times the word appears around the keyword,

Definition 3. Relevance between concept attribute and the word:

$$P(attr \mid t_j) = w_{kj} \times \frac{e^{-\lambda d(attr, t_j)}}{\sum_j e^{-\lambda d(attr, t_j)}} . \quad (4)$$

λ is the influence factor, and $d(attr, t_j)$ represents the distance between $attr$ and t_j.

When we calculate the relevance of all the extended words, then we need to calculate the relevance of the extended words to the subject sentence.

Definition 4. The relevance of the word t_j to the key sentence A:

$$R(t_j, A) = \sum_{ai \in A} P(t_j \mid a_i) \quad (5)$$

Q is a set of all the different concept attributes in the key sentence.

When selecting the extended word, we need to calculate the relevance of the word and the key attributes. At the same time, through Definition 4, we can get an extended attribute which is most relevant to the content of the entire topic sentence. Our scheme extends the semantics based on the context of concept attributes. For example, for Figure 4, we extend its context semantics in Figure 5.

Similarly, for the user's query sentence, we also need to construct a corresponding conceptual graph. And in order to return the search results which best match the user's search intent, our paper adopts the method of [18] to construct a user interest model with semantic information on the user's input topic through the wordnet synset.

4.1.3. Index and Trapdoor Constructions. After we obtain the context conceptual graph, we need to construct corresponding index structure, which can store all semantic information of conceptual graph. We take Figure 5 as an example to illustrate our construction scheme.

First, we design two vectors for the index. The first vector is mainly used to match the semantic structure in the query request. The second vector is used to store the weight of the semantic role, so that we can know the theme of the document. In our scheme, we ignore the conceptual type information in the conceptual graph because it is dispensable in our semantics. Meanwhile, we need to construct a hash table to store the corresponding concept attribute values. For the extended concept attributes, we only need to store it in the corresponding vector, so that the semantic information of the entire conceptual graph can be completely stored through our index structure.

The construction process is as follows:

$$D_1[j] = \begin{cases} |c_j|, & |c_j| > 0 \\ 0, & |c_j| = 0 \end{cases} \quad (6)$$

$$DW_1[j] = \begin{cases} \sum_{|c_j|} q_i, & |c_j| > 0 \\ 0, & |c_j| = 0 \end{cases} \quad (7)$$

$$DM_1[j] = \begin{cases} \forall (c_{ij}, r_i); & |c_{ij}| > 0 \\ null, & |c_{ij}| = 0 \end{cases} \quad (8)$$

For the first vector D_1, if the conceptual graph CG contains a semantic role r_i and it has $|c_j|$ number concept

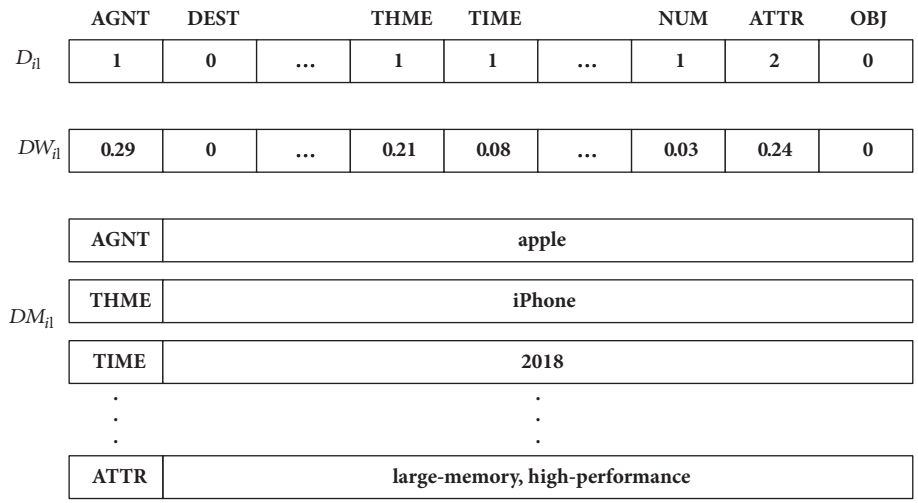


Figure 6: The index structure.

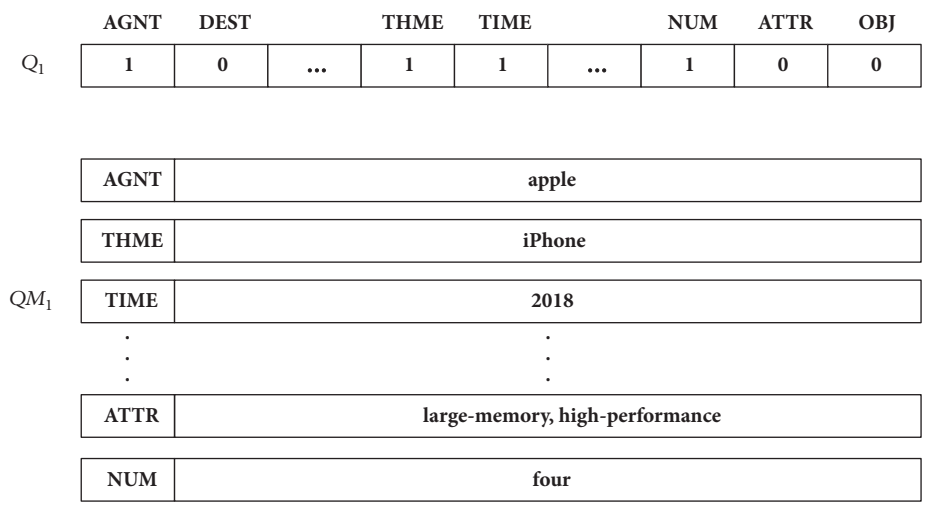


Figure 7: The trapdoor structure.

attributes ($|c_j|$ represents the number of concept attributes), $D_1[j] = |c_j|$. For second vectors DW_1, the weight of each semantic role $DW_1[j]$ is equal to the sum of all the weights of the concept attributes. Meanwhile, we construct the hash table $DM_1[j]$ to store the corresponding concept attribute values. The key is to store the corresponding semantic role; then its value can store its corresponding concept attribute values. The index structure is shown in Figure 6.

Similarly, we can also generate corresponding conceptual graph for user-entered query sentence and also construct corresponding trapdoor structures. For example, the user enters a query statement: "Apple tipped to launch four iPhones in 2018." We get its trapdoor structure as shown in Figure 7.

4.1.4. Retrieval Calculation. Then, we give our retrieval scheme. The data user generates a vectors and hash table Q_1, QM_1 based on the query sentence. The cloud server first calculates the inner product of D_1 vector and Q_1 vector and multiplies the weight vector DW_1 of document semantic role to select the N documents set with the largest correlation score. Then, the cloud server will match DM_1 and QM_1, that is, whether the corresponding semantic roles have corresponding concept attribute values, and calculate the final score so as to filter out the most relevant K documents from the N documents set. As shown in Algorithm 1 *score* is the threshold for $R(D_1, Q_1)$.

4.2. ESSEC Scheme. We use the MRSE framework [10, 25] to construct a searchable encrypted ESSEC scheme based on context-sensitive conceptual graph. At the same time, we combine submatrix techniques to reduce the encryption time in conjunction with the [11] scheme. The encryption scheme contains four steps: KeyGen, BulidIndex, Trapdoor, and Query. By calculating the cosine distance between the two vectors, we can get the similarity score, described as follows.

KeyGen: The data owner first constructs a secret key SK, generating two $(n+2) \times (n+2)$ invertible matrices M_1, M_2

Algorithm 1 RCG

(1) Input: $F, D_1, DW_1, DM_1, Q_1, QM_1$
(2) Output: $F(Q)$
(3) For each document F_i in F do
(4) If $R(D_{i1}, Q_1) = D_{i1} \cdot Q_1 > score$ then
(5) Calculate DW_{i1}, DM_{i1} and QM_1
(6) $R(DW_{i1}, Q_1) = DW_{i1} \cdot Q_1$
(7) $R(DM_{i1}, QM_1) = \forall(\exists Val(DM_{i1})\ in\ QM_1)$
(8) Insert a new element $(R(DW_{i1}, Q_1), R(DM_{i1}, QM_1), FID)$ into $RList$.
(9) Else return;
(10) End if
(11) End for

Algorithm 1: Retrieval algorithm.

which generated randomly and a $(n+2)$ bit vector S generated randomly, to form $SK = \{S, M_1, M_2\}$.

BulidIndex(D, DM): The scheme generates subindex $\overrightarrow{D}$ by applying dimension expansion and splitting procedures on D, which is similar to the secure KNN algorithm [10]. In this process, we generate two vectors $\{D_i', D_i''\}$. And we set the $(n+1)$-th dimensions in $\overrightarrow{D}$ to a random number ε; $(n+2)$-th dimensions is set to 1. Therefore, $\overrightarrow{D} = (D_i, \varepsilon_i, 1)$. Finally, the encrypted subindex $I_i = \{M_1^T \overrightarrow{D}'_i, M_2^T \overrightarrow{D}''_i\}$ for each encrypted document C_i. And the DM is encrypted by using $\pi(\bullet)$, which is an off-the-self hash function.

Trapdoor(Q, QM): The user generates a n-bit vector $\overrightarrow{Q}$ for query sentence. Then, a similar splitting process will be applied. We extend $\overrightarrow{Q}$ to $(n+1)$-dimension, and $(n+1)$-th is set to 1. Then scale it with a random number $r \neq 0$. And $\overrightarrow{Q}$ is extended to $(n+2)$-dimension. Therefore, $\overrightarrow{Q} = (rQ, r, t)$, t is random. The formulation of T_q is $\{M_1^{-1}\overrightarrow{Q}', M_1^{-1}\overrightarrow{Q}''\}$. Similarly, the QM is encrypted by using $\pi(\bullet)$.

Query: The cloud server calculates the encrypted trapdoor and encrypted index based on cosine measure. The final relevance score is

$$\begin{aligned}
&\cos(I_i, T_Q) \\
&= \cos\left(\left\{M_1^T \overrightarrow{D}'_{i1}, M_2^T \overrightarrow{D}''_{i1}\right\} \cdot \left\{M_1^{-1} \overrightarrow{Q}'_1, M_2^{-1} \overrightarrow{Q}''_1\right\}\right) \\
&= \cos\left(\overrightarrow{D}'_{i1} \cdot \overrightarrow{Q}'_1 + \overrightarrow{D}''_{i1} \cdot \overrightarrow{Q}''_1\right) \\
&= \cos\{(D_{i1}, \varepsilon, 1) \cdot (rQ_1, r, t)\}
\end{aligned} \tag{9}$$

Then cloud server can compare whether $\pi(DM)$ are the same as $\pi(QM_{i1})$ in document set $F(Q_1)$.

5. Performance Analysis

In essence, our proposed scheme is only some post processing further considered compared with the method in [19]. Therefore, the security of our scheme directly inherits the security of the method in [19]. In addition, we adopt the secure KNN inner product scheme [10].

In this section, to assess the feasibility of our scheme, we use java+stanfordNLP to build our experimental platform. Our implementation platform is Windows 7 server with Core CPU 2.85GHz. The dataset is a real-world dataset: CNN set (https://edition.cnn.com/) which is available to construct the outsourced dataset [26]. In our experiment, we use approximately 1000 documents.

5.1. Precision. Precision means that users can get what they want based on their queries' sentence. In our scheme, we expand the conceptual graph based on context semantic information. In order to achieve a balance between security and precision, we use 2 layers of index to store all the semantic information of the conceptual graph and also store the extended context semantic information. Thus, the retrieval accuracy of our scheme has a wider range of breadth and deeper precision.

5.2. Efficiency. In our scheme, we need to segment the documents of the dataset and remove the stop-word. We get topic sentences by word2vec, word-embedding, and other NLP methods, but we do not calculate the time, because the time is influenced by the corpus. Thus the time of index construction consists of 2 parts: one is to make a syntactic analysis of the subject sentence and the other is to construct the corresponding index and encrypt the index.

We can see in Figure 9 the relationship between the time of the index construction and the number of documents. Table 1 shows the required time cost and space cost for each index when the size of the document is about 1000. This is because our scheme needs to count the weights of the concept attributes and also needs to extend the context semantics of the central concept attributes. Thus, our index construction needs more time. Our scheme is different from the traditional keywords searchable scheme. We have taken into account the content of the document when constructing the index, which has greatly improved in accuracy and semantic aspects. Meanwhile, compared with the MRSE [10]

Table 1: Index Construction overhead for 1000 documents.

Scheme	Index vectors size	The time of index vectors for each file
MRSE [10]	12898KB	0.9s
USSCG [19]	8394KB	1.79s
Our	10738KB	1.84s

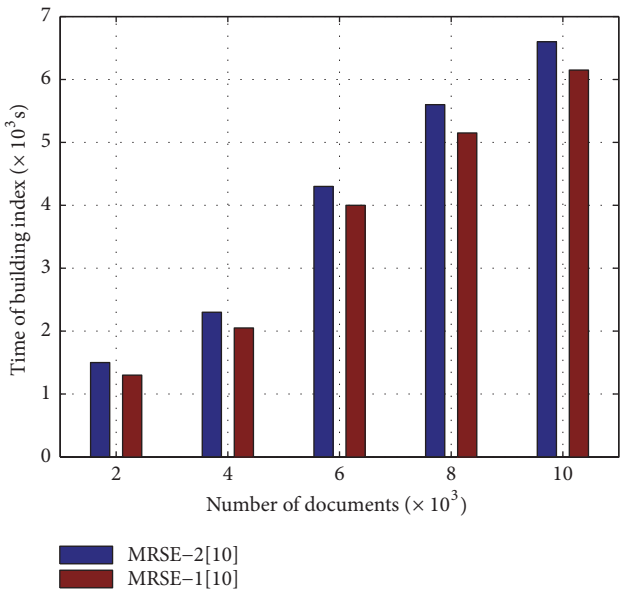


Figure 8: The time cost for generating index vectors in MRSE [10].

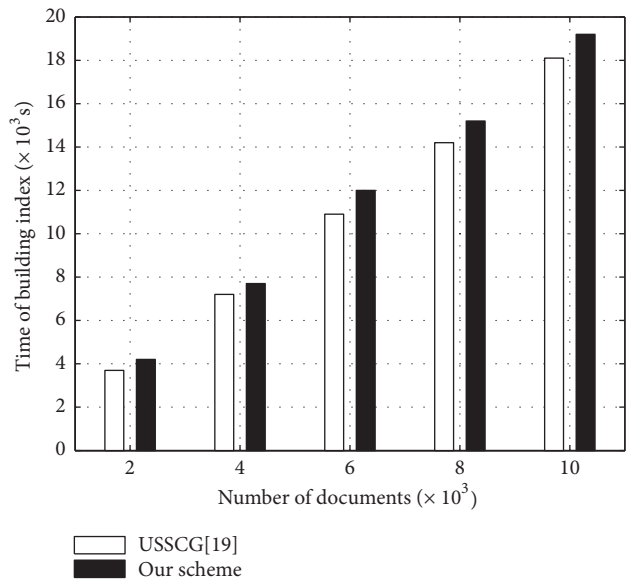


Figure 9: The time cost for generating index vectors.

index construction time (Figure 8), our scheme proved to be acceptable.

Figures 10 and 11 are the analysis diagrams of the relationship between the query time and the number of documents. It can be clearly seen that the query time is proportional to the number of documents, because the increase in the number of documents leads to an increase in the number of conceptual graphs indexes and the increase in the complexity of the context extension so that the query time eventually increases. Despite the results, our scheme has more time than MRSE [10] (Figure 10) and USSCG [19]. However, because our scheme carries the semantic information of the document content, we return more accurate results to compensate for the loss of efficiency.

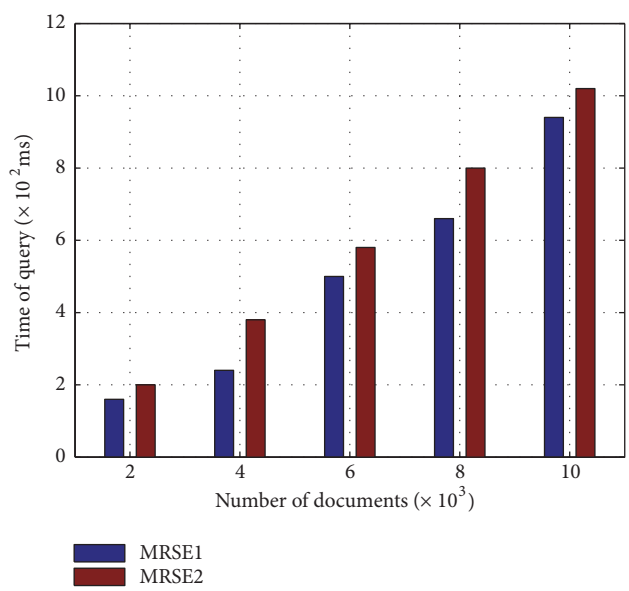


Figure 10: The time cost for query in MRSE [10].

6. Conclusion and Future Work

In this paper, for the first time, we take the relationship between semantic information of the context and conceptual graph into consideration, and we design a semantic search encryption scheme for context-based conceptual graph. By choosing the central key attributes in the topic sentence, not all attributes, our scheme performs a tradeoff between functionality and efficiency. To generate the conceptual graphs, we apply a state-of-the-art technique, i.e., word embedding and Tregex, a tool for simplifying sentences in our method. Also for the literature [10], we put forward a supplementary plan. When constructing the conceptual graph, we considered the semantic information of the context. By extending the context of the central concept attribute, we enhance the relevance of our semantic query and achieve a balance of precision and efficiency. Experimental results demonstrate the efficiency of our proposed scheme.

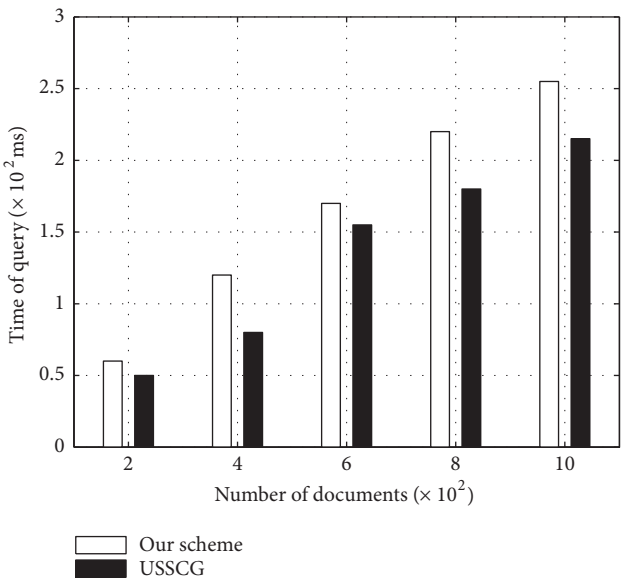


Figure 11: The time cost for query.

In the future, we will continue to focus our research on semantic searches using grammatical relations and other natural language processing. In addition, we are considering modifying the process of changing a conceptual graph into a numerical vector which can help improve accuracy and efficiency.

Data Availability

Our dataset is a real-world dataset: CNN set (https://edition.cnn.com/) which is available to construct the outsourced dataset [26]. And we construct the conceptual graphs by [24].

Conflicts of Interest

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

Acknowledgments

This work is supported by the National Natural Science Foundation of China under grant U183610040, 61772283, U1536206, U1405254, 61602253, 61672294, 61502242; by the National Key R&D Program of China under grant 2018YFB1003205; by China Postdoctoral Science Foundation (2017M610574); by the Jiangsu Basic Research Programs-Natural Science Foundation under grant numbers BK20150925 and BK20151530; by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund; by the Major Program of the National Social Science Fund of China (17ZDA092), Qing Lan Project, Meteorology Soft Sciences Project; by the Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET) fund, China.

References

[1] J. Leskovec, A. Rajaraman, and J. D. Ullman, "Mining of massive datasets," Cambridge University Press, 2011.

[2] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in *Proceedings of the International Conference on Applied Crygptography and Network Security*, pp. 442–455, 2005.

[3] B. Dan, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," *Lecture Notes in Computer Science*, vol. 3027, no. 16, pp. 506–522, 2004.

[4] S. Hensman and J. Dunnion, "Automatically building conceptual graphs using verbnet and wordnet," in *Proceedings of the International Symposium on Information and Communication Technologies*, pp. 115–120, 2004.

[5] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: improved definitions and efficient constructions," *Journal of Computer Security*, vol. 19, no. 5, pp. 895–934, 2011.

[6] 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, 2000*, pp. 44–55, IEEE, Berkeley, Calif, USA, May 2000.

[7] P. Golle, J. Staddon, and B. Waters, "Secure conjunctive keyword search over encrypted data," *Lecture Notes in Computer Science*, pp. 31–45, 2004.

[8] L. Ballard, S. Kamara, and F. Monrose, "Achieving efficient conjunctive keyword searches over encrypted data," *ICICS*, pp. 414–426, 2005.

[9] Y. L. Liu, H. Peng, and J. Wang, "Verifiable Diversity Ranking Search Over Encrypted Outsourced Data," *Computers Materials & Continua*, vol. 55, no. 1, pp. 37–57, 2018.

[10] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222–233, 2014.

[11] W. Sun, B. Wang, N. Cao et al., "Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security, ASIA CCS 2013*, pp. 71–81, May 2013.

[12] R. Li, Z. Xu, W. Kang, K. C. Yow, and C.-Z. Xu, "Efficient multi-keyword ranked query over encrypted data in cloud computing," *Future Generation Computer Systems*, vol. 30, no. 1, pp. 179–190, 2014.

[13] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 9, pp. 2546–2559, 2016.

[14] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *Proceedings of the IEEE INFOCOM, 2010*, pp. 1–5, IEEE, San Diego, CA, USA, 2010.

[15] Z. Fu, J. Shu, X. Sun, and N. Linge, "Smart cloud search services: Verifiable keyword-based semantic search over encrypted cloud data," *IEEE Transactions on Consumer Electronics*, vol. 60, no. 4, pp. 762–770, 2014.

[16] Z. Fu, X. Wu, Q. Wang, and K. Ren, "Enabling central keyword based semantic extension search over encrypted outsourced data," *IEEE Transactions on Information Forensics & Security*, vol. 12, no. 12, 2017.

[17] B. Wang, S. Yu, W. Lou, and Y. T. Hou, "Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud," in *Proceedings of the 33rd IEEE Conference on Computer Communications, IEEE INFOCOM 2014*, pp. 2112–2120, May 2014.

[18] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang, "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE Transactions on Parallel & Distributed Systems*, vol. 27, no. 9, p. 1, 2016.

[19] Z. Fu, F. Huang, X. Sun, A. V. Vasilakos, and C. Yang, "Enabling semantic search based on conceptual graphs over encrypted outsourced data," *IEEE Transactions on Services Computing*, 2016.

[20] J. F. Sowa, "Conceptual structures: information processing in mind and machine," 1983.

[21] G. W. Mineau, B. Moulin, and J. F. Sowa, *Conceptual Graphs for Knowledge Representation*, Springer Berlin Heidelberg, 1993.

[22] R. Ihaka and R. Gentleman, "R: a language for data analysis and graphics," *Journal of Computational and Graphical Statistics*, vol. 5, no. 3, pp. 299–314, 1996.

[23] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in Neural Information Processing Systems*, vol. 26, pp. 3111–3119, 2013.

[24] S. Miranda-Jiménez, A. Gelbukh, and G. Sidorov, "Summarizing Conceptual Graphs for Automatic Summarization Task," in *Proceedings of the International Conference on Conceptual Structures*, vol. 7735, pp. 245–253, 2013.

[25] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in *Proceedings of the 30th IEEE International Conference on Distributed Computing Systems, ICDCS 2010*, pp. 253–262, June 2010.

[26] Y. L. Liu, N. Hu, H. Xu, H. Ning, and L. K. Wu, "A real-time monitoring technique for local plasticity in metals based on Lamb waves and a directional actuator/sensor set," *Computers, Materials & Continua*, vol. 40, no. 1, pp. 1–20, 2014.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 6459326, 14 pages
https://doi.org/10.1155/2018/6459326

Research Article

Flow Correlation Degree Optimization Driven Random Forest for Detecting DDoS Attacks in Cloud Computing

Jieren Cheng,[1,2,3] **Mengyang Li**,[1,2] **Xiangyan Tang**,[1,2] **Victor S. Sheng**,[4] **Yifu Liu**,[1,2] **and Wei Guo**[1,2]

[1] *Key Laboratory of Internet Information Retrieval of Hainan Province, Hainan University, Haikou 570228, China*
[2] *College of Information Science and Technology, Hainan University, Haikou 570228, China*
[3] *State Key Laboratory of Marine Resource Utilization in South China Sea, Haikou 570228, China*
[4] *Department of Computer Science, University of Central Arkansas, Conway, AR 72035, USA*

Correspondence should be addressed to Mengyang Li; 1098743772@qq.com

Received 23 August 2018; Accepted 1 November 2018; Published 19 November 2018

Guest Editor: Lianyong Qi

Distributed denial-of-service (DDoS) has caused major damage to cloud computing, and the false- and missing-alarm rates of existing DDoS attack-detection methods are relatively high in cloud environment. In this paper, we propose a DDoS attack-detection method with enhanced random forest (RF) optimized by genetic algorithm based on flow correlation degree (FCD) feature. We define the FCD feature according to the asymmetric and semidirectivity interaction characteristics and use the two-tuples FCD feature consisting of packet-statistical degree (PSD) and semidirectivity interaction abnormality (SDIA) to describe the features of attack flow and normal flow. Then we use a genetic algorithm based on the FCD feature sequences to optimize two key parameters of the decision tree in the RF: the maximum number of decision trees and the maximum depth of every single decision tree. We apply the trained RF model with optimized parameters to generate the classifier to be used for DDoS attack-detection. The experiment shows that the proposed method can effectively detect DDoS attacks in cloud environment with a higher accuracy rate and lower false- and missing-alarm rates compared to existing DDoS attack-detection methods.

1. Introduction

Cloud computing is a powerful technology to perform massive-scale and complex computing in which a huge amount of storage, data, and services is available over the Internet. Cloud services are distributed in nature so they can be sharable by millions of users, so that the cloud environment has to face numerous security challenges; in particular, distributed denial-of-service (DDoS) is one of the most prominent security attack in cloud computing. In recent years, DDoS attacks are on rise in frequency and severity in cloud computing and have become a growing problem because automated tools have been continuously improved and botnets of computers can be easily rented and organized to launch attacks by less sophisticated attackers [1, 2].

A DDoS trend and analysis report [3] shows that the average global enterprise encounters 237 DDoS attacks each month, which is equivalent to eight attacks per day. The main purpose of attackers is to force enterprise system servers unavailable or steal sensitive data. At the same time, the average number of DDoS incidents that global companies have experienced every month (Q3 2017) has increased by 35%. The scale and harm of DDoS attacks are increasing by leaps and bounds. Various forms of flooding and vulnerability attacks still affect and destroy networks and services. What is more, the Internet of things (IoT), industry 4.0, smart cities, and novel artificial-intelligence (AI) applications that require devices to be connected to cloud platforms provide an increasing wide range of potential botnet zombies, and the issue of controlling these botnets to launch DDoS attacks has become increasingly severe and important in cloud computing environment. Research in this area is important and significant.

Through the above analysis, we can understand the necessity of a DDoS attack-detection method. This paper seeks a better feature for attack-detection and a relatively

accurate and stable random forest (RF) attack-detection model by experiments and analysis. The organization of this paper is as follows. Section 2 introduces related work. In Section 3, we analyze attack characteristics and flow correlation degree (FCD) features. Section 4 introduces a random forest detection model based on genetic algorithm optimization. Section 5 introduces our experiments and their results. We provide our conclusions and ideas for future work in Section 6.

2. Related Research

Much research has been dedicated to DDoS attack-detection technology. Soft computing or artificial-intelligence methods are widely used in attack-detection [4]. Depending on the analysis method, DDoS detection methods can be classified into the three types of misuse, anomaly-based, and hybrid detection.

(1) Feature-based detection is also known as misuse, pattern, knowledge-based, and rule-based detection. This approach captures the required behavior from available datasets (such as protocol provisions and network-traffic events) and collects information about various attacks and system risks. This type of method uses the signature or mode of an attack, and such information as the index of the source IP address, destination IP address, and key of the port and packet payload in the IP packet. It matches incoming traffic to a stored pattern to identify an attack instance. IDES and INBOUNDS [5] are both signature-based detection methods. In recent years, new research has been conducted. Zhou et al. [6] proposed a DDoS attack-detection method which distinguished the constant attacks and the pulsing attacks from normal traffic by using the expectation of packet size. However, this method relies excessively on packet size and cannot adapt to multiple attack scenarios. Dodig et al. [7] proposed a new data structure based on a novel Dual Counting Bloom Filter to reduce detection errors for matching packages and theoretically analyzed the detection probability of determining the error rate and the requirement of increasing memory.

(2) Detection methods based on anomalies (also known as outliers and performance-based) can detect new types of attacks and unknown or emerging (undefined) attacks. When the difference between observed and expected behavior exceeds a predefined threshold, the detection system will generate an alarm. This method uses statistical methods, data mining, artificial intelligence, information theory, K-nearest neighbor, and other methods to identify anomalies in network traffic. Bhuyan et al. [8] proposed a scheme for DDoS flooding attack-detection and IP traceback by measuring the metric difference between the lightweight extended entropy of normal flow and attack flow. Latif et al. [9] proposed an enhanced decision tree algorithm based on a lightweight iterative pruning technique to detect DDoS attack and evaluated the performance of the proposed algorithm from classification accuracy, time, and space complexity, but the algorithm displays some defects in robustness due to flaws in decision tree classifier.

(3) Hybrid-based DDoS attack-detection combines two or more of the above strategies. A hybrid model can analyze common system behavior and inappropriate attacker behavior to improve the monitoring capabilities of the detection system. If hybrid system has both detection technology based on anomalies and features, the hybrid system can handle familiar and anonymous attacks and has characteristics of two detection methods, such as a high detection rate and low false-alarm rate [10]. Feature-based systems use anomaly-based techniques to detect attackers who try to change the attack patterns in the stored signature database. In recent years, some researchers have conducted extensive research on hybrid detection techniques. Gu et al. [11] presented a semisupervised clustering detection method using multiple features to solve the problems of large amount of unlabeled data in supervised learning, low detection accuracy and slow convergence speed of unsupervised learning. Liu et al. [12] proposed a DDoS attack-detection method based on conditional random fields, in which two sets of traffic feature conditional entropy (TFCE) and behavior profile deviate degree (BPDD) were depicted the characteristics of DDoS attacks. However, the training convergence speed of this method is slow. Bojović et al. [13] proposed a DDoS attack-detection method based on an exponential moving average algorithm. However, this method cannot detect attacks well when the packet forwarding rate of attack traffic is small.

Recent DDoS attack-detection methods have tended to be hybrid methods using a combination of multimode and multipart detection in the expectation of better performance. At the same time, the advent of the cloud computing era has seen increased security analysis and strategic research in these related realms. For example, research on providing reliable, stable, efficient, and secure services as well as data to the users of cloud computing [14–21], research on security strategies and privacy protection on the IoT [22–29], research on efficient cryptography to improve system security [30–32], and research on data processing, feature extraction, and information protection by machine learning method [33, 34] are all continuously deepened. There is also more research related to machine learning and integrated learning, combining attack features or optimization algorithms with time-series, ensemble learning, and deep-learning methods for network security analysis and traffic analysis. Intrusion-detection and attack-detection can improve detection results and speed. Cheng et al. [35] proposed a prediction approach based on abnormal network flow feature sequence to solve the problems of long response time and large computing resources of a DDoS attack detector in the big-data environment. However, this method requires relatively high stability for time-series data. Jia et al. [36] proposed a hybrid heterogeneous multiclassifier ensemble learning method to detect DDoS attacks, and constructed a heuristic detection system based on singular value decomposition, but the computational efficiency of this system may be low.

In general, the false- and missing-alarm rates of existing DDoS attack-detection methods are still relatively high in a cloud computing environment. In response to the problem, this paper analyzes network traffic, proposes a flow correlation degree feature, applies a random forest detection model,

optimizes its parameters to accurately and effectively detect DDoS attacks, and conducts research on attack characteristics and algorithm detection-performance optimization.

3. DDoS Attack Feature Extraction

3.1. DDoS Attack Feature Analysis. DDoS attack features have an important impact on attack-detection results. A feature that can effectively and steadily reflect DDoS attacks has a significant improvement in detection. Generally, DDoS attack features are extracted by describing the current network state through certain parameters or by observing changes in network parameter values, such as IP addresses, ports, payloads, and sizes of IP packets. The following two points are drawn from the consideration of cloud computing environments, as well as a great deal of research on feature extraction of DDoS attacks [37–39].

(1) The net source address and destination address, source address and destination port, and destination port and destination address all have a "many to one" relationship resulting in attacks that present the characteristics of flow asymmetry. Currently, many flooding attacks rely on botnets to attack target hosts or networks, forming a many to one attack mode to expand the scope of attacks and increase the harm of attacks, which can restrict or even paralyze them. At the same time, attacks can be more targeted, resulting in a certain service in the target network that cannot be used normally. Furthermore, system resources are attacked on multiple ports, so that multiple services cannot be used normally. Attacks can present a large amount of flow asymmetry.

(2) The network flows in direct or reflected DDoS attacks have higher semidirectivity interaction. In addition to flooding attacks, for an open shared-resource platform that lacks source IP address authentication or authentication capability of the packet source, the attacker uses packet source IP spoofing to attack. Using existing tools, numerous fake IP data packets are sent to the target network or host, causing abnormal or degraded network service. Most of the normal traffic at the monitoring point will respond to the destination and destination-to-source addresses. A large number of attacks will seriously affect the interaction. Therefore, the source IP address cannot receive a valid reply from the destination IP address. That is, the attack will greatly increase semidirectivity interaction of the network. Therefore, based on flow asymmetry and semidirectivity interaction characteristics, we propose the following feature extraction process.

3.2. Feature Extraction Rules. Assume that, within a unit time T, the net flow F is $< (t_1, s_1, d_1, dp_1), \ldots, (t_i, s_i, d_i, dp_i), \ldots, (t_n, s_n, d_n, dp_n) >$. Among them, $i = 1, 2, \ldots, n$, t_i, s_i, d_i, dp_i, represent the time of the i-th packet, source IP address, destination IP address, and destination port number. To classify these n packets, we use the following rules:

(1) Packets with the same source and destination IP addresses are grouped in the same category. All data with the source IP address A_m and the destination IP address A_n are marked. The packet formation class is $SDIP(A_m, A_n)$. For the above formed classes, execute the following deletion rule.

If there are different destination IP addresses A_n and A_k, ensure that the classes $SDIP\ (A_m,\ A_n)$ and $SDIP\ (A_m, A_k)$ are not empty, and delete all the classes whose source IP address is A_m.

Assume that the last remaining classes are $RSD_1, \ldots, RSD_m$, which define the packet-statistical degree (PSD) of the network flow F as

$$PSD_F = \sum_{i=1}^{m} W\left(RSD_i\right). \tag{1}$$

where $W(RSD_i) = \alpha Port(RSD_i) + (1-\alpha)Packet(RSD_i)$, $(0 < \theta < 1)$, $Port(RSD_i)$ is the number of different port numbers of class RSD_i, $Packet(RSD_i)$ is the number of packets in the class of RSD_i, and α is the weighted value. In general, $\alpha = 0.5$.

(2) Classifying the n packets, separate data packets from the same source and destination IP addresses in the same class. $SIPC\ (A_m)$ represents the class of data packets with source IP address A_m. $DIPC\ (A_n)$ represents the class of data packets with destination IP address A_n.

If the source IP address A_m of class $SIPC\ (A_m)$ causes $DIPC\ (A_m)$ to be NULL, we define all of the data packets as source semidirectivity interaction flow and mark them as $SOH(A_m)$. This respects the property of source semidirectivity interaction, and we mark the different port numbers as $Port(SOH(A_m))$.

According to the above definition of source semidirectivity interaction, we obtain all the source semidirectivity interaction flow SOHs, expressed as $SOH_1, \ldots, SOH_s$.

Classifying the flow of SOH, we place the SOHs with the same destination IP in the same class marked as $SDH\ (Mton_m, A_m), m = 1, 2, \ldots, l$, l represents the amount of the destination IP address in SOH flow. The number of SOH flows with different source IP addresses and the same destination IP address is marked as $Mton_m$.

Suppose $Mton_m \geq M(M \geq 2$, where a greater value of M signifies a better effect of removing normal flow interference. To improve the coverage of attack-detection, we define $M = 2$. If we have SDH class asSDH_1, SDH_2,..., SDH_k, the number of destination port numbers in a class is expressed as $Port(SDH_i), i = 1, 2, \ldots, k$.

Semidirectivity interaction abnormality (SDIA) of the network flow F is defined as

$$SDIA_F = \frac{1}{f(k)}\left(\sum_{i=1}^{k}\left(Mton_i + weight\left(Port\left(SDH_i\right)\right)\right) - k\right). \tag{2}$$

Here, $f(x) = \{x, x > 1; 1, x \leq 1\}$, $weight(x) = \{x, x/\Delta t > \theta_1; 0, x/\Delta t \leq \theta_1\}$, Δt is the sampling-time period, θ_1 is weighted thresholds for the number of different destination ports, and $\theta_1 = \max(Port(SDH_i))\ /\Delta t, i = 1, 2, \ldots, k$. One can also specify a threshold based on experience.

(3)Combined with the feature extraction rule of (1) and (2), in a unit time T, two features of PSD and SDIA are

calculated and extracted, respectively, and a two-tuple feature is structured from these two features of PSD and SDIA to generate the network flow correlation degree (FCD) feature of the network flow F; we compute

$$FCD_F = \left(PSD_F, SDIA_F\right). \tag{3}$$

Normal network flow and DDoS attack flow in large data environment have the characteristics of high capacity, diversity, and burst, but FCD feature can still reflect the essential difference between normal flow and attack flow. First, the two parts in FCD feature are both extracted based on the asymmetry of DDoS attacks, and the FCD eigenvalues in attack cases are significantly larger than those in normal cases and last longer. Second, PSD features extraction is the weighted statistical features of the source IP address and port of the network flows of the "many to one" and "one to one" session mode, which eliminates the interference the network flows of "one-to-multi" session mode and reflects the correlation between attack flow and normal flow in the network more clearly. However, what the SDIA feature extracts is the weighted statistical information of the one-way flows of the "many to one" session mode in the network flow, which can more accurately describe the dramatic increase of the one-way flow when the network is attacked by DDoS attack. The combination of these two pieces of statistical information can accurately describe the phenomenon that attack flows converge at the injured end and directly affect the change of normal traffic and that a partially converged attack flow is mixed with a large amount of normal flow. This feature can present the higher source address distribution, destination address concentration, source destination IP address asymmetry, and high-traffic bursts for DDoS attacks in cloud computing environment, which provides more accurate, timely, and complete information about the network before and after the attack.

4. Implementation of DDoS Attack-Detection Method Based on Random Forest and FCD

4.1. FCD Feature Sequence Extraction. According to the rule described above in Section 3.2, the data of net flow are sampled by time interval, and the values of PSD and SDIA in each sampling-time are calculated and integrated into a two-element combination. After N samples, FCD time-series sample M is obtained, $M(N, \Delta t) = \{FCD_i, i = 1, 2, \ldots, N\}$, where N is the sequence length. With the accumulation of sampling-time Δt, the sequence is a time-characteristic sequence with a time length of N. Based on the FCD feature sequence extracted above, we can construct a RF classifier to detect DDoS attacks.

4.2. Random Forest. Random forest is a classification method of integrated learning. In the training process, it can use a resampling technique (bootstrap method) in which each sample returned from the original training data is randomly selected from the same number of samples, consisting of a new training dataset, and multiple decision trees are independently generated. In each decision tree, according to some evaluation criteria like the information entropy and Gini coefficient, the selection of the best test from the new training dataset is used as the decision point to carry on the split test, and then the result of the single decision tree is produced; the final decision is formed by calculating the mode of classification results of all decision trees. A formal description is given below.

Suppose the whole RF classifier is $R(x)$; decision tree i is denoted as $t(x)$, $R(x) = \{t_i(x), i \in [0, n\_estimators]\}$, where $n\_estimators$ represents the number of decision trees in the RF, x is the input training sample to be classified, and $\text{sign}(x) \in S$ is the tag value of x, in which S is the set of labeled categories, the output of the $t_i(x)$ is a certain value in S, and the output of the $R(x)$ is the mode of the estimated value of $\{t_i(x), i \in [0, n\_estimators]\}$. In the use of RF for testing, x is the value of the new training dataset randomly generated by resampling technology in the FCD feature training set; there are only two kinds of labels in DDoS attack-detection, which represent abnormal and normal. Therefore, $S = \{-1, 1\}$, and $\text{sign}(x)$ can only take the value -1 or 1 to represent the attack sample labels and normal sample labels, respectively.

$$Gini(D) = 1 - \sum_{i=1}^{k} p_i^2. \tag{4}$$

In this paper, the Gini coefficient is selected as the quantitative evaluation criterion of the single-decision tree division, as specified in formula (4). In this equation, D represents the sample space of n samples and k categories, and p_i represents the proportion of the i samples of the entire sample. When used in a specific experiment, D is a sample space constructed for the set of feature datasets for training, where $k = 2$, and n is the size of the training sample. The Gini coefficient represents the impurity of the training model. The smaller the value, the lower the purity and the better the characteristics. In addition, the main reason for the use of Gini coefficients as splitting indices of decision trees in the RF is that the coefficient cannot only reflect the proportion of all categories of samples and different types of sample proportion changes but can also make their values meet between $(0, 1)$, to facilitate the processing analysis.

According to the information above, the RF-detection model is constructed based on the FCD feature sequence. In the construction process, a genetic algorithm is selected to optimize and determine the number of decision trees and the maximum depth of the single decision tree in the RF. The process is introduced in Section 4.3.

4.3. Genetic Algorithm Optimization of Random Forest. The genetic algorithm is based on Darwin's biological theory of evolution. We search for the optimal solution by simulating the process of natural evolution in a certain range of solution sets. The solution set most in accordance with the "survival of the fittest" principle as in generational evolution is the approximate optimal solution. As a global optimization probability algorithm, a genetic algorithm can guarantee effectiveness in a large dataset using a heuristic method, and it can search the optimal solutions of all problems in any sense of form and function in a global sense. Therefore, the

range of key parameters in RF can be determined based on empirical values and mathematical derivation. In a relatively simple way, a genetic algorithm is used to select more reliable detection parameters.

In the process of constructing RFs, the parameters in a forest, such as the number of producing subdecision trees, the number of random attributes, and the maximum depth of trees, will affect the final classification results. Whether the number of subdecision trees selected is appropriate for the training results of a RF has a critical impact. Too small a number will lead to inadequate training, which cannot produce good results, while too large a number will lead to a long construction time and overly complex RF. A single-decision-tree depth also has a great impact on the training results and training time. The appropriate depth can guarantee the subtree of the leaf node to have a more reasonable classification, and it also reduces the training time. Therefore, we choose two key parameters, the number of estimators ($n\_estimators$) and the maximum depth (max_depth) of the subtree as the parameters to be optimized. The process is as followed.

(1) Choose the parameter-selection strategy and fitness function. Parameter selection includes the determination of the population size, the number of iterations, selection, crossover, and mutation probability. Fitness function is the basis of genetic variation of individuals and population evolution in genetic algorithm. Here, considering the impact of constructing RF and finding optimal parameters on the time of constructing classification model, the following parameter values and the ranges of parameters to be optimized are determined. Set that the initial population size is 10, the number of iterations is 30, the range of $n\_estimators$ in RF is (2, 30), the range of max_depth of the subdecision tree is (2, 8), and the mutation rate and cross rate are default. Considering the generality and reliability of the fitness function value, the average value of the area (area under curve, AUC) under the ROC curve in the cross validation of the training sample is selected as the fitness function value. The greater the value, the more conducive to the inheritance and evolution of the individual.

(2) Encode and initialize the population. The binary encoding method is used for coding. From a given set of two positive integer parameter ranges, the parameter combination ($n\_estimators, max\_depth$) is randomly selected and encoded as chromosome $X = \{n\_estimators, max\_depth\}$. The initial population G is randomly initialized by multiple individuals resulting from the crossover and mutation of the chromosome X. Binary coding of chromosomes can increase the likelihood of mutation and crossover, thus providing more diverse solutions.

(3) Evaluate the fitness value. According to the fitness function value mentioned above in (1), the fitness value of each individual population can be calculated, as shown in formula (5), in which K represents the fold number of cross validation, AUC is the area under the calculated ROC curve when the training sample is tested as a test sample in cross validation, and when this value is greater, the fitness value is better. Then the fitness values of each individual are calculated. By comparing the fitness values of individuals, those with the best fitness value are selected to generate the initial individuals of the next generation of the population, so as to carry out subsequent crossover and mutation operations.

$$Fitness = \frac{1}{K}\sum_{i=1}^{K} AUC_i. \tag{5}$$

(4) Judge terminating conditions. In the process of continuous iteration, it is judged whether the fitness meets the established standard. If it is not satisfied, then step 3 is repeatedly performed until the termination condition is reached. At this time, we select the individual with the largest fitness value in the population and extract the corresponding decimal values of binary-coded chromosome X in the individual as the optimal parameters of RF for training.

(5) Apply the optimal parameters. The optimal $n\_estimators$ and max_depth values are selected as the parameters of the RF for training, the RF classifier is trained based on the training set of FCD feature sequence and this two optimal parameters, and the DDoS attack-detection model based on genetic algorithm optimization and RF is constructed.

By optimizing the parameters above and constructing RF model, we can obtain an RF-detection model optimized by the genetic algorithm, which is more accurate than the general RF-detection model. Considering the heuristic searching ability of genetic algorithm, the combination of genetic algorithm and RF can effectively improve the classification ability of RF, so as to detect DDoS attacks more accurately and effectively.

4.4. Random Forest Detection Based on Genetic Algorithm Optimization. According to the above description in Section 4, we optimized the parameters based on the FCD feature, trained the RF classifier and obtained the genetic algorithm-optimized random forest (GAORF) based on the FCD feature sequence. In this paper, the DDoS attack-detection method with the model generated by FCD feature sequence and GAORF algorithm is referred to as FGRF attack-detection method. The process of the application of the method in this paper is shown in Figure 1.

An attack can be identified according to the model of the FGRF detection method trained to characterize the network state. The model actually solves the problem of binary classification in machine learning. The detection task can only identify an attack or not. Assuming that the detection model detects that net flow does not have feature anomalies during a certain period of time under normal conditions, we set the detection model output flag to 1. Under attack conditions, the FCD feature value will rise obviously with the time change, which is gradually higher than the normal value, and then we set the output flag of the detection model to another value, and we set it to -1 in this paper. These two settings can characterize whether the network is attacked or not. As the FGRF detection method is used to detect the real DDoS attack, after the FCD value of the net flow is entered into the model, the output flag returned by the model can reflect whether the network is attacked.

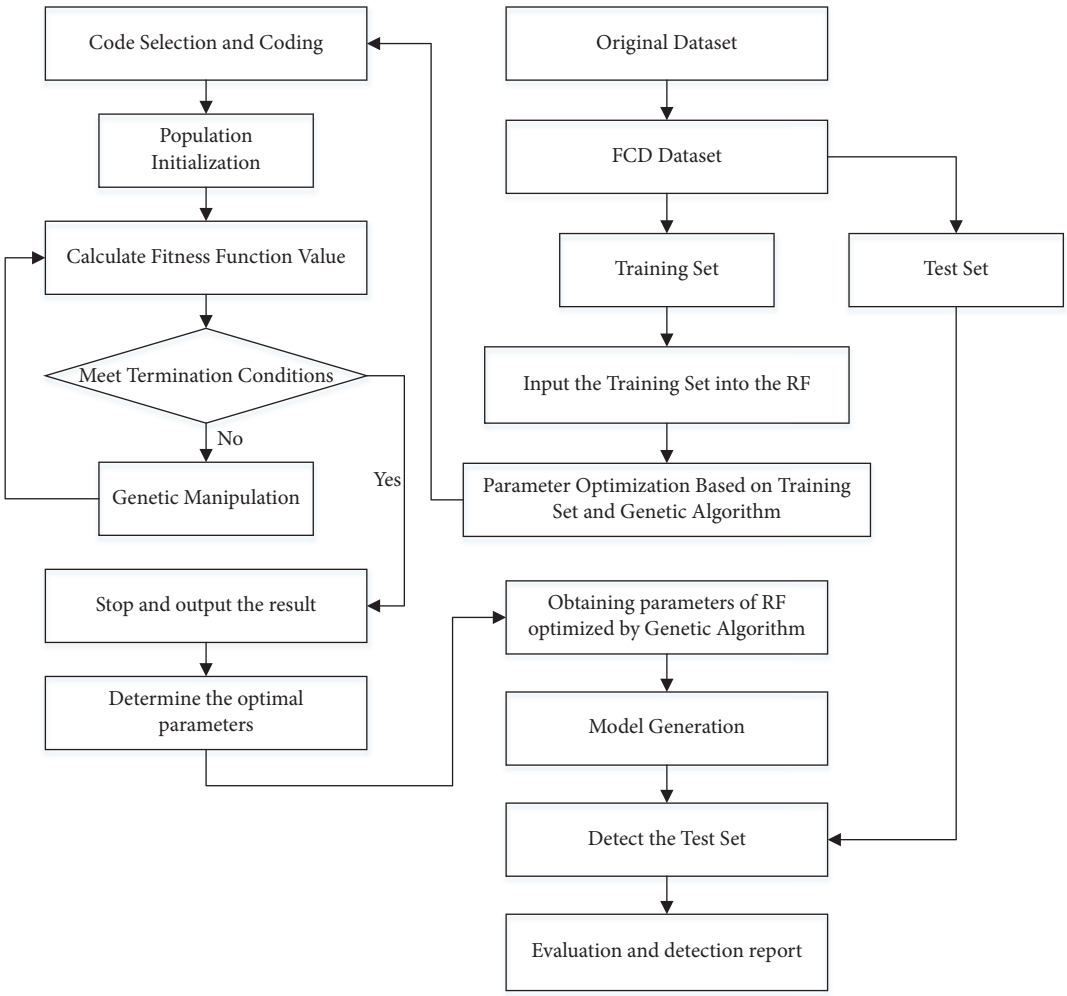


Figure 1: The process of FGRF DDoS attack-detection method.

The analysis in Section 3.2 of this paper shows that FCD feature sequence can better reflect the different state characteristics of normal flow and DDoS attack flow in cloud computing environment. Multiple decision trees are integrated in the RF, the bootstrap method is used to reduce the size of the single-decision-tree training sample set, and a more reasonable classification result is selected using the voting mechanism. The combination of these mechanisms in RF can improve the accuracy of detecting high-capacity traffic information in DDoS attacks under cloud computing environment. Moreover, the method based on genetic algorithm to optimize RF parameters effectively improves the classification ability of RF. Therefore, the FGRF attack-detection method proposed in this paper can effectively detect DDoS attacks under cloud computing environment.

5. Experiment

5.1. Data Set and Evaluation Criteria. The experimental hardware had 8G memory and an i7 processor. The experiment was carried out on a Windows 10 64-bit system running Python 3.5.2 |Anaconda 4.2.0 (64 bits).

The experiment was based on the dataset of the CAIDA DDoS attack in 2007 [40]. It contained data on an anonymous DDoS attack that lasted for about an hour on August 4, 2007. This type of attack attempts to prevent access to target servers by consuming computing resources on servers and all the bandwidth of connecting servers to Internet networks. The total size of the dataset was 21 GB, accounting for about an hour (20:50:08 UTC -21:56:16 UTC). The attack started at about 21:13 and caused the rapid growth of the network load (in a few minutes) from about 200,000 bits/sec to 80 MB/sec. The attack traffic was divided into five-minute files and stored in PCAP format.

To judge the validity of attack-detection, some evaluation criteria were used to fully illustrate the performance of the test, including the accuracy rate, missing-alarm rate (MR), and false-alarm rate (FR). Suppose TP is the number of normal samples marked correctly, TN is the number of attack samples that are correctly marked, FN is the number of attack samples marked in error, and FP is the number of normal samples marked in error.

$$accuracy = \frac{TP + TN}{TP + TN + FN + FP}. \quad (6)$$

$$FR = \frac{FP}{TP + FP}. \quad (7)$$

$$MR = \frac{FN}{TN + FN}. \quad (8)$$

The accuracy rate is the proportion of the correctly identified samples in all samples; the false-alarm rate is the proportion of samples judged to be attacked in the normal sample, and the missing-alarm rate is the proportion of the sample that is not successfully identified. Then TN/TN + FN is the detection rate. Through the environment, datasets, and evaluation criteria described above, experiments are carried out according to the process described in Section 4. FCD feature sequences are extracted from the data sets described in Section 5.1, and all normal samples are labeled as 1 and all attack samples are labeled as -1 according to Section 4.4. Training and test sets are selected from the FCD feature sequences. The parameters of RF are optimized by genetic algorithm based on training set, and the model of the FGRF attack-detection method is established, and the performance of classification model is verified by test set. SVM algorithm is more classic and has better classification results because of its use of the mechanism of hyperplane classification. In order to better illustrate the good performance of the FGRF attack-detection method proposed in this paper, the model is compared with several detection models generated by a variety of SVMs which is trained based on the FCD feature sequences. The scikit-learn [41] toolkit was used to complete the implementation of RF and GAORF. The LIBSVM [42] toolkit was used to complete the contrast test in SVMs. The experimental process and its results are introduced in Section 5.2.

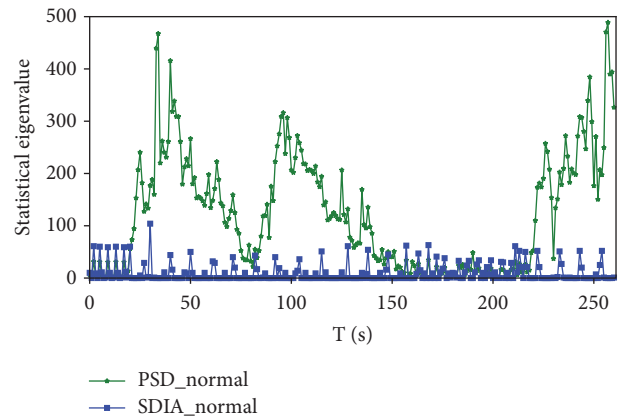


Figure 2: Comparison of PSD and SDIA features in normal flow.

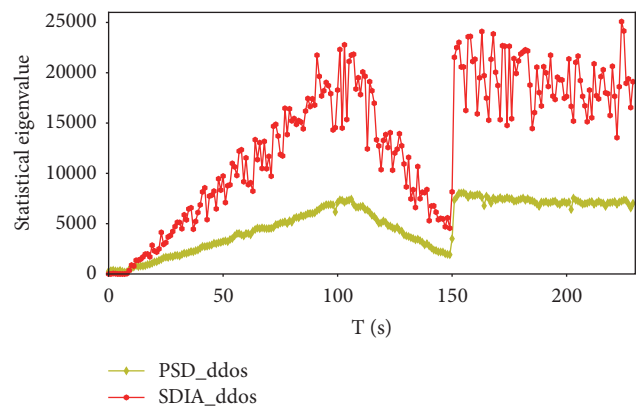


Figure 3: Comparison of PSD and SDIA features in attack flow.

5.2. Experimental Data Analysis. We obtained a normal data sample from ddostrace.20070804_134936.pcap and an attack data sample from ddostrace.20070804_141436.pcap in the DDoS Attack 2007 dataset. According to the feature extraction rules in Section 3.2 and the feature sequence extraction method in Section 4.1, FCD feature sequence was extracted from normal and attack samples. For convenience of calculation and processing, we set $\Delta t = 1$s as the sampling interval. The parameters of PSD and SDIA in FCD feature were set according to Section 3 and the FCD value time-series sample M as shown in Figures 2 and 3.

As shown in Figure 2, in the normal flow, the sequence of the PSD eigenvalues shows a stronger volatility, and the highest feature value can reach about 500, while the sequence of the SDIA eigenvalues is relatively stable and their values are floating within the range of 150. The PSD feature statistics are the characteristic information of the network flows of "one to one" and "many to one" session mode, and because of network congestion, similar network flows are more common in normal flow, so the values of PSD features will fluctuate in a certain range, which can better reflect the abnormal changes of normal flow state caused by attack flow than SDIA features. The SDIA feature statistics are the characteristic information of the one-way flows of the "multi to one" session mode. In the normal network, the one-way flows are relatively less than the bidirectional flows, so the sequence of SDIA eigenvalues is more stable.

As shown in Figure 3, both PSD eigenvalues and SDIA eigenvalues increase with the increase of DDoS attack flow, but SDIA eigenvalues are relatively higher than PSD eigenvalues at the same time, The SDIA eigenvalue reaches a peak value of about 25000, while the PSD eigenvalue reaches a peak value of about 8000. Obviously, the change of the SDIA feature between them is more obvious. The one-way flows of the "many to one" session mode in the network will increase rapidly caused by DDoS attacks; both PSD and SDIA feature have weighted the information of the one-way flows of the "many to one" session mode, so their values will increase and can reflect the attack state to a certain extent. In addition, the two eigenvalues in Figure 3 show a sudden decrease and then continue to increase, which is caused by the decrease of the one-way flows of the "many to one" session mode in the network caused by such factors as the delay of attack at that time. The SDIA feature, which is different from PSD feature in the weighted calculation method, more centrally describes the related information about the one-way flows of the "many to one" session mode, so it can describe attack flow more accurately than PSD feature. It can well reflect the semidirectivity interaction of large-area network flow caused by DDoS attacks.

The combination of PSD feature and SDIA feature is the FCD feature proposed in this paper. This feature can integrate the advantages of the two features, not only can describe the attack flow well, but also can reflect the abnormal changes of the normal flow state caused by the attack flow, so it can better identify the attack.

In the process of experiment, training and testing samples were selected first. To facilitate integration, calculation, and processing, 200 FCD features were selected as test datasets, which include 100 normal flow features and 100 attack flow features, respectively. In the rest of the features of FCD, 100 normal flow features and 120 attack flow features are selected as training samples. To study the negative sample, which is the attack sample, we made an appropriate increase under the restriction of the existing characteristic dataset, so as to obtain better training.

After selecting the training set and test set from the whole feature set, the data samples are normalized, and the genetic algorithm is used to optimize the RF model trained by the training set. Due to the small number of samples, it is still necessary to ensure reasonable and effective testing. Therefore, $K = 2$ in formula (5) are used when evaluating the fitness value in Section 4.3.

In the experiment of optimizing the parameters, the number of training samples is small, and the initial population size is large. Considering the good classification performance of the RF algorithm itself, a good parameter-solution set will be found quickly within the specified number of iterations. In addition, the properties of the genetic algorithm in random search of the optimal parameter-solution set in the prescribed range also increase the possibility of producing better results. Therefore, the combination of genetic algorithm and RF algorithm can find the approximate optimal solution of these parameters to a large extent in the global scope.

In the end, after the 30 iterations we set in Section 4.3, a relatively high-quality parameter-solution set was determined based on the training sample set, that is, the value of the two optimal parameters of the number of subtrees and the maximum depth of subtree. These two parameter values were brought into the RF model for training, and a classification model was generated for detection. Finally, the results described in Section 5.1 were used to judge the test results. To make the results more effective and reliable, we conducted comparative experiments. The experimental results are introduced in Section 5.3.

5.3. Experiments and Results. To verify the detection capabilities of our proposed FCD feature combined with the detection model constructed by the RF algorithm and the genetic algorithm, we performed comparison experiments, and the specific steps and the results of the comparison experiment are as follows:

(1) In accordance with the description in Section 5.2, the training data set and test data set are selected. Here, the test data set is kept unchanged, and the following two change operations were performed on the normal sample and the attack sample in the training set to perform comparison experiments: the number of fixed attack training samples was 120, and the number of normal training samples was increased to 100 on the basis of 10 normal training samples, in order to simulate the change of normal flow in network caused by the delay of DDoS attack and other factors; the number of fixed normal training samples was 100, and the number of attack training samples was increased to 120 on the basis of 10 attack training samples to simulate the situation that the normal network is gradually starting to be attacked by DDoS attacks, resulting in a gradual increase in the attack flow. The different training samples were applied to train each model to detect the same test set, and the final test results were obtained.

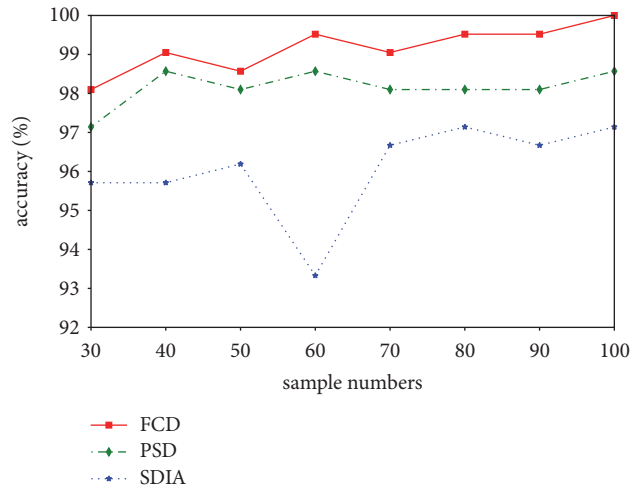


Figure 4: Accuracy comparison results of three statistical features with changing numbers of normal training samples (%).

(2) In order to further verify the good performance of the FCD feature proposed in this paper for DDoS detection, the feature FCD was compared and analyzed with the PSD and SDIA features during the experimental operation (1) in Section 5.3. The PSD, SDIA, and FCD features are extracted from the same training samples, and three classifiers are generated based on three features training RF model respectively, and then the same test set is used to test the three classifiers in order to compare the ability of the three features to distinguish between normal flow and attack flow. With the number of fixed attack training samples, Figure 4 shows the accuracy rates obtained by changing the number of normal training samples, and Figure 5 shows the false- and missing-alarm rates obtained by changing the number of normal training samples. With the number of fixed normal training samples, Figure 6 shows the accuracy rates obtained by changing the number of attack training samples, and Figure 7 shows the false- and missing-alarm rates obtained by changing the number of attack training samples. Among them, FCD_MR, PSD_MR, and SDIA_MR are missing-alarm rates based on FCD feature, PSD feature, and SDIA feature, respectively. FCD_FR, PSD_FR, and SDIA_FR are false-alarm rates based on FCD feature, PSD feature, and SDIA feature, respectively.

As shown in Figures 4 and 5, all three features can better identify attack, among which the FCD feature is the best. Seen from the aspect of accuracy, with the increase of

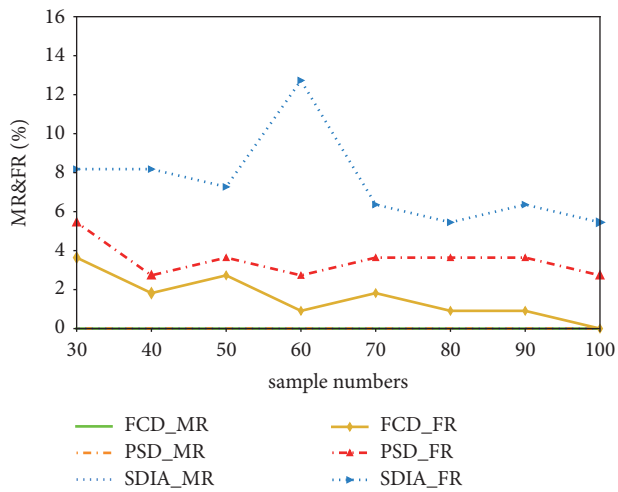


Figure 5: False-alarm rate and missing-alarm rate comparison results of three statistical features with changing numbers of normal training samples (%).

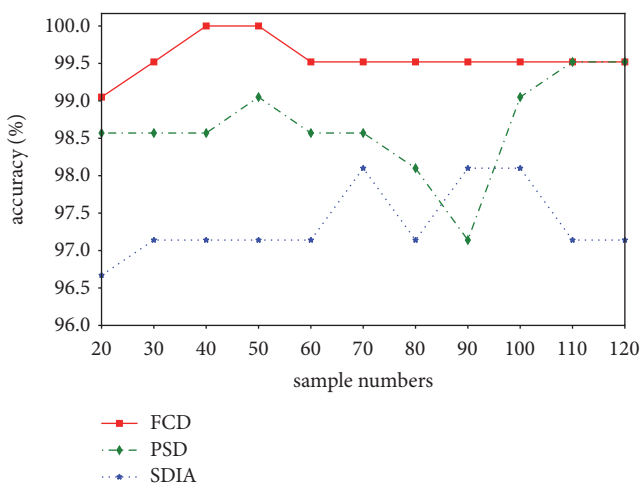


Figure 6: Accuracy comparison results of three statistical features with changing numbers of attack training samples (%).

normal samples, the accuracy rate based on FCD feature is the highest, which keeps above 98%, and increases to nearly 100%. The accuracy of PSD feature is also increased, but it is about 1% lower than that of FCD feature. As for the SDIA feature, the accuracy is kept below 97%. From the aspect of false- and missing-alarm rates, as the number of normal flow increases, FCD_MR, PSD_MR, and SDIA_MR are all zero. FCD_FR also tends to zero, although PSD_FR is down steadily to about 3%, and SDIA_FR decreases to 2% in fluctuation; they were still higher relative to the combined feature. Among them, when the number of normal samples is 60, the accuracy rate of SDIA feature decreased to about 93% and SDIA_FR increased to 13%. The accuracy rate of PSD features is about 5% higher than that of SDIA features, while the accuracy rate of FCD remains above 99%, and PSD_FR is about 10% lower than that of SDIA_FR, and FCD_FR is less than 2%. The PSD feature is the statistics of the network flows of the "many to one" and "one to one" session mode, including normal flow, its value will change with the increase of normal flow, that is, the PSD feature can better reflect the abnormal changes of normal flow state caused by attack flow, so PSD features maintain higher accuracy rate and lower false-alarm rate than the SDIA feature. The SDIA feature is the statistics of the one-way flows of the "many to one" session mode. It can describe attack characteristics more centrally, but cannot describe subtle changes of normal flow state better. Therefore, when the number of normal training samples is 60, the false-alarm rate of the detection suddenly increases, thus reducing the accuracy. FCD features contain two statistical information provided by PSD and SDIA features, so the accuracy rate of FCD features is higher, and the missing- and false-alarm rate are lower. Compared with FCD and SDIA features, FCD features can better identify DDoS attacks.

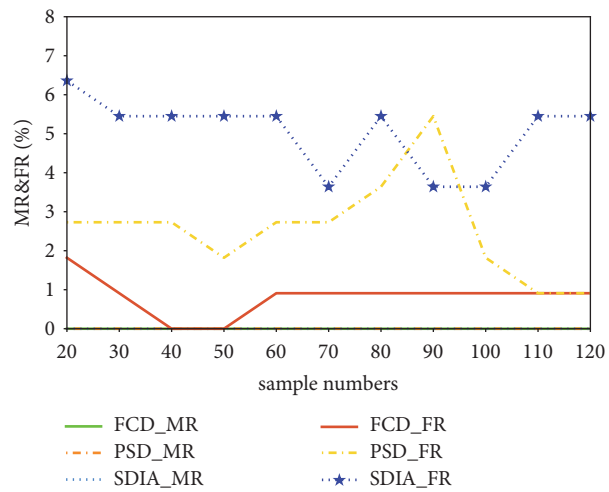


Figure 7: False-alarm rate and missing-alarm rate comparison results of three kinds of statistical feature with changing numbers of attack training samples (%).

Figures 6 and 7 show that the FCD-based RF-detection method can maintain higher accuracy rate with low false- and missing-alarm rates compared to that based on PSD and SDIA features. When the attack flow increases, the detection based on FCD features has a high accuracy of up to 99% and low false- and missing-alarm rates below 2%. PSD_FR and SDIA_FR both fluctuate over 1%, resulting in low accuracy. FCD_MR, PSD_MR, and SDIA_MR, are all zero. When the number of attack samples increases to 90, the false-alarm rate of PSD features suddenly increases to more than 5%, which is about 1% higher than that of SDIA features, and its accuracy rate decreases to less than that of SDIA features. In this case, the accuracy rate of FCD feature still maintains accuracy above 99% and false-alarm rate about 1%. Among the above data analysis results, the detection results of the three characteristics are mainly reflected in the trend of false-alarm rate. The PSD feature can well reflect the abnormal changes of normal flow state caused by attack flow, so when the proportion of normal flow in the network is still large and attack flow changes little, PSD_FR is generally lower than SDIA_FR. However, the PSD and SDIA eigenvalues are

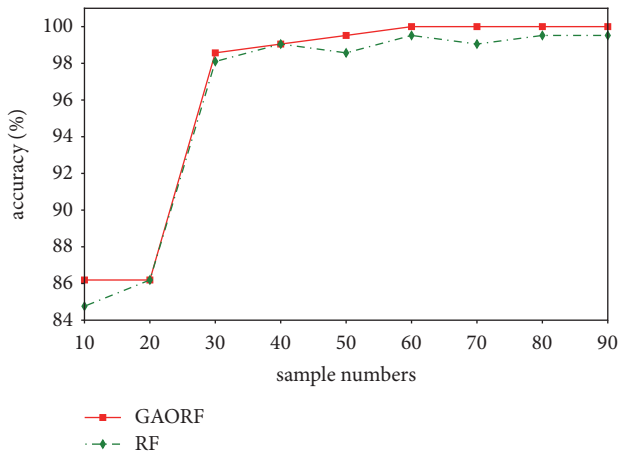


Figure 8: Comparison of accuracy rate between optimization and common model detection with changing numbers of normal training samples (%).

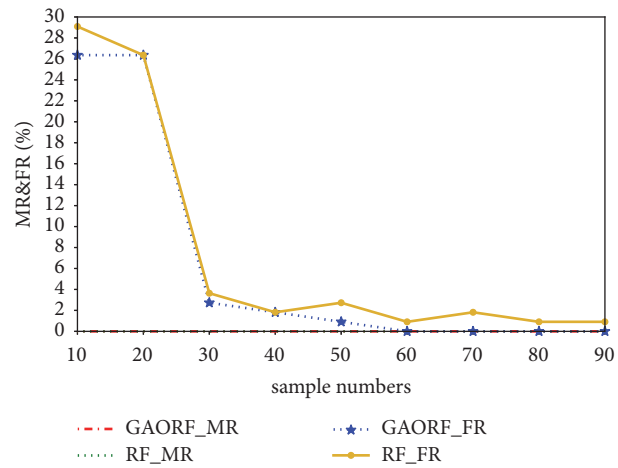


Figure 9: Comparison of false-alarm rate and missing-alarm rate between optimization and common model detection with changing numbers of normal training samples (%).

generally small in normal flow, the early attack traffic is generally small and the impact on normal flow is also small, so the PSD and SDIA eigenvalues change little in the early attack and are more likely to cause false- and missing-alarm rates. The SDIA feature is the statistics of the one-way flows of the "many to one" session mode, which can describe attack characteristics more centrally, Therefore, SDIA_FR will be significantly reduced when the early attack traffic is small or the attack is delayed, which results in a situation similar to that when the number of attack samples is 90. As for the FCD feature, it contains the information provided by the above two features, so the feature has better detection results and can better identify DDoS attacks.

(3) In order to further verify the validity of the genetic algorithm in optimizing the RF classification model, a comparison experiment was made between the RF classifier with parameter optimization by genetic algorithm and the RF classifier without parameter optimization based on the FCD feature sequences during the experimental operation (1) in Section 5.3. Two classifiers are generated based on FCD feature sequence training RF model and GAORF model respectively, and then the same test set is used to test the two classifiers in order to compare the classification ability of RF model and GAORF model.

Figure 8 shows the accuracy rates from the number of fixed attack training samples and the number of varied normal training samples. Figure 9 shows the false- and missing-alarm rate from the number of fixed attack training samples and the number of varied normal training samples. Figure 10 shows a comparison of accuracy rates from changing the number of attack training samples and fixing the number of normal training samples. Figure 11 shows a comparison of the false- and missing-alarm rates from changing the number of attack training samples and fixing the number of normal training samples. Here, GAORF_MR and RF_MR are the missing-alarm rate of GAORF detection and RF detection, respectively. GAORF_FR and RF_FR are the false-alarm rate of GAORF detection and RF detection, respectively.

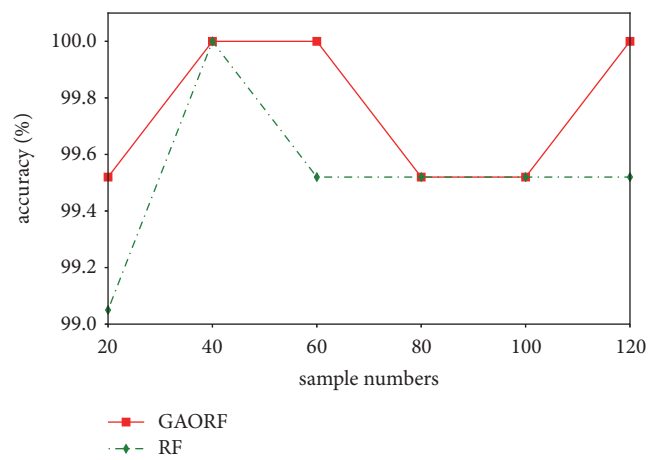


Figure 10: Comparison of accuracy rate between optimization and common model detection with changing numbers of attack training samples (%).

Combined with Figures 8 and 9, it can be seen that the accuracy rates of RF-detection model and GAORF detection model based on FCD feature sequences increases to a certain extent, and the false-alarm rates decrease gradually when the attack training sample is invariable and the normal training sample is increasing. The accuracy rate of GAORF detection model is about 2% higher and the false-alarm rate is about 2% lower. Because the heuristic parameter searching method of genetic algorithm can find better training parameters for RF classifier based on the correlation between normal flow and DDoS attack flow, which is shown by PSD features contained in FCD features, the classification performance of GAORF detection model is improved. It is worth considering that the parameter optimization process will also be constrained by the number of normal training samples, but the genetic algorithm can still find better training parameters for RF-detection model, so that can maintain the original better detection results.

Table 1: Comparison results of four algorithm detection evaluation criteria with changing numbers of normal training samples.

		Sample numbers			
		30	50	70	90
GAORF (%)	accuracy	98.57	99.52	100	100
	MR	0	0	0	0
	FR	2.72	0.91	0.0	0.0
nu-SVM (%)	accuracy	93.33	85.24	99.05	100
	MR	0	0	0	0
	FR	12.72	28.18	1.81	0
C-SVM (%)	accuracy	91.90	100	100	100
	MR	0	0	0	0
	FR	15.45	0	0	0
one-class-SVM (%)	accuracy	37.62	38.10	40.95	45.71
	MR	21.00	21.00	21.00	21.00
	FR	100	99.09	93.64	84.55

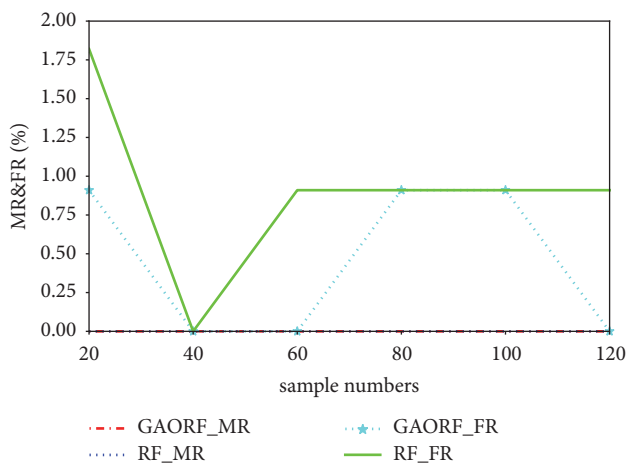


Figure 11: Comparison of false-alarm rate and missing-alarm rate between optimization and common model detection with changing numbers of attack training samples (%).

As shown in Figures 10 and 11, when the normal training samples remain unchanged and the attack training samples increase, the GAORF detection model has no missing-alarm and the false-alarm rate is about 1% lower than the RF-detection model; thus the overall accuracy rate is about 1%. Because the genetic algorithm can optimize the GAORF detection model based on the asymmetry and semidirectivity interaction characteristics of the attack flow described by the SDIA feature included in the FCD feature, the classification performance of the RF-detection model can be improved. Because the attack traffic in the early stage of DDoS attack has little influence on normal flow, the value of PSD and SDIA features in the FCD features in the early stage of DDoS attack is lower, thus affecting the detection results of the model. Genetic algorithm can still find better training parameters for the RF-detection model, so as to maintain better detection results. To sum up, using genetic algorithm to optimize the parameters of RF-detection model can effectively improve accuracy rate and reduce the false-alarm rate of DDoS attack detection.

(4) To further verify the good performance of the FGRF attack-detection method proposed in this paper, the GAORF classification model was compared with nu-SVM, C-SVM, and one-class-SVM classification models based on the FCD feature sequence during the experimental operation (1) in Section 5.3. Considering that SVM is a supervised learning algorithm with good classification performance and is widely used in previous research for DDoS attack detection, furthermore, nu-SVM, C-SVM, and one-class-SVM among SVM algorithms show stronger mode identification and classification ability; thus we chose these three SVM algorithms as the comparison algorithms. The FCD feature sequence was trained in the GAORF and three classical SVM classification methods, respectively, and then the same test set is used to test the four classifiers. We fixed the number of training samples in the attack flow and changed the number of training samples in the normal flow. The results are shown in Table 1. We fixed the number of training samples in the normal flow and changed the number of training samples in the attack flow. The results are shown in Table 2.

From Table 1, we can see that FCD combined with the GAORF detection method has higher accuracy and lower false-alarm and missing-alarm rates compared with three traditional SVM detection methods, especially when the number of normal training samples is relatively small. When the attack training samples remain unchanged, with the increase of normal training samples, the accuracy of GAORF detection model remains above 98% and the false-alarm rate remains below 3%. On the one hand, RF has a good and stable classification performance, which can be used to mine and utilize FCD features to represent the abnormal changes of normal flow state caused by attack. On the other hand, genetic algorithm optimizes RF parameters and improves RF classification ability by learning normal training sample set, so the classification effect of GAORF classification model is the best. The false-alarm rate of nu-SVM detection model fluctuates greatly, and the accuracy ranges from 85% to 100%.

Table 2: Comparison results of four algorithm detection evaluation criteria with changing numbers of attack training samples.

		Sample numbers			
		30	60	90	120
GAORF (%)	accuracy	100	100	100	100
	MR	0	0	0	0
	FR	0	0	0	0
nu-SVM (%)	accuracy	90.0	98.1	99.05	100
	MR	0	0	0	0
	FR	19.09	3.63	1.82	0
C-SVM (%)	accuracy	97.14	100	100	100
	MR	6.0	0	0	0
	FR	0	0	0	0
one-class-SVM (%)	accuracy	65.0	65.0	65.0	65.0
	MR	0	0	0	0
	FR	70.0	70.0	70.0	70.0

The training set contains some data with lower attack eigenvalues in the early stage of the attack, and these eigenvalues are similar to the normal flow eigenvalues; it is difficult to distinguish normal samples in the classification hyperplane of nu-SVM model, thus affecting the detection results. We can see that C-SVM detection model has no classification error when the number of normal training samples is more than 50, while the false-alarm rate is about 15% when the number of normal samples is 30. As the penalty coefficient of C-SVM does not change due to the excessive increase of normal training samples, the model shows good stability. However, when the number of normal samples is small, the model is difficult to obtain the optimal classification hyperplane, resulting in a sudden increase in false-alarm rate. For the one-class-SVM detection model, the detection of this model keeps the accuracy rate under 50%, the higher false-alarm rate and the false-alarm rate. The reason is that one-class-SVM can only train normal training samples to generate classification model, which makes it more difficult to recognize attacks. Therefore, it is difficult to achieve a more ideal classification effect.

As shown in Table 2, when the number of normal training samples is constant and the number of attack training samples increases, the GAORF detection method does not have a classification error, showing a better performance compared with the SVM detection methods. On the one hand, RF itself has good and stable classification performance and can better mine and utilize FCD features to characterize the characteristics of attack flow; on the other hand, genetic algorithm optimizes RF parameters by learning attack training sample set and improves the classification ability of RF so the classification effect of GAORF classification model in the four classification models is still best. The nu-SVM detection model has a better detection effect when the attack training samples increase, but its detection result is much worse than that of GAORF model when the attack training samples are few. In the early stage of attack, the attack eigenvalues are small, which can easily affect the location of the optimal hyperplane of SVM classification model, affect the recognition of normal flow, and increase the false-alarm rate. As for the C-SVM detection model, when the attack training sample is 30, the accuracy rate is 97.14% and the missing-alarm rate is 6%. With the increase of attack training samples, the classification performance becomes better. This is still the result of different fitting degree of the attack training samples, but the overall performance is still worse than GAORF. In addition, the one-class SVM detection model can only train normal training samples, thus increasing the number of attack samples, and it does not change the classification results. However, the accuracy rate of one-class-SVM attack-detection model based on FCD itself remains below 50% and missing-rate and false-alarm rate are higher, and its performance is much worse than that of the FGRF attack-detection method.

The comprehensive Tables 1 and 2 show that the GAORF classification model has stronger learning classification ability and robustness than the various classic SVM classification models for the constant change of normal samples and attack samples. Especially in the cloud computing environment, the sample feature dimension and the scale of datasets are increasing. Compared with the SVM classification model, RF can better adapt to the requirements of cloud computing. At the same time, facing the difficulty of finding the effective parameters for the detection model in cloud computing, the genetic algorithm provides a simple and effective search method, which can find the relative ideal parameters for the attack detection in a larger data range and the higher-dimension data sets. According to the characteristics of the FCD features, the characteristics of the two algorithms of GA and RF, and the experimental results, it is known that the FGRF detection method can detect attacks effectively, reduce the false- and missing-alarm rates and have good robustness. This detection method has better adaptability to DDoS attack-detection in a cloud computing environment.

6. Conclusion

In this paper, we proposed a DDoS attack-detection method based on FCD-RF, which can enhance the accuracy of

DDoS attack-detection in a cloud computing environment. We designed a feature-tuple with the statistical features of PSD and SDIA, which can describe the features of attack flow and normal flow, i.e., the FCD feature. This feature can reflect the asymmetric and semidirectivity interaction characteristics of the attack flow. The classification model was trained by the FCD feature sequence using the optimized RF based on a genetic algorithm. It could increase the accuracy rate of DDoS attack-detection and reduce the false- and missing-alarm rates. The experiment demonstrates that the detection model based on FCD and optimized RF can achieve higher accuracy and lower false- and missing-alarm rates with relatively good adaptability and robustness in a cloud computing environment.

A possible goal for our future research would be to consider multilayer mitigation and defense using profound resources in cloud computing.

Data Availability

The CAIDA UCSD "DDoS Attack 2007" Dataset used to support the findings of this study were supplied by the Information Marketplace for Policy and Analysis of Cyber-risk and Trust (IMPACT) under license and so cannot be made freely available. Requests for access to these data should be made to the Information Marketplace for Policy and Analysis of Cyber-risk and Trust (IMPACT), registration and authorization on the official website of https://www.impactcybertrust.org/. After registration, datasets can be downloaded on the CAIDA official website of http://www.caida.org/data/passive/ddos-20070804_dataset.xml.

Conflicts of Interest

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

Authors' Contributions

For the six authors in this manuscript, Jieren Cheng and Mengyang Li completed the main tasks of conducting experiments, writing and revising manuscripts. Xiangyan Tang and Victor S Sheng revised and perfected English grammar and language expression. Yifu Liu modified the format of the manuscript. Wei Guo perfected and standardized the format of the references of this manuscript.

Acknowledgments

This work was supported by the Hainan Provincial Natural Science Foundation of China [2018CXTD333, 617048]; the National Natural Science Foundation of China [61762033, 61702539]; Hainan University Doctor Start Fund Project [kyqd1328]; and Hainan University Youth Fund Project [qnjj1444].

References

[1] S. Behal and K. Kumar, "Characterization and comparison of DDoS attack tools and traffic generators - a review," *International Journal of Network Security*, vol. 19, no. 3, pp. 383–393, 2017.

[2] A. Pras, J. J. Santanna, J. Steinberger et al., "DDoS 3.0 - How Terrorists Bring Down the Internet," in *Proceedings of the 18th International GI/ITG Conference on Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*, pp. 1–4, Springer International Publishing, 2016.

[3] *CORERO DDOS TRENDS REPORT (Q2 - Q3 2017)*, Corero Network Security, 2017.

[4] N. Singh, A. Hans, K. Kumar, and M. P. Singh Birdi, "Comprehensive Study of Various Techniques for Detecting DDoS Attacks in Cloud Environment," *International Journal of Grid and Distributed Computing*, vol. 8, no. 3, pp. 119–126, 2015.

[5] T. Xia, G. Qu, S. Hariri et al., "An efficient network intrusion detection method based on information theory and genetic algorithm," in *Proceedings of the 24th IEEE International Performance, Computing, and Communications Conference*, pp. 11–17, 2005.

[6] L. Zhou, M. Liao, C. Yuan, and H. Zhang, "Low-Rate DDoS Attack Detection Using Expectation of Packet Size," *Security and Communication Networks*, vol. 2017, Article ID 3691629, 14 pages, 2017.

[7] I. Dodig, V. Sruk, and D. Cafuta, "Reducing false rate packet recognition using Dual Counting Bloom Filter," *Telecommunication Systems*, vol. 68, no. 1, pp. 67–78, 2018.

[8] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "E-LDAT: a lightweight system for DDoS flooding attack detection and IP traceback using extended entropy metric," *Security and Communication Networks*, vol. 9, no. 16, pp. 3251–3270, 2016.

[9] R. Latif, H. Abbas, S. Latif, and A. Masood, "EVFDT: an enhanced very fast decision tree algorithm for detecting distributed denial of service attack in cloud-assisted wireless body area network," *Mobile Information Systems*, vol. 2015, Article ID 260594, 13 pages, 2015.

[10] T.-M. Choi, H. K. Chan, and X. Yue, "Recent Development in Big Data Analytics for Business Operations and Risk Management," *IEEE Transactions on Cybernetics*, vol. 47, no. 1, pp. 81–92, 2017.

[11] Y. Gu, Y. Wang, Z. Yang, F. Xiong, and Y. Gao, "Multiple-features-based semisupervised clustering ddos detection method," *Mathematical Problems in Engineering*, vol. 2017, Article ID 5202836, 10 pages, 2017.

[12] Y. Liu, Z.-P. Cai, P. Zhong, J.-P. Yin, and J.-R. Cheng, "Detection approach of DDoS attacks based on conditional random fields," *Journal of Software*, vol. 22, no. 8, pp. 1897–1910, 2011.

[13] P. D. Bojović, B. Ilija, O. Stanislav et al., "A Practical Approach to Detection of DDoS Attacks Using a Hybrid Detection Method," *Computers and Electrical Engineering*, 2017.

[14] J. Zhan, X. Fan, L. Cai, Y. Gao, and J. Zhuang, "TPTVer: A trusted third party based trusted verifier for multi-layered outsourced big data system in cloud environment," *China Communications*, vol. 15, no. 2, pp. 122–137, 2018.

[15] 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.

[16] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[17] Z. Zhou, M. Dong, K. Ota, G. Wang, and L. T. Yang, "Energy-efficient resource allocation for d2d communications underlaying cloud-ran-based lte-a networks," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 428–438, 2016.

[18] W. Lin, S. Xu, L. He, and J. Li, "Multi-resource scheduling and power simulation for cloud computing," *Information Sciences*, vol. 397-398, pp. 168–186, 2017.

[19] Y. Xu, L. Qi, W. Dou, and J. Yu, "Privacy-Preserving and Scalable Service Recommendation Based on SimHash in a Distributed Cloud Environment," *Complexity*, vol. 2017, Article ID 3437854, 9 pages, 2017.

[20] 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.

[21] 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.

[22] L. Qi, X. Zhang, W. Dou, C. Hu, C. Yang, and J. Chen, "A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment," *Future Generation Computer Systems*, vol. 88, pp. 636–643, 2018.

[23] W. Jiang, G. Wang, M. Z. A. Bhuiyan, and J. Wu, "Understanding graph-based trust evaluation in online social networks: Methodologies and challenges," *ACM Computing Surveys*, vol. 49, no. 1, 2016.

[24] C. Yan, X. Cui, L. Qi, X. Xu, and X. Zhang, "Privacy-Aware Data Publishing and Integration for Collaborative Service Recommendation," *IEEE Access*, vol. 6, pp. 43021–43028, 2018.

[25] E. Luo, Q. Liu, and G. Wang, "Hierarchical Multi-Authority and Attribute-Based Encryption Friend Discovery Scheme in Mobile Social Networks," *IEEE Communications Letters*, vol. 20, no. 9, pp. 1772–1775, 2016.

[26] T. Peng, Q. Liu, D. Meng, and G. Wang, "Collaborative trajectory privacy preserving scheme in location-based services," *Information Sciences*, vol. 387, pp. 165–179, 2017.

[27] W. Gong, L. Qi, and Y. Xu, "Privacy-Aware Multidimensional Mobile Service Quality Prediction and Recommendation in Distributed Fog Environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 3075849, 8 pages, 2018.

[28] J. Kaur and K. Kaur, "A fuzzy approach for an IoT-based automated employee performance appraisal," *Computers, Materials and Continua*, vol. 53, no. 1, pp. 24–38, 2017.

[29] 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.

[30] 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.

[31] 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, no. 1, pp. 20632–20640, 2018.

[32] 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.

[33] 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.

[34] 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.

[35] J. R. Cheng, R. M. Xu, and X. Y. Tang, "An Abnormal Network Flow Feature Sequence Prediction Approach for DDoS Attacks Detection in Big Data Environment," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 95–119, 2018.

[36] B. Jia, X. Huang, R. Liu, and Y. Ma, "A DDos attack detection method based on hybrid heterogeneous multiclassifier ensemble learning," *Journal of Electrical and Computer Engineering*, vol. 2017, Article ID 4975343, 9 pages, 2017.

[37] S. Yu, Y. Tian, S. Guo, and D. O. Wu, "Can we beat DDoS attacks in clouds?" *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 9, pp. 2245–2254, 2014.

[38] J. R. Cheng, J. Yin, Y. Liu et al., "Detecting distributed denial of service attack based on address correlation value," *Journal of Computer Research and Development*, vol. 46, no. 8, pp. 1334–1340, 2009.

[39] J. Cheng, X. Tang, and J. Yin, "A change-point DDoS attack detection method based on half interaction anomaly degree," *International Journal of Autonomous and Adaptive Communications Systems*, vol. 10, no. 1, pp. 38–54, 2017.

[40] The Cooperative Association for Internet Data Analysis, *The Caida Ucsd 'DDoS Attack 2007' Dataset*, 2007.

[41] F. Pedregosa, G. Varoquaux, and A. Gramfort, "Scikit-learn: machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[42] C. C. Chang and C. J. Lin, "LIBSVM: a Library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 27, pp. 1–27, 2011.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 3210207, 13 pages
https://doi.org/10.1155/2018/3210207

Research Article

A Sequence Number Prediction Based Bait Detection Scheme to Mitigate Sequence Number Attacks in MANETs

Rutvij H. Jhaveri,[1] **Aneri Desai**,[2] **Ankit Patel**,[2] **and Yubin Zhong**[3]

[1]*Delta-NTU Corporate Laboratory, Nanyang Technological University, Singapore 639798*
[2]*SVM Institute of Technology, Bharuch 392001, India*
[3]*Guangzhou University, Guangzhou 510006, China*

Correspondence should be addressed to Yubin Zhong; zhong_yb@163.com

Received 25 June 2018; Accepted 15 October 2018; Published 15 November 2018

Guest Editor: Lianyong Qi

The characteristics of MANET such as decentralized architecture, dynamic topologies make MANETs susceptible to various security attacks. Sequence number attacks are such type of security threats which tend to degrade the network functioning and performance by sending fabricated route reply packets (RREP) with the objective of getting involved in the route and drop some or all of the data packets during the data transmission phase. The sequence number adversary attempts to send a fabricated high destination number in the RREP packet which attracts the sender to establish a path through the adversary node. This paper proposes a proactive secure routing mechanism which is an improvement over the authors previously proposed scheme. It makes use of linear regression mechanism to predict the maximum destination sequence number that the neighboring node can insert in the RREP packet. As an additional security checkpoint, it uses a bait detection mechanism to establish confidence in marking a suspicious node as a malicious node. The proposed approach works in collaboration with the ad hoc on-demand distance vector routing (AODV) protocol. The simulation results depict that the approach improves the network performance in the presence of adversaries as compared to previously proposed scheme.

1. Introduction

The use of wireless network has increased tremendously due to the nonrestriction of the nodes to be stagnant physically [1]. MANETs are such infrastructure-less wireless networks where the communication between the nodes is performed through multihop paths [2]. MANETs have gained popularity in various domains such as military operations, natural calamities, maritime communications, vehicular computing, and remote weather forecasting due to the properties such as dynamic topology, easy configuration of nodes, and distributed administration [3, 4]. Despite the popularity of MANETs, its characteristics bring various vulnerabilities to its doorstep [5, 6].

In a MANET, each and every node has the responsibility to route the packets [7]. The routing protocols in MANET are divided into two major categories, namely, proactive routing protocols and reactive routing protocols [8]. The proactive protocols have per-defined routes between the nodes in the network whereas the reactive protocols establish on-demand routes; i.e., they are created when there is a need of communication between the nodes. The predefined routes may waste the network resources if no communication takes place through that route. As a result, the reactive routing protocols have gained more popularity for such networks [4]. However, the reactive routing protocols are prone to different types of attacks.

An adversary may take the benefit of the nodes being routers and perform many malicious activities to hinder the smooth communication between the nodes. This is due to the fact that the normal legitimate nodes may come under the influence of the adversaries and get compromised as there are no security mechanisms present in the traditional routing protocols [9, 10]. The issue of data privacy also exists in the infrastructure-less networks such as MANETs [11, 12]. Many researchers have done their research in finding

the solutions that addresses these various issues [13–15]. In order to facilitate smooth communication in the presence of such adversary nodes, various secure routing algorithms are proposed to overcome the negative effects of the adversaries. The cryptographic approaches are casually used to provide confidentiality in the network [13, 16]. The use of hashing mechanism is also used to resolve the privacy issues in the smooth communication of data between mobile nodes and vehicles [17]. In addition, cluster management and classification based techniques are also used to overcome the negative effects caused due to the dynamic topology of the nodes in a MANET [18, 19]. Moreover, many secure routing approaches have been proposed to achieve quality-of-services (QoS) by addressing the availability issue infringed by denial-of-service (DoS) attacks [8].

The sequence number attack (such as grayhole attack or blackhole attack) is a type of DoS attack where the attacker's intention is to prohibit the benign node from receiving the data packets [8]. The sequence number attacks cause packet forwarding misbehaviors during data transmission with the sole intention to degrade the network performance [3]. In the initial phase, the adversary node first attempts to become the part of the route. To accomplish this task, the adversary sends a fabricated route reply packet (RREP) claiming that it has fresher route towards the destination [20]. The adversary node does this by sending an RREP packet with a fabricated destination sequence number which indicates a high level of freshness of the route. As a result, the source node gets the impression that the node sending RREP (the adversary node) has a fresher route towards the destination [8]. Thus, the adversary node, after entering in the route between the source and the destination starts packet dropping behavior.

Many researchers have designed different schemes to overcome the performance losses caused by the sequence number attacks by targeting the common routines that the adversary follows [3]. The use of fuzzy systems also helps in overcoming the sequence number attackers [21, 22]. Recently machine learning approaches have achieved a great deal of attention from the researchers to overcome the negative effects of the adversary nodes [23, 24]. The detection of the adversaries can be either performed during the route discovery phase (i.e., proactive manner) or during the transmission of data (i.e., reactive manner). As the reactive approaches tend to detect the adversaries after some packet loss, they compromise QoS of the network. In this paper, we propose a reactive approach which detects adversary nodes during the route discovery phase as critical applications such as industry control systems or military operations may not afford to lose data packets. The proposed scheme, sequence number prediction based bait detection scheme (SNPBDS), is an enhancement to our previous scheme, sequence number based bait detection scheme (SNBDS) [3]. SNPBDS incorporates an additional mechanism based on linear regression [25] which predicts the threshold value of the destination sequence number of the RREP packet. When a node sends RREP with higher sequence number compared to the predicted threshold value, the node is marked as a suspicious node. To confirm the adversary node as a malicious node, a bait detection scheme is employed. If the suspicious node is marked as a malicious node, it is excluded from the route and control packets received from that node are ignored.

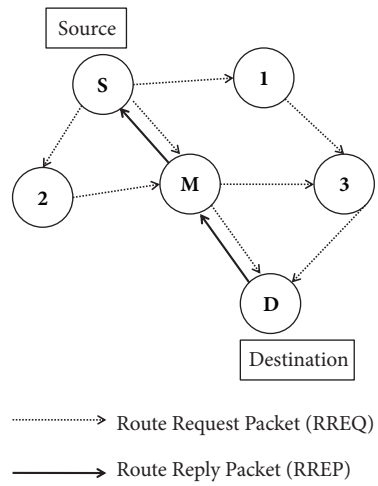


Figure 1: Operation of AODV protocol [2].

The paper is organized as follows. The working of the traditional sequence number based packet forwarding misbehavior attack is presented in Section 2. Section 3 presents the enhanced adversary model. Section 4 provides the related work and Section 5 presents the proposed approach followed by Simulation Results in Section 6. Finally Section 7 provides the conclusion to the paper.

2. Operation of the Sequence Number Attack

In MANETs adopting AODV routing protocol, the source node wishing to communicate to the destination first generates an RREQ packet and broadcasts the packet to its neighbors. The neighbors broadcast the request further until the packet reaches the destination or an intermediate node with a valid fresher path [2]. This node then replies with an RREP packet towards the reverse path to the source node. The RREP packet contains a destination sequence number which is used to denote the freshness of the route [4].

Figure 1 shows the route establishment in the AODV-based MANETs. The source node S generates an RREQ packet and broadcasts the packet to its neighboring nodes 1, 2, and M. These nodes pass the packets further and the RREQ packet reaches the destination D. The destination node selects the reverse path having the less hop count and, therefore, the RREQ from node 3 is discarded. Thus, the destination node D generates an RREP packet and forwards it to node E which then forwards the same to node S. In this way a path is formed as S-M-D for data communication.

As aforementioned, every RREP packet contains a destination sequence number to indicate freshness of the route. A sequence number adversary node in order to get involved in the route sends a fabricated RREP packet with a higher destination sequence number despite having a route towards the destination [2]. The operation of the AODV protocol in the presence of adversary node is shown in Figure 2. A legitimate internal node M turns into an adversary node which discards

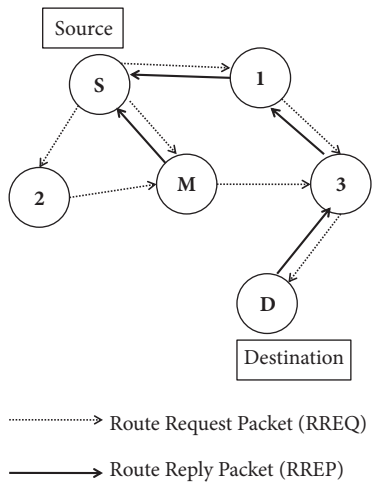


Figure 2: Operation of adversary during route discovery [3, 4].

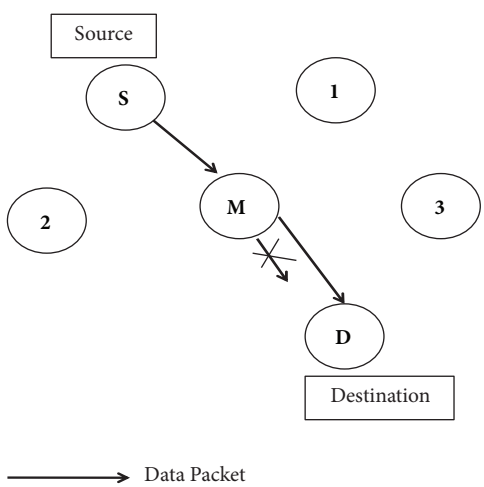


Figure 3: Operation of adversary during data transmission [3].

an RREQ packet which is supposed to be rebroadcasted to establish a path to the node D. Instead the adversary node M generates a forged reply packet with a higher destination sequence number and sends it on the reverse path towards the node S with a motive to deceive the source S that the node M is having a fresher valid route towards the destination D. As a result, the source node S gets the impression that node M has a fresher route to the destination D. On the other hand, the source node S ignores the benign RREP packet received from the node 1 which is generated by the destination node D as the RREP has a lower sequence number and higher hop count as compared to those received from the fabricated RREP generated by the adversary node M.

Once the route is formed through the adversary node M, the source node S starts sending the data packets. Node M after receiving the data packet may either forward or drop that packet. The same is illustrated in Figure 3. The adversary node may act as a genuine node for some time duration and as a malicious node for the remaining time [2, 20]. This unpredictive nature of the adversary makes its detection not so easy.

3. Related Work

Sequence number attacks degrade the network performance by taking the advantage of lack of security mechanism in the reactive routing protocols [3]. This has provided the motivation to researchers to incorporate distinct types of safety mechanisms in the routing protocols. In this section we discuss various security approaches which detect the adversary nodes either during the route discovery phase or during the data transmission phase.

3.1. Detection during Data Transmission Phase. An Extended Data Routing Information (EDRI) approach presented in [26] detects the adversaries by keeping the track of the data packets sent and received to and from the neighboring nodes in the EDRI table. This approach keeps the track of the neighboring nodes regarding the forwarding of the data packets with the help of promiscuous mode. If a neighboring node drops data packets more than predefined threshold, the neighboring node is considered as an adversary node. An enhancement to the EDRI approach is presented in [27] which includes a preventive mechanism along with the detection mechanism by using an alarm packet to alert all the nodes in the network about the detected malicious nodes with the help of data routing tables. A trust based approach is presented in [1, 28] where the nodes are assigned a trust value based on the past data communication. The trust value for the node is updated on the basis of the number of packets sent by the node. The node receiving the RREP accepts it if the forwarding node is marked as trusted node in the routing table; otherwise that RREP packet is discarded. A cooperation based defense mechanism (CBDM) scheme is presented in [29] where the cooperation value is calculated for every node using the probabilistic model. If the cooperation value of a node crosses the threshold value then that node is considered as suspicious node. As an additional check, a bait request is sent to the suspicious node and if the suspicious node replies to that request, then that node is considered to be malicious node. Another trust based approach is presented in [30] which makes the use of the contradiction mechanism where the data transmission is facilitated via the nodes having higher trust value. The trust value is calculated on the basis of the packets exchanged between the nodes.

3.2. Detection during the Route Discovery Time. The peak value calculation approach is presented in [31–33] where the node receiving the RREP packet calculates a threshold value of the destination sequence number. This threshold value is calculated with the help of the three parameters, namely, number of RREQs received and the number of RREPs received and the routing table sequence number. If the RREP received by the node contains a higher sequence number than the calculated threshold value, that RREP packet is discarded and the sender of that RREP packet is considered as a malicious node and that malicious node is excluded from the route. A cooperative bait detection scheme is presented in [34] where the source node selects the cooperating neighbor as the bait destination address. The source node then generates a bait request selecting the neighbor as the

```
Procedure 1: Actions by the malicious node after receiving an RREQ
    Discard the received RREQ
    If (RREQ is NOT for me) then
        If (valid fresher route is available in the routing table) then
         Fill up RREP with Dest_Seqno=Routing_table_Dest_Seqno+Random(1,7) and Hop_Count=Random(1,3)
           Unicast the forged RREP on the reverse path to the source
        End If
    Else
    Fill up RREP with own Seqno and Hop_Count=1
       Unicast the genuine RREP on the reverse path to the source
    End If
Procedure 2: Actions by the malicious node after receiving a data packet from the source node
    If (data packet is NOT for me) then
    If (Packet_ID mod Random(1,3) == 0) then
    Drop the data packet received from the source
    Else
    Forward the data packet
    End If
    Else
    Receive the data packet for me
    End If
```

Algorithm 1: Operation of adversary during Route discovery and data transmission [1, 3] (Algorithm 1 is reproduced from Rutvij et al. (2015), ([under the Creative Commons Attribution License/public domain)).

destination and then broadcasts the bait request for a route to that destination. If the node receiving the bait request sends the reply, that node is considered as a malicious node. A graph based approach is presented in [35] where the nodes with their neighbors for a graph like structure where every node monitors the control packets delivery of the neighboring nodes. Based upon the frequency of the communication, the nodes are assigned a fielder value which helps in deciding the next hop for the discovering the route.

4. Adversary Model

An enhanced and powerful adversary model is provided in [1, 3, 10]. In this model, the adversary node, as soon as it receives an RREQ packet, it generates a fabricated RREP packet which will have a marginally higher sequence number to attract the source node to form a path through it. The adversary node may generate this RREP packet even though it does not have a route towards the destination.

In this adversary model, the attacker node just increments the value of the destination sequence number by a random smaller value which keeps the fake destination number marginally higher. The adversary node then adds the fabricated and fraudulent destination sequence number and hop count values into the RREP packet. This mode of operation makes the detection of an attacker's presence in the network more difficult. Once one or more adversary nodes get into the route they may pretend to be as a benign node for some time period and carry out packet forwarding misbehaviors for other time periods [3].

The operations of the adversary during the route discovery phase and during the data transmission phase are shown in Algorithm 1 [1]. As shown in the algorithm, when the adversary node receives an RREQ packet, it fetches the destination sequence number form the routing table and adds a marginally incremented random value to that in order to forge the destination sequence number for the RREP packet. In addition, it enters a random hop count field in the fabricated RREP packet. The adversary node thus fools the source node about having the fresher and shorter route to the destination, and, as a result, it becomes part of this bogus route. The adversary now starts packet forwarding misbehavior by dropping the data packets in a random way. The nature of this capricious adversary makes its detection very difficult.

5. Proposed Work

The proposed approach, sequence number prediction based bait detection scheme (SNPBDS), attempts to detect the adversary nodes during the route discovery phase. This proactive detection during route discovery is imperative in several critical applications where we cannot afford to lose the data packets.

SNPBDS provides advancement to the SNBDS scheme presented in [3]. SNPBDS adds various fields in the routing table and in the neighbor table. A field for recording the past sequence numbers for a node is added in the routing table and the status field is added in the neighbor table to mark the status of the neighboring node as normal, suspicious, or malicious. Whenever any node receives an RREQ or RREP packet for a destination node, the past data field in the routing table is updated. Using the past sequence number history, we use linear regression technique to predict the highest destination sequence number possible for the RREP packet sent by the replying node.

5.1. Linear Regression Technique of Predicting Sequence Number [25]. The linear regression is defined with the help of a plot on the X- and Y-axis. There are two lines of regression that of Y on X and X on Y. The line of regression of Y on X is given by Y = a_0+ a_1X where a_0 and a_1 are unknown constants known as intercept and slope of the equation. This is used to predict the unknown value of the variable Y when value of the variable X is known. The equation for prediction is as follows:

$$\mathrm{Y} = a_0 + a_1\mathrm{X} \tag{1}$$

The equations for calculating a_0 and a_1 are as follows:

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2} \tag{2}$$

$$a_0 = \overline{y} - a_1\overline{x} \tag{3}$$

Using (1), (2), and (3), we can predict the value Y which is based on X. In addition, to improve prediction we find error at every point and based on this error we improve our prediction. The equation for the calculating the error is as follows:

$$\min \sum_{i=1}^{n} e_i = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 x_i\right) \tag{4}$$

Equation (4) defines the error at a particular point. Based on the last error, we can improve our prediction by performing addition and subtraction of the error value to the predicted value. The equation for the final predicted value is as follows:

$$P = Y + e \tag{5}$$

5.2. Application of Linear Regression in SNPBDS. Using linear regression technique discussed above, we now tend to predict the threshold value of the destination sequence number which is sent in the RREP packet by the neighboring node. We calculate the destination sequence number based on the time factor. We assume time (denoted as T) as the value of X and the sequence number (denoted as N) as the value of Y in (1). For predicting the new value of N, we need the past records of N and T. Table 1 shows the past history of the values of T and N.

As shown in Table 1, as we have 5 records till now, we take n=5. Now we wish to predict the threshold value of the destination sequence number for the received RREP packet.

According to (1) we have

$$\mathrm{N} = a_0 + a_1\mathrm{T} \tag{6}$$

So now we first calculate the values of a_0 and a_1 using (2) and (3). According to (2) we have

$$\begin{aligned} a_1 &= \frac{n\sum T_i N_i - \sum T_i \sum N_i}{n\sum T_i^2 - \left(\sum T_i\right)^2} \\ a_1 &= \frac{5 \times 30147 - 432 \times 287}{5 \times 44202 - 287 * 287} \\ a_1 &= 0.19295 \end{aligned} \tag{7}$$

According to (3) we have

$$\begin{aligned} a_0 &= \overline{N} - a_1\overline{T} \\ a_0 &= 57.4 - 0.19295 \times 86.4 \\ a_0 &= 40.72912 \end{aligned} \tag{8}$$

Replacing the values of (7) and (8) in (1) we have

$$N = 40.72912 + 0.19295 \times T \tag{9}$$

We want to predict the value of the destination sequence number at time of 160 seconds. Therefore, we take the value of T=160.

$$\begin{aligned} N &= 40.72912 + 0.19295 \times 160 \\ N &= 71.60112 \end{aligned} \tag{10}$$

Even though we calculated the value of N, it contains some error. To address it, we use (4). We calculate the error at time 138 and the sequence number at that time is 96. Therefore, we calculate

$$\begin{aligned} e &= 96 - 40.72912 - 0.19295 \times 138 \\ e &= 28.64378 \end{aligned} \tag{11}$$

Now accumulating the error in the predictive value according to (5) we get

$$\begin{aligned} P &= N + e \\ P &= 71.60112 + 28.64378 \\ P &= 100.2449 \end{aligned} \tag{12}$$

Thus by the use of linear regression technique, at time of 160 seconds, the predicted threshold of the destination sequence number is 100.

5.3. SNPBDS Methodology. This section describes the operations of the nodes adopting SNPDS while receiving RREQ and RREP packets.

5.3.1. Actions Performed by the Node Receiving the RREQ Packet. When a node receives an RREQ packet, it first checks the status of the node sending the RREQ packet in the neighbor table. If the status of the node in the neighbor table is marked as malicious, the node discards that RREQ packet. If the status of the node that has sent the RREQ packet has its status as normal, then the node receiving the RREQ packet will update the routing table entry for that particular destination.

(1) Algorithm. The steps followed by the node after receiving RREQ packet are shown in Algorithm 2.

5.3.2. Actions Performed by the Node Receiving the RREP Packet. The node receiving an RREP packet checks the status of the node forwarding the RREP packet in the neighbor table. If the status for that node is malicious,

```
Procedure 1: Actions by the node after receiving an RREQ
    Retrieve the Status of the node forwarding RREQ packet
    If (Status == malicious ) then
        Discard the RREQ packet
    End If
    Else If (valid fresher route is available in the routing table) then
        Generate the RREP packet and forward it towards the Source Node.
    End If
    Else
        Update the Routing Information and forward the RREQ further.
Exit
```

Algorithm 2: Operation of node after receiving RREQ packet.

```
Procedure 1: Actions by the node after receiving an RREQ
Retrieve the Status of the node forwarding RREP packet
If (Status == malicious) then
    Discard the RREP packet
End If
Else If (Status==normal) then
    Evaluate the predictive seq. number (PRED_SEQNO)
End If
If (DEST_SEQNO > PRED_SEQNO)
    Status=suspicious
    Update the Status of the node in the neighboring table.
End If
Else
    Perform the normal routing operations.
If (Status = suspicious)
    Send BAIT_REQUEST to suspicious node
End if
If (BAIT_REPLY received)
    Change the status of the node from suspicious to malicious
    Delete the routing entry for the malicious node.
    Initiate a local route discovery process to find an alternate route to the destination.
End If
Else
    Change the Status from suspicious to normal and perform regular routing operations
Exit
```

Algorithm 3: Operation of node after receiving RREP packet.

the received RREP packet is discarded. If the status value for the forwarding node is normal, the linear regression technique is employed to predict the threshold value of the destination sequence number based on the historical data. If the predicted destination sequence number is greater than the destination sequence number in the received RREP packet, the routing table is updated if necessary and the RREP is forwarded towards the source node. If the predicted sequence number is less than the destination sequence number received in the RREP packet, the receiving node marks the status of the node sending RREP packet as *suspicious*. The receiving node then sends a bait (forged) request packet to the suspicious node. If the suspicious node responds to the bait request, status of the suspicious node is changed from *suspicious* to *malicious* in the neighboring table and the RREP is discarded. The routing table entry having the malicious node as next hop node is then deleted and a local route discovery process is initiated to discover an alternate route. However, if the suspicious node does not reply to the bait request, the *suspicious* status of the node is changed back to *normal*. The steps followed by the node receiving RREP packet are depicted in Algorithm 3.

(1) Algorithm. See Algorithm 3.

5.4. Illustrative Example. As shown in Figure 4, the source node S wants to communicate to destination node D. The source node S generates the route request packet RQ1 and broadcasts it to its neighbor nodes 1 and 2. Nodes 1 and 2 then add the necessary information in RQ1 and generate the

Table 1: Historical data based on time.

T	N	T∗N	T∗T
30	17	510	900
67	49	3283	4489
85	58	4930	7225
112	73	8176	12544
138	96	13248	19044
Total = 432	**Total = 287**	**Total = 30147**	**Total = 44202**

Table 2: Parameters of request and reply packets.

	RQ1	RQ2	RQ3	RQ4	RP1	RP2	RP3	RP4
Source IP	S	1	2	3	D	3	1	M
Dest Seq, No.	15	15	15	15	17	17	17	17+5=22
Origin IP	S	S	S	S	S	S	S	S
Destination IP	D	D	D	D	D	D	D	D
Hop Count	1	2	2	3	1	2	3	2

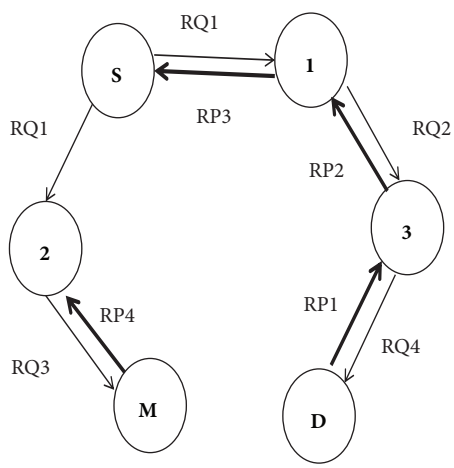


Figure 4: Route discovery process for a path from node S to node D.

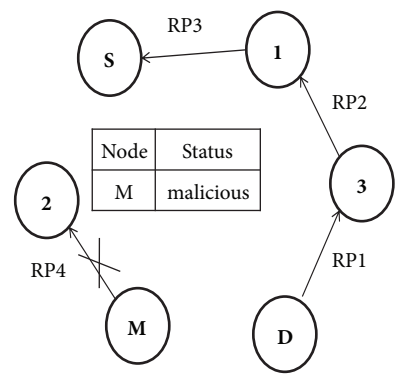


Figure 5: Avoiding the RREP from the malicious node.

packet RQ2 and RQ3, respectively, and forward it to their respective neighboring nodes 3 and M. Node 3 after adding the necessary information in RQ2 generates the packet RQ4 and forwards it to the destination node D. The destination node D generates the RREP packet and sends that packet through node 3 to the source node S as shown in Figure 4.

Node M behaving mischievously does not forward the RQ3 packet to node D. Rather it discards the request packet and generates a fabricated reply packet RP4 and sends it to node 2 as shown in Figure 7. The malicious node M randomly increments the destination sequence number by 3 and sets the hop count to 2 and inserts this fabricated information in the RREP packet RP4

The contents of the route request packets (RQ1, RQ2, RQ3, and RQ4) and route reply packets (RP1, RP2, RP3, and RP4) are shown in Table 2.

Node 2, after receiving the fabricated reply packet RP4, checks the status value for node M in the neighbor table. If the status value is equal to malicious, node 2 immediately discards the reply packet which is shown in Figure 5.

If the status value of node M in the neighbor table of node 2 is normal, node 2 applies the linear regression technique to predict the value of the destination sequence number. Node 2 now predicts the threshold value of the destination sequence number by considering the past history data of the sequence numbers. The collection of such data is shown in Table 3. The table shows that the predicted value of the destination sequence number is 10 which is less than the destination sequence number received in RP4 sent by M. Therefore, node 2 marks the status of node M as *suspicious* in the neighbor table.

As shown in Figure 6, when status of the node M changes from normal to suspicious, node 2 generates a bait request packet BRQ1. This is a dummy request packet to verify whether the suspicious node blindly replies to the request or not.

Node M, which is marked as suspicious, after receiving the bait request generates a reply BRP1 and sends it to node 2 as shown in Figure 7. Node 2 receives the packet BRP1 in reply

Table 3: Collection of past sequence numbers.

Node	Dest	Time	Seq No	Historical Data	
				Time	Seq No.
				55	6
M	D	115	10	...	...
				115	8
				115	10

Table 4: Parameters of bait request packet and its reply.

	BRQ1	BRP1
Source IP	4	4
DestSeq, No.	22	24
Origin IP	S	S
Destination IP	D	D
Hop Count	0	0

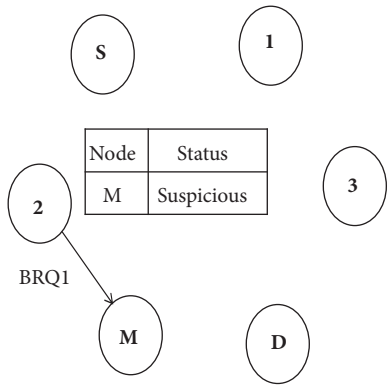


Figure 6: Node 2 sends bait request to node M.

to the bait request. Therefore, node 2 now marks the status of the node M as malicious and updates the value of status from suspicious to malicious.

The parameters of BRQ1 and BRP1 are shown in Table 4.

Node 2 after marking the status of node M as malicious discards the RREP packet and deletes the routing table entry having the node M as the next hop node. As a result, the node M is not allowed to enter the route. Node 2 now initiates a local route discovery process for the destination for which the routing table entry is discarded

6. Simulation Results and Analysis

6.1. Experimental Setup. In our experiments, we carry out simulations on the NS-2 simulator [36]. In order to prove that the SNPBDS approach provides better performance compared to the SNBDS approach, we compare the performance of both the approaches by varying various network parameters. For our experimental work, we select the maximum simulation time of 200 seconds with the terrain area of 1500 m x 1500 m. The performance of SNPBDS approach is compared with the simple AODV protocol, the AODV protocol with the adversary, and the SNBDS approach. The

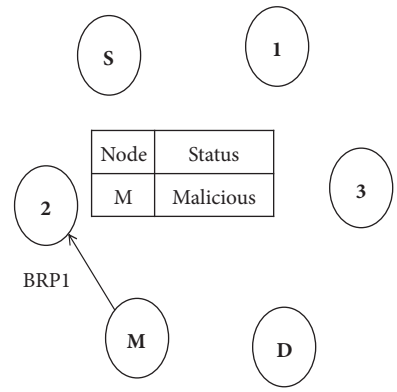


Figure 7: Node M replies to bait request.

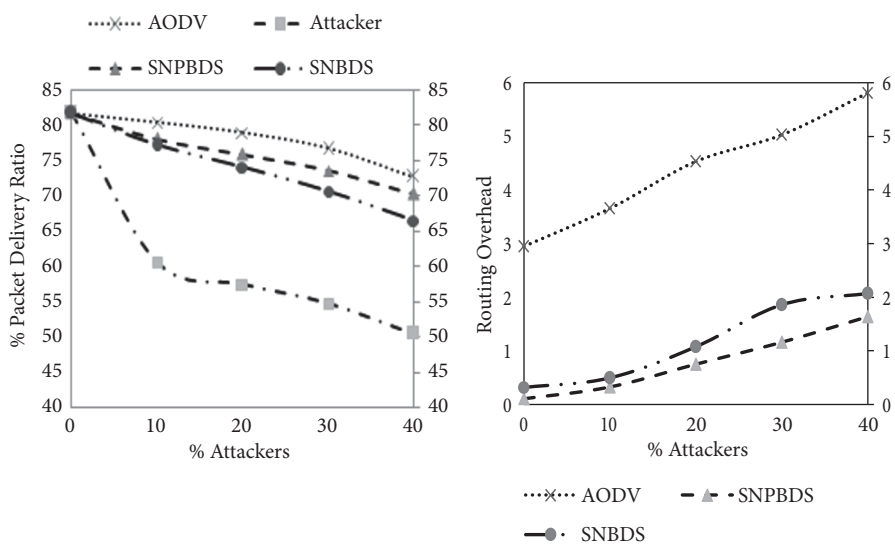
performance comparison of various approaches is based on the performance metrics such as packet delivery ratio and routing overhead. The detailed simulation parameters are shown in Table 5.

6.2. Result Analysis. We perform various tests to evaluate the performance AODV protocol, AODV protocol with adversary node, SNBDS approach, and the SNPBDS approach. The metrics selected for the evaluation of the approaches are the packet delivery ratio (PDR) and routing overhead. Packet delivery ratio (PDR) is defined as the ratio of the number of packets received by the destination to the number of packets sent by the source node [2]. Routing overhead refers to the ratio of the control packets transmitted to the ratio of the data packets transmitted [2]. The various test cases for the evaluation of the performance of different approaches are discussed below.

6.2.1. Test 1: Varying Number of Adversary Nodes. Figure 8(a) shows the graph of the packet delivery ratio of the various protocols. We have evaluated the PDR by varying the attacker count. The number of nodes in the network is 100. The range of the number of attacker nodes varies from 0% to 40% of the number of nodes in the network. Figure 8(a) shows the decrease of the PDR with the increase of the number of attacker nodes. The PDR of the AODV protocol in the presence of adversaries decreases from 80% to 50% with the increase in the number of adversaries. The SNPBDS approach provides the PDR in the range of 83% to 70%, The graph shows that the PDR of the SNPBDS approach is higher than the SNBDS approach. Figure 8(b) shows the graph of the routing overheard of the network operating

Table 5: Simulation parameters.

Parameters	Values
Simulator	NS 2.35
Routing Protocols	AODV, Attacker1, SNBDS, SNPBDS
Coverage Area	1500m x 1500 m
Mobility Model	Random Way Point
Simulation Time	200s
Number of nodes (varying)	50 – 100
Maximum Mobility (varying)	5 m to 25 m/s
Pause time (varying)	5 -25 s
No. of Connections (varying)	2 to 10
Transmission Rate (varying)	5 to 25 packets per second



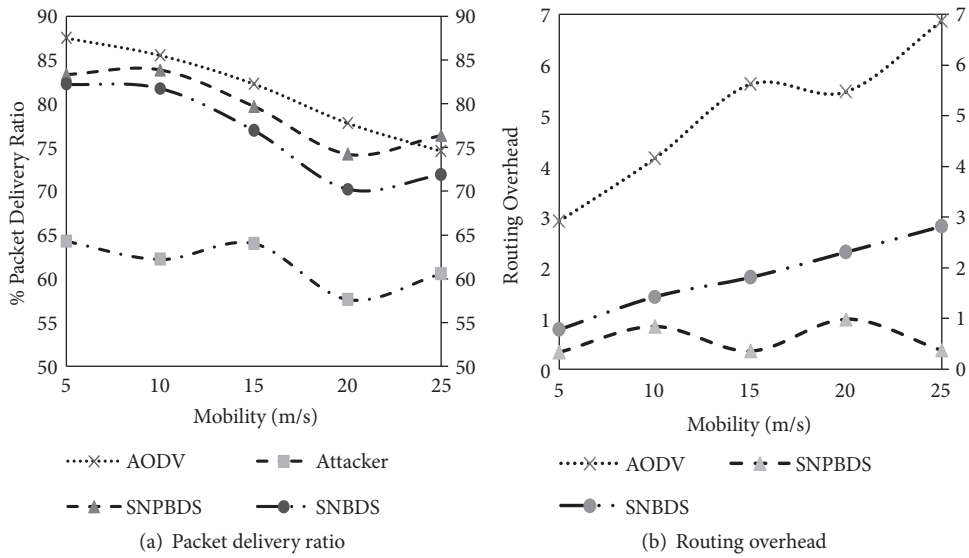
(a) Packet delivery ratio (b) Routing overhead

Figure 8: Performance comparison by varying number of adversary nodes.

with different protocols by varying the number of attacker nodes in the network. The routing overhead in the AODV protocol ranges from 3.0 to 6.0 with the increase in the number of adversaries. The routing overhead of the SNPBDS approach is in the range of 0 to 1.5 whereas the routing overhead of SNBDS approach falls in the range of 0.2 to 2.0. Thus SNPBDS protocol produces lower routing overhead compared to AODV protocol and SNBDS protocol. This is because the SNPBDS protocol eliminates the malicious node which results in the reduction of the frequency of route discovery which in turn leads to lower routing overhead.

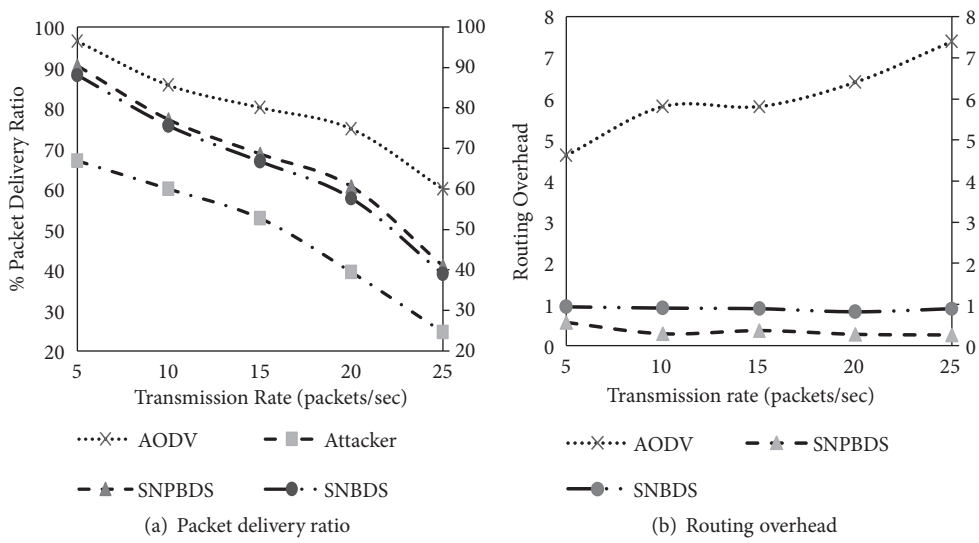
6.2.2. Test 2: Varying Mobility Speed. Figure 9(a) shows the PDR of the protocols by keeping the number of nodes and the number of attacker nodes fixed and varying the mobility speed. The number of nodes is 100 and the attacker nodes count is 10% of the total number of nodes. We vary the mobility speed of the nodes from 5m/s to 25m/s. We can see that, with the increase in the mobility speed, the PDR of AODV protocol gradually decreases from around 90% to 70%. With the attacker's interference, the PDR appears to be in the range of 60 to 65% with the varying speeds of the nodes. The SNBDS approach has PDR range of 70% to 82%. The SNPBDS approach provides the PDR of around 76% to 83%. Thus the performance of SNPBDS is better than the SNBDS approach. Figure 9(b) shows the routing overhead of the network by varying the mobility speed while keeping other parameters intact. The routing overhead of the SNPBDS approach is in the range of 0 to 1 which is better compared to SNBDS approach having routing overhead in the range of 1 to 3 and AODV protocol having the routing overhead in the range of 3 to 7 with the increase in the mobility speed. The results show that the effect of increase of mobility speed does not have a great impact on the value of routing overhead whereas, in AODV protocol and SNBDS approach, the routing overhead increases with the increase in the mobility speed which is due to the larger number of route discoveries.

6.2.3. Test 3: Varying Transmission Rate. Figure 10(a) shows the packet delivery ratios of various protocols under the effect of variable transmission speed. Varying the number of packets sent per unit time also impacts the performance of



(a) Packet delivery ratio (b) Routing overhead

Figure 9: Performance comparison by varying mobility speed.

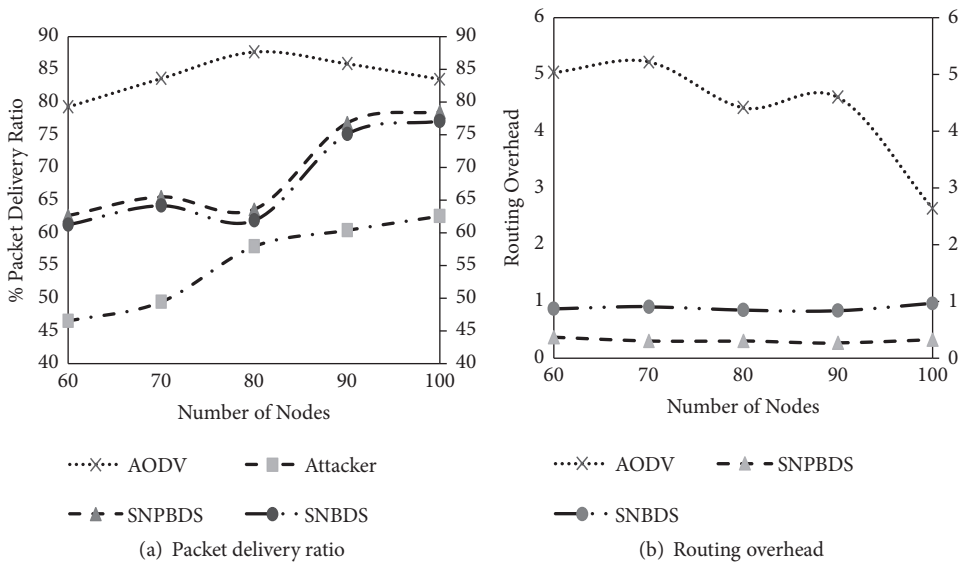


(a) Packet delivery ratio (b) Routing overhead

Figure 10: Performance comparison by varying transmission rate.

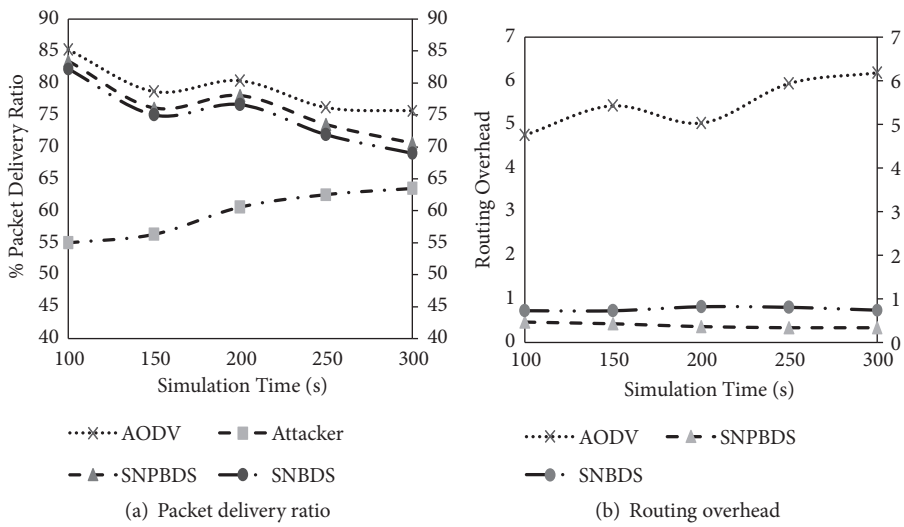
the network. We take the same number of nodes as 100 and 10% nodes as attacker nodes. Here we take a constant mobility speed for all the nodes and we vary the transmission speed from 5 packets per second to 25 packets per second. From the figure we observe that the PDR of all the protocols tend to decrease with the increase of the transmission speed. The AODV protocol provides the PDR in the range of 65 to 90%. In the presence of attacker nodes, the PDR declines from around 60% to 25% with the increase in the transmission speed. The SNBDS approach has the PDR in the range of 66% to 80%. The SNPBDS approach results in the PDR in the range of 70% to 81%. Thus in the presence of attacker nodes and by varying the transmission speeds, the SNPBDS approach provides better performance compared to SNBDS approach. Figure 10(b) shows the routing overhead incurred in the network while varying the transmission rate of packets and keeping other parameters intact. The routing overhead of the AODV protocol with the variation in the transmission time ranges from 4.5 to 7.5 which is very high compared to SNBDS approach having routing overhead of 1. The SNPBDS has the lowest routing overhead of 0.1 to 0.2. Figure 10 shows that the SNPBDS approach results in the steady routing overhead with the increase in the transmission rate of packets

6.2.4. Test 4: Varying Number of Nodes. Figure 11(a) shows the performance of the network by varying the number of nodes in the network. We take 10% of the nodes as the adversary nodes. The mobility speed and the transmission speed of the nodes are kept the same. The number of nodes varies from 60 to 100. We observe that the PDR of the network in the



(a) Packet delivery ratio (b) Routing overhead

Figure 11: Performance comparison by varying number of nodes.



(a) Packet delivery ratio (b) Routing overhead

Figure 12: Performance comparison by varying simulation time.

attacker's presence is very low in the range of 40 to 60%. The SNBDS approach results in the PDR range of approximately 60 to 75% while the SNPBDS approach results in the PDR range of 70 to around 80%. Thus SNPBDS approach provides better results compared to SNBDS approach. Figure 11(b) depicts the routing overhead in the network obtained by varying the number of nodes in the network. The results show that the AODV protocol has lower routing protocol with the increase in the number of nodes. This is because as the number of nodes increases, the network becomes denser and nodes have path to majority of the destinations which results in sending of RREP packet from the intermediate nodes. As a result the RREP does not reach the destination which results in lower number of RREQ and RREP packets. This reduces the routing overhead in AODV protocol. The SNPBDS approach has better results compared to AODV and SNBDS approach. This is because the attacker is eliminated from the route during the route discovery which would result in increase of data packets without rediscovering the route.

6.2.5. Test 5: Varying Simulation Time. Figure 12(a) shows the performance of the network by varying the simulation time. We keep all the parameters as fixed and just vary the simulation time from 100 seconds to 300 seconds. The results show that the PDR reduces with the increase in the simulation time. The PDR in the presence of adversaries without any security mechanism falls in the range of 55% to 63%. The SNPBDS approach provides the PDR in the range of 70% to 84% compared to the SNBDS approach which provides the PDR in the range of 67% to 83%. Thus the

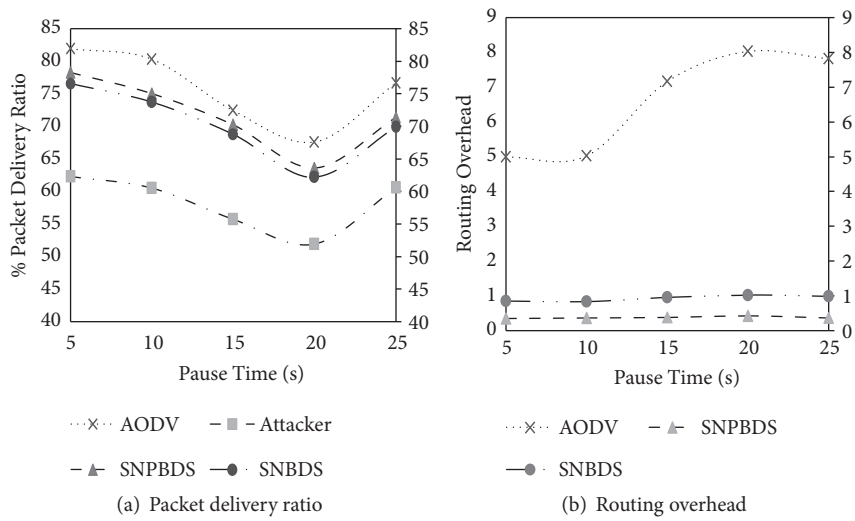


Figure 13: Performance comparison by varying pause time.

performance of SNPBDS is again better than the performance of the AODV protocol under the presence of adversaries and the SNBDS protocol. Figure 12(b) depicts the routing overhead of the network operating with the AODV, SNBDS, and SNPBDS approach by varying the simulation time and keeping other parameters intact. The routing overhead of the ADOV protocol goes from 4.7 to 6.0 with the increase of the simulation time. The routing overhead of the SNBDS approach is approximately around 0.8 to 0.9 which is higher compared to the SNPBDS approach which provides the routing overhead of 0.1 to 0.3. The results show that the SNPBDS approach has lower routing overhead due to the fact that the prediction algorithm will work for the entire simulation and the more the simulation time we have the more the past data we will have and the closer the value of predicted sequence number we will have. So this would result in lower routing overhead compared to other approaches.

6.2.6. Test 6: Varying Pause Time. Figure 13(a) shows the performance of the network by varying the pause time. The pause time is varied from 5 seconds to 25 seconds while keeping the other parameters intact. The PDR of the SNPBDS resides in the range of 72% to 78% which is better compared to SNBDS approach having the PDR range of 70% to 76% and AODV protocol with adversaries having PDR in the range of 60% to 62%. The results show that the PDR in SNPBDS approach is better than the SNBDS approach and the AODV protocol in the presence of attacker nodes. Figure 13(b) shows the routing overhead incurred in the network by varying the pause time while other parameters are kept intact. The SNPBDS approach produces the lower routing overhead of 0.2 compared to SNBDS approach having the routing overhead of 1. The AODV protocol provides very high routing overhead of 5 to 8 with the variation in the pause time. The SNPBDS approach provides the lowest routing overhead compared to the three approaches shown in Figure 13(b).

7. Conclusion

The nodes in MANET need to depend on other nodes to facilitate communication in the network. The characteristics of MANET provide great value to the adversaries which tend to degrade the network performance. Our proposed proactive scheme (SNPBDS) counters the threat of such adversaries by predicting adversaries in the route discovery phase. The proposed scheme attempts to prevent the adversaries form entering the route and, hence, increases the packet delivery rate and thereby the quality-of-services. The prediction of the destination sequence number and the bait request provide a double security check to confirm the status of the node as malicious. The scheme is evaluated under various network conditions against a strong adversary model. The performance evaluation of SNPBDS against SNBDS shows that SNPBDS provides considerable improvement packet delivery rate and normalized routing overhead.

The scheme can be enhanced by implementing hybrid approach (proactive and reactive) which would provide two-layer security during route discovery as well as data transmission.

Data Availability

Data used to support the findings of this study are available upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] R. H. Jhaveri and N. M. Patel, "Attack-pattern discovery based enhanced trust model for secure routing in mobile ad-hoc networks," *International Journal of Communication Systems*, vol. 30, no. 7, 2017.

[2] A. D. Patel and R. H. Jhaveri, "Addressing packet forwarding misbehavior with two phase security scheme for AODV-based MANETs," *International Journal of Computer Network and Information Security*, vol. 8, no. 5, pp. 55–62, 2016.

[3] R. H. Jhaveri and N. M. Patel, "A sequence number based bait detection scheme to thwart grayhole attack in mobile ad hoc networks," *Wireless Networks*, vol. 21, no. 8, pp. 2781–2798, 2015.

[4] A. D. Patel and K. Chawda, "Blackhole and grayhole attacks in MANET," in *Proceedings of the International Conference on Information Communication and Embedded Systems (ICICES '14)*, pp. 1–6, Chennai, India, February 2014.

[5] R. H. Jhaveri and N. M. Patel, "Mobile ad-hoc networking with AODV: A review," *Internation Journal of Next Generation Computing*, vol. 6, no. 3, pp. 165–191, 2015.

[6] M. S. Khan, D. Midi, M. I. Khan, and E. Bertino, "Fine-grained analysis of packet loss in MANETs," *IEEE Access*, vol. 5, pp. 7798–7807, 2017.

[7] Z. Zhao, H. Hu, G.-J. Ahn, and R. Wu, "Risk-aware mitigation for MANET routing attacks," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 2, pp. 250–260, 2012.

[8] R. H. Jhaveri, S. J. Patel, and D. C. Jinwala, "DoS attacks in mobile ad hoc networks: A survey," in *Proceedings of the 2nd International Conference on Advanced Computing and Communication Technologies (ACCT '12)*, pp. 535–541, January 2012.

[9] R. H. Jhaveri, N. M. Patel, Y. Zhong, and A. K. Sangaiah, "Sensitivity Analysis of an Attack-Pattern Discovery Based Trusted Routing Scheme for Mobile Ad-Hoc Networks in Industrial IoT," *IEEE Access*, vol. 6, pp. 20085–20103, 2018.

[10] R. H. Jhaveri and N. M. Patel, "Evaluating energy efficiency of secure routing schemes for mobile ad-hoc networks," *International Journal of Next Generation Computing*, vol. 7, no. 2, pp. 130–143, 2016.

[11] B. Li, Y. Huang, Z. Liu, J. Li, Z. Tian, and S.-M. Yiu, "HybridORAM: Practical oblivious cloud storage with constant bandwidth," *Journal of Information Sciences*, 2018.

[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] 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.

[14] 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.

[15] 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.

[16] 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.

[17] Y. Zhang, J. Li, D. Zheng, P. Li, and Y. Tian, "Privacy-preserving communication and power injection over vehicle networks and 5G smart grid slice," *Journal of Network and Computer Applications*, vol. 122, 2018.

[18] F. Aftab, Z. Zhang, and A. Ahmad, "Self-organization based clustering in MANETs using zone based group mobility," *IEEE Access*, vol. 5, pp. 27464–27476, 2017.

[19] 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.

[20] A. D. Patel and K. Chawda, "Dual Security Against Grayhole Attack in MANETs," *Advances in Intelligent Systems and Computing*, vol. 309, no. 2, pp. 33–37, 2015.

[21] 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.

[22] M. N. Mejri and J. Ben-Othman, "GDVAN: a new greedy behavior attack detection algorithm for VANETs," *IEEE Transactions on Mobile Computing*, vol. 16, no. 3, pp. 759–771, 2017.

[23] Z. Liu, Z. Wu, T. Li, J. Li, and C. Shen, "GMM and CNN hybrid method for short utterance speaker recognition," *IEEE Transactions on Industrial Informatics*, 2018.

[24] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an, and H. Ye, "Significant Permission Identification for Machine Learning based android malware detection," *IEEE Transactions on Industrial Informatics*, 2018.

[25] Onlinestatbook.com, http://onlinestatbook.com/2/regression/intro.html.

[26] G. Singh Bindra, A. Kapoor, A. Narang, and A. Agrawal, "Detection and removal of co-operative blackhole and grayhole attacks in MANETs," in *Proceedings of the International Conference on System Engineering and Technology (ICSET '12)*, pp. 1–5, Bangdung, Indonesia, September 2012.

[27] A. D. Patel, R. H. Jhaveri, and S. N. Shah, "I-EDRI Scheme to Mitigate Grayhole Attack in MANETs," *Advances in Intelligent Systems and Computing*, vol. 309, no. 2, pp. 39–43, 2015.

[28] R. H. Jhaveri, N. M. Patel, and D. C. Jinwala, "A composite trust model for secure routing in mobile ad-hoc networks," in *Adhoc Networks*, J. H. Ortiz, Ed., chapter 2, pp. 19–45, Intech, 2017.

[29] J. P. Bhoiwala and R. H. Jhaveri, "Cooperation based defense mechanism against selfish nodes in DTNs," in *Proceedings of the 10th International Conference on Security of Information and Networks (SIN '17)*, pp. 268–273, October 2017.

[30] N. Schweitzer, A. Stulman, R. D. Margalit, and A. Shabtai, "Contradiction based gray-hole attack minimization for ad-hoc networks," *IEEE Transactions on Mobile Computing*, vol. 16, no. 8, pp. 2174–2183, 2017.

[31] W. Yang, G. Wang, M. Z. A. Bhuiyan, and K.-K. R. Choo, "Hypergraph partitioning for social networks based on information entropy modularity," *Journal of Network and Computer Applications*, vol. 86, pp. 59–71, 2017.

[32] Q. Liu, G. Wang, F. Li, S. Yang, and J. Wu, "Preserving privacy with probabilistic indistinguishability in weighted social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 5, pp. 1417–1429, 2017.

[33] H. Shen, C. Gao, D. He, and L. Wu, "New biometrics-based authentication scheme for multi-server environment in critical systems," *Journal of Ambient Intelligence and Humanized Computing*, vol. 6, no. 6, pp. 825–834, 2015.

[34] J.-M. Chang, P.-C. Tsou, I. Woungang, H.-C. Chao, and C.-F. Lai, "Defending against collaborative attacks by malicious nodes in MANETs: A cooperative bait detection approach," *IEEE Systems Journal*, vol. 9, no. 1, pp. 65–75, 2015.

[35] Y. Liu and W. Trappe, "Topology adaptation for robust ad hoc cyberphysical networks under puncture-style attacks," *Tsinghua Science and Technology*, vol. 20, no. 4, pp. 364–375, 2015.

[36] Isi.edu, https://www.isi.edu/nsnam/ns/.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 3723691, 13 pages
https://doi.org/10.1155/2018/3723691

Research Article

Test Sequence Reduction of Wireless Protocol Conformance Testing to Internet of Things

Weiwei Lin,[1,2,3] **Hongwei Zeng**,[1] **Honghao Gao**,[4,5] **Huaikou Miao**,[1] **and Xiaolin Wang**[1,2]

[1] *School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China*
[2] *Shanghai Key Laboratory of Computer Software Testing and Evaluating, Shanghai 201114, China*
[3] *Department of Information Science and Technology, Taishan University, Taian 271000, China*
[4] *Computing Center, Shanghai University, Shanghai 200444, China*
[5] *Shanghai Key Laboratory of Intelligent Manufacturing and Robotics, Shanghai 200072, China*

Correspondence should be addressed to Honghao Gao; gaohonghao@shu.edu.cn

Received 23 August 2018; Accepted 2 October 2018; Published 1 November 2018

Guest Editor: Lianyong Qi

Wireless communication protocols are indispensable in Internet of Things (IoT), which refer to rules and conventions that must be followed by both entities to complete wireless communication or service. Wireless protocol conformance testing concerns an effective way to judge whether a wireless protocol is carried out as expected. Starting from existing test sequence generation methods in conformance testing, an improved method based on overlapping by invertibility and multiple unique input/output (UIO) sequences is proposed in this paper. The method is accomplished in two steps: first, maximum-length invertibility-dependent overlapping sequences (IDOSs) are constructed, then a minimum-length rural postman tour covering the just constructed set of maximum-length IDOSs is generated and a test sequence is extracted from the tour. The soundness and effectiveness of the method are analyzed. Theory and experiment show that desirable test sequences can be yielded by the proposed method to reveal violations of wireless communication protocols in IoT.

1. Introduction

Wireless communication is essential and critical to Internet of Things (IoT) [1, 2]. The reliability of wireless communication transmission largely depends on whether the wireless communication protocol is implemented as specified. Conformance testing [3, 4] is widely used to check whether an implementation conforms to its specification in areas such as traditional communication protocols and reactive systems; i.e., there must be the same behavior in the implementation for any I/O behavior observed in the specification. In FSM-based conformance testing, a protocol is called a specification and is expressed as a Finite State Machine (FSM), while an implementation under test is considered to be a "black box", the I/O behavior of which can only be observed. A test sequence is an I/O sequence such that whether an implementation conforms to the specification may be concluded by delivering the input sequence of the test sequence to the implementation and comparing the resulting output sequence with the output sequence in the test sequence.

Test techniques based on state identification [5–10] are well-known in FSM-based conformance testing. Test techniques based on state identification are supposed to identify every state and verify every transition of the specification in the implementation. A state is said to be identified when a state identifier is delivered to the state. A transition is said to be verified when the end state of the transition is identified. A state identifier of a state is a nonempty set of input sequences for the state such that the set of corresponding output sequences can characterize the state. Unique input/output (UIO) sequence is a popular state identifier such that UIO-based techniques are commonly used for test sequence generation.

Since wireless communication protocols can also be modeled by FSMs, FSM-based conformance testing is also applicable to wireless protocol conformance testing in

Table 1: UIO sequence of FSM corresponding to the connection and release process of Zigbee.

No.	States	UIO(s)
Ex1	s_1	i_1/o_1
Ex2	s_2	i_6/o_1
Ex3	s_3	i_5i_6/o_1
Ex4	s_4	i_2i_3/o_1o_2 i_2i_4/o_1o_3

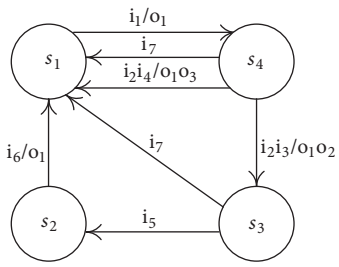


Figure 1: FSM corresponding to the connection and release process of Zigbee.

IoT. Zigbee is a typical wireless communication protocol and is divided into four layers: physical layer (PHY), media access control lay (MAC), network layer (NWK), and application layer (APL). The connection and release process of nodes in the MAC layer is modeled by the FSM in Figure 1 and UIO sequences of every state are listed in Table 1. On this basis, test sequences based on FSMs can be constructed using UIO sequences as state identifiers. There is a test sequence $s_1(i_1/o_1)s_4(i_2i_3/o_1o_2)s_3(i_5)s_2(i_6/o_1)s_1(i_1/o_1)s_4(i_2i_4/o_1o_3)s_1(i_1/o_1)s_4(i_7)s_1(i_1/o_1)s_4(i_2i_3/o_1o_2)s_3(i_7)s_1(i_1/o_1)s_4$ of the FSM in Figure 1 based on UIO sequences in Table 1. When the input sequence of this test sequence is applied to an implementation, every state of the FSM in Figure 1 can be identified and every transition of the FSM in Figure 1 can be verified in the implementation. Meanwhile, an output sequence can be obtained. It can be concluded whether the connection and release process of nodes is executed as the specification specifies by comparing the obtained output sequence with the expected one in the test sequence.

Test sequence reduction has long been an active research topic in FSM-based conformance testing. One approach is to convert the problem into the Rural Chinese Postman Problem from which a test sequence is extracted. For the purpose of transition verification every transition of an FSM is followed by an appended UIO sequence in the test sequence. Bo Yang et al. reduced test sequences by overlapping and multiple UIO sequences [11]. Benefiting from overlapping, more than one transition may be verified by a single appended UIO sequence. Hierons improved the method of UIO sequences through the use of an invertibility criterion, thereby achieving more overlapping [12], i.e., verifying even more transitions of an FSM with a single appended UIO sequence. However, multiple UIO sequences which are conducive to test sequence reduction are not considered by Hierons.

In order to reduce the cost of testing while assuring the effectiveness, this paper presents an improved method to generate reduced test sequences for wireless protocol conformance testing of IoT. Transitions in a test sequence are of three kinds: a copy of transitions in an FSM, UIO sequences which have been appended to the test sequence for the purpose of transition verification for the FSM, and the transitions which have been appended to the test sequence. In the improved method, invertibility is taken into account to leverage as much overlapping as possible such that all the transitions of an FSM can be verified with as few appended UIO sequences as possible. Rural symmetric augmentation is the main measure for test sequence generation. The replicated transitions during the rural symmetric augmentation are exactly the transitions for concatenation of test subsequences in the test sequence. More options of rural symmetric augmentation can be supplied by multiple UIO sequences than those supplied by single UIO sequences; i.e., a sensible choice of UIO sequences can lead to a minimum number of replicated transitions during a rural symmetric augmentation, and a minimum number of transitions for concatenation are achieved in the test sequence. In this way, further reduced test sequences are obtained by the proposed method.

The rest of the paper is organized as follows. In Section 2, the basic concepts and assumptions used in this paper are presented. The improved method is described with simple examples in Section 3, and the soundness and effectiveness of the method are discussed. Experimental evaluation is set out in Section 4. Then, the related work about the testing of IoT is reviewed in Section 5 and the paper is concluded briefly in Section 6.

2. Preliminaries

In this section, we introduce the definitions related to FSMs and graphs, together with the assumptions necessary for FSM-based conformance testing.

2.1. Definitions. An FSM is formally defined as a 6-tuple $M = (I, O, S, s_0, \delta, \lambda)$ where

(i) I and O are finite and nonempty sets of input symbols and output symbols, respectively;

(ii) S is a finite and nonempty set of states;

(iii) $s_0 \in S$ represents the initial state of M;

Table 2: UIO sequence of M_1.

No.	States	UIO(s)
Ex1	s_1	$a/0$
Ex2	s_2	$b/1$ $(a/1)(a/1)$ $(a/1)(b/1)$
Ex3	s_3	$(a/1)(a/0)$

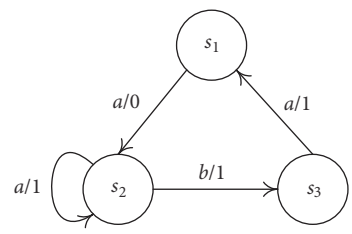


Figure 2: FSM M_1.

(iv) $\delta : S \times I \longrightarrow S$ denotes the state transition function;

(v) $\lambda : S \times I \longrightarrow O$ is the output function.

According to this definition, when an input symbol i is delivered to the current state s, M moves to the state $s' = \delta(s, i)$ with an output produced by $\lambda(s, i)$. The transition function and output function can be extended to finite input sequences, i.e., for an input symbol i, an input sequence $\beta \in I^*$ (where I^* is the set of finite sequences of input symbols), and a state s, $\delta(s, i\beta) = \delta(\delta(s, i), \beta)$, and $\lambda(s, i\beta) = \lambda(s, i)\lambda(\delta(s, i), \beta)$ where concatenation is denoted by juxtaposition.

A transition t is defined by a tuple $(s, s', i/o)$ where s is the start state, i is an input of s, $o = \lambda(s, i)$ is the associated output, and $s' = \delta(s, i)$ is the end state. A transition $(s, s', i/o)$ is invertible if it is the unique transition ending at s' with input i and output o.

A UIO sequence of a state is an input/output sequence such that the input/output behavior exhibited by the state is unique, i.e., given a UIO sequence β of a state s, and the input sequence of β is denoted by β_{in}; for any state $s' \neq s$, it is always true that $\lambda(s, \beta_{in}) \neq \lambda(s', \beta_{in})$. A UIO sequence is irreducible if it is not a UIO sequence anymore when the end i/o symbol of the sequence is deleted. UIO sequences in this paper will be supposed irreducible unless otherwise specified. There may be more than one UIO sequence for a state and the UIO sequences may be of different length.

An example FSM M_1 is described in Figure 2 where $S = \{s_1, s_2, s_3\}$, $I = \{a, b\}$, $O = \{0, 1\}$ and s_1 is the initial state of M_1. There are four invertible transitions $(s_1, s_2, a/0)$, $(s_2, s_2, a/1)$, $(s_2, s_3, b/1)$, and $(s_3, s_1, a/1)$. The UIO sequences for every state are shown in Table 2.

An FSM M can be perceived as a labeled, directed graph G. A state of M is represented as a node of G, and there is an edge labeled with i/o from node s to s' in G if and only if $\delta(s, i) = s'$ and $\lambda(s, i) = o$ in M, where s and s' are the start and end node of the edge. The numbers of incoming and outgoing edges of a node are called the in-degree and out-degree of the node, respectively. If the in-degree equals out-degree at each node then the directed graph is symmetric. Suppose that G is an asymmetric directed graph; if G^* is a symmetric directed graph generated from G by making copies of the edges then G^* is a symmetric augmentation of G. When the total cost of copies of edges is minimized, G^* is said to be a minimal symmetric augmentation of G. Suppose that E is the set of edges of G and $E' \subseteq E$, if G^* is a symmetric directed graph generated from G by making copies of the edges such that each edge in E' is included in G^* at least once and the total cost of copies of edges is minimized then G^* is called a rural symmetric augmentation of G. A sequence of contiguous edges $\beta = (s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2) \ldots (s_{k-1}, s_k, i_{k-1}/o_{k-1})$ forms a path of G where s_1 and s_k are the start and end node of the path, respectively. A path that starts and ends at the same node forms a tour, furthermore, if the tour traverses every edge of G exactly once then it is an Euler tour. A rural postman tour is a tour that traverses a given set of edges at least once. The Rural Chinese Postman Problem is to find a minimum-length rural postman tour for a given set of edges. FSMs and their directed graph representations are used interchangeably throughout this paper.

2.2. Assumptions. Given a specification FSM M with n states, the fault domain $\Psi(I)$ of M is the set of all possible implementations of M over the input alphabet I of M. $\Psi_n(I)$ refers to the implementations with up to n states in $\Psi(I)$. This paper only focuses on implementations in $\Psi_n(I)$.

An FSM M is strongly connected if for any two distinct states s and s' there is an input sequence β that takes M from s to s'; i.e., $\forall s, s' \in S, s \neq s', \exists \beta \in I^* \cdot (\delta(s, \beta) = s')$.

Two states s and s' are equivalent if for any input sequence there are always the same output sequences from s and s'. An FSM M without equivalent states is said to be minimal or reduced; otherwise M is reducible by joining equivalent states.

An FSM M is deterministic if at any state for any input there is at most one transition leading to the next state. Otherwise, M is nondeterministic.

Only strongly connected, minimal and deterministic FSMs are considered in this paper. In addition, it is assumed that UIO sequences for each state of an FSM are available and are derived from successor trees in advance. It is noted that only state transition functions in forms of $S \times I \longrightarrow S$ are considered; i.e., state transitions without any input in IoT are out of the scope of this paper.

3. Test Sequence Reduction

3.1. Key Properties of the Method

3.1.1. Overlapping by Invertibility

Definition 1 (invertibility-dependent UIO sequence). If a transition $(s, s', i/o)$ is invertible and β is a UIO sequence of s', then $(i/o)\cdot\beta$ is an invertibility-dependent UIO sequence of s.

The transition $(s_1, s_2, a/0)$ of M_1 in Figure 2 is invertible and $(a/1)(b/1)$ is a UIO sequence of s_2, then $(a/0)(a/1)(b/1)$ is an invertibility-dependent UIO sequence of s_1.

Definition 2 (invertibility-dependent overlapping sequence). Given a transition sequence $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ and a UIO sequence β of s_{j+1}, if every transition $(s_k, s_{k+1}, i_k/o_k)(1 \leqslant k \leqslant j-1)$ is verified by an invertibility-dependent UIO sequence $(i_{k+1}/o_{k+1}) \cdot (i_{k+2}/o_{k+2})\ldots(i_j/o_j) \cdot \beta$ when $(s_j, s_{j+1}, i_j/o_j)$ is verified by β, then $(s_1, s_2, i_1/o_1), (s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ is an invertibility-dependent overlapping sequence (IDOS).

There is a transition sequence $s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1$ of M_1 in Figure 2 and a UIO sequence $a/0$ of s_1. When the last transition $(s_3, s_1, a/1)$ of $s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1$ is verified by the UIO sequence $a/0$ of s_1, by working backward $(s_2, s_3, b/1)$ is verified by $(a/1)(a/0)$, $(s_2, s_2, a/1)$ is verified by $(b/1)(a/1)(a/0)$, and $(s_1, s_2, a/0)$ is verified by $(a/1)(b/1)(a/1)(a/0)$; i.e., all the other transitions except the last one are verified by invertibility-dependent UIO sequences. As a result, $s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1$ is an IDOS.

For $(i_{k+1}/o_{k+1}) \cdot (i_{k+2}/o_{k+2})\ldots(i_j/o_j) \cdot \beta$ $(1 \leqslant k \leqslant j-1)$ to be invertibility-dependent UIO sequences, indispensable requirements for transitions in $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ are put forward.

Theorem 3. *Given a transition sequence $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ and a UIO sequence β of s_{j+1}, $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ is an IDOS if and only if every transition $(s_k, s_{k+1}, i_k/o_k)$ $(2 \leqslant k \leqslant j)$ is invertible.*

Proof. The sufficiency of the condition is proved inductively by working backward. In the base case when $k = j$, $(s_{j-1}, s_j, i_{j-1}/o_{j-1})$ is sure to be verified by an invertibility-dependent UIO sequence $(i_j/o_j) \cdot \beta$ since $(s_j, s_{j+1}, i_j/o_j)$ is invertible. In the inductive step, assume that $(s_{k-1}, s_k, i_{k-1}/o_{k-1})$ $(3 \leqslant k \leqslant j-1)$ is verified by an invertibility-dependent UIO sequence γ, then $(s_{k-2}, s_{k-1}, i_{k-2}/o_{k-2})$ is definitely verified by an invertibility-dependent UIO sequence $(i_{k-1}/o_{k-1}) \cdot \gamma$ since $(s_{k-1}, s_k, i_{k-1}/o_{k-1})$ is invertible. In this way, every transition $(s_k, s_{k+1}, i_k/o_k)$ $(1 \leqslant k \leqslant j-1)$ is backward verified inductively by an invertibility-dependent UIO sequence $(i_{k+1}/o_{k+1}) \cdot (i_{k+2}/o_{k+2})\ldots(i_j/o_j) \cdot \beta$ when $(s_j, s_{j+1}, i_j/o_j)$ is verified by β. Thus, it is a sufficient condition for a transition sequence $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ to be an IDOS that every transition $(s_k, s_{k+1}, i_k/o_k)$ $(2 \leqslant k \leqslant j)$ is invertible.

Next, the necessity of the condition is proved by contradiction. As shown in Figure 3, suppose that $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ is an IDOS with a noninvertible transition $(s_k, s_{k+1}, i_k/o_k)$ $(2 \leqslant k \leqslant j)$; i.e., there is another transition $(s_{k'}, s_{k+1}, i_k/o_k)$ ending at s_{k+1} with the same input and output. Obviously, s_k cannot be identified by $(i_k/o_k) \cdot (i_{k+1}/o_{k+1})\ldots(i_j/o_j) \cdot \beta$; i.e., $(s_{k-1}, s_k, i_{k-1}/o_{k-1})$ cannot be verified by $(i_k/o_k) \cdot (i_{k+1}/o_{k+1})\ldots(i_j/o_j) \cdot \beta$. This contradicts with the assumption

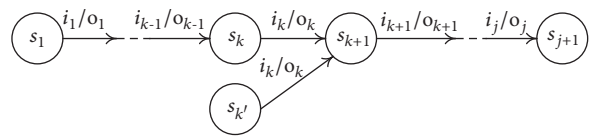


Figure 3: A fragment of an FSM.

that $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ is an IDOS. Accordingly, it is a necessary condition for a transition sequence $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$ to be an IDOS that every transition $(s_k, s_{k+1}, i_k/o_k)$ $(2 \leqslant k \leqslant j)$ is invertible. □

Definition 4 (set of IDOSs). Given a set T in which every sequence is an IDOS of an FSM M, if every transition of M is included in one and only one IDOS of T then T is a set of IDOSs of M.

Of all the IDOSs from a state, a maximum-length IDOS is the one contains no fewer transitions than any other ones. There is no doubt that the longer IDOSs are, the more overlapping can be achieved, and the shorter test sequences will be obtained. For maximum overlapping, this paper is only interested in maximum-length IDOSs.

The set $\{s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1\}$ is a set of IDOSs of M_1 in Figure 2. There is only one IDOS in the set since the only IDOS $s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1$ already covers all the transitions of M_1. Obviously, $s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1$ is also a maximum-length IDOS starting from s_1.

3.1.2. Multiple UIO Sequences. In the improved method, for any maximum-length IDOS $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$, the associated test subsequence is expressed as $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j) \cdot \beta$ where β is a UIO sequence of s_{j+1}. A minimum-length rural postman tour covering all the test subsequences is subsequently constructed by a rural symmetric augmentation and a test sequence is obtained from the tour. Accordingly, transitions for concatenation of test subsequences in the test sequence are derived from the transition replications during the rural symmetric augmentation. It is noted that different choice of UIO sequences may result in different rural symmetric augmentations; i.e., a minimum number of transition replications can be achieved by a sensible choice of UIO sequences during the rural symmetric augmentation. The minimum number of transition replications during the rural symmetric augmentation indicates the minimum number of transitions for concatenation of test subsequences in the test sequence, leading to a reduced test sequence. In other words, the result of using single UIO sequences can only in best-case scenarios obtain the same length of minimum-length rural postman tours as that of using multiple UIO sequences.

3.2. Design of the Method. It is known from Theorem 3 that noninvertible transitions restrict the generation of IDOSs.

From this point of view, FSMs can be partitioned into two subsets. One is FSMs with only invertible transitions and the other is FSMs with noninvertible transitions. For FSMs with only invertible transitions, if the FSMs are symmetric test sequences can be obtained directly. Otherwise, the FSMs should be augmented firstly. So FSMs with only invertible transitions can also be partitioned into two subsets. One is symmetric FSMs with only invertible transitions and the other is asymmetric FSMs with only invertible transitions. Generally, FSMs are classified into three categories: symmetric FSMs with only invertible transitions, asymmetric FSMs with only invertible transitions and FSMs with noninvertible transitions. The improved method is described in two steps for each type of FSMs and the detail varies for different types.

Step 1. Construct the set of maximum-length IDOSs.

Step 2. With the consideration of multiple UIO sequences, generate a minimum-length rural postman tour covering the set of maximum-length IDOSs and extract a test sequence from the tour.

3.2.1. Symmetric FSMs with Only Invertible Transitions

Step 1 (maximum-length IDOSs generation). There is an Euler tour in a directed graph if and only if the directed graph is strongly connected and symmetric. Under the assumption that all the FSMs are strongly connected, there must be Euler tours for symmetric FSMs with only invertible transitions. An Euler tour starting from and ending at the initial state is definitely an IDOS since there are only invertible transitions; furthermore, it is a maximum-length IDOS from the initial state since there is no other one containing more transitions. In other words, an Euler tour of a symmetric FSM with only invertible transitions is the only sequence in the set of maximum-length IDOSs.

Note that an Euler tour starting from and ending at the initial state may not be unique; i.e., there may be nonunique sets of maximum-length IDOSs. Nonetheless, all the sets of maximum-length IDOSs hold the following properties.

(i) There is only one maximum-length IDOS covering all the transitions of the FSM in every set of maximum-length IDOSs.

(ii) The start state of the maximum-length IDOS is the initial state of the FSM in every set of maximum-length IDOSs.

(iii) The end state of the maximum-length IDOS is the initial state of the FSM in every set of maximum-length IDOSs.

Step 2 (test sequence generation). For symmetric FSMs with only invertible transitions, test sequences are denoted by $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_1, i_j/o_j) \cdot \beta$ where $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_1, i_j/o_j)$ is an Euler tour staring from and ending at s_1 and β is a minimum-length UIO sequence of s_1. It can be inferred that test sequences from different sets of maximum-length IDOSs are always of the same length when there is more than one set of maximum-length IDOSs. As a result, a randomly generated set of maximum-length IDOSs will do for test sequence generation of symmetric FSMs with only invertible transitions.

M_1 in Figure 2 is a symmetric FSM with only invertible transitions and $s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1$ is an Euler tour starting from and ending at s_1. It is known from Step 1 that $\{s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1\}$ is a set of maximum-length IDOSs of M_1. A test sequence $s_1(a/0)s_2(a/1)s_2(b/1)s_3(a/1)s_1(a/0)s_2$ is resulted from Step 2.

3.2.2. Asymmetric FSMs with Only Invertible Transitions

Step 1 (maximum-length IDOSs generation). An FSM is asymmetric which refers to the fact that there are nodes whose out-degree does not equal in-degree. It is known that $\sum_{s\in S} d_{out}(s) = \sum_{s\in S} d_{in}(s) = |E|$ where $d_{out}(s)$ and $d_{in}(s)$ denote the out-degree and in-degree of a node, respectively [13]. The notation $|E|$ refers to the number of edges in a directed graph. It is concluded from $\sum_{s\in S} d_{out}(s) = \sum_{s\in S} d_{in}(s)$ that $\sum_{s\in S^-} d_{out>in}(s) = \sum_{s'\in S^+} d_{in>out}(s')$ where S^- and S^+ are the sets of nodes whose out-degree outnumbers in-degree and in-degree outnumbers out-degree, respectively. Correspondingly, $d_{out>in}(s)$ and $d_{in>out}(s')$ represent the amount of out-degree over in-degree for a node in S^- and the amount of in-degree over out-degree for a node in S^+, respectively. The relation $\sum_{s\in S} d_{out}(s) = \sum_{s\in S} d_{in}(s) = |E|$ implies that if a path passes through as many edges as possible exactly once then the path consumes as many out-degree and in-degree as possible. Every path through a node takes up one incoming edge as well as one outgoing edge, so if the out-degree outnumbers in-degree at a node, its outgoing edges cannot be traversed completely by paths passing through the node; i.e., some of its outgoing edges must instead be traversed by paths starting from the node. Thus, it is advisable to start the paths from nodes whose out-degree outnumbers in-degree if all the edges should be traversed exactly once with as few paths as possible. Moreover, it can be proved by contradiction that the paths from nodes whose out-degree outnumbers in-degree must end at nodes whose in-degree outnumbers out-degree otherwise the paths will continue to extend.

According to the above analysis, maximum-length IDOSs generation for asymmetric FSMs with only invertible transitions is performed as follows: start from a node $s \in S^-$ and go down along an outgoing edge until a node $s' \in S^+$ is reached and all the outgoing edges of s' have been traversed. Thus a maximum-length IDOS from s to s' is obtained. Add the maximum-length IDOS to the set of maximum-length IDOSs and delete the corresponding edges in the directed graph. Repeat the above process until all the nodes of the directed graph are isolated which implies that the set of maximum-length IDOSs is obtained.

Similarly, there may be nonunique sets of maximum-length IDOSs for asymmetric FSMs with only invertible transitions and all the sets of maximum-length IDOSs hold the following properties:

(i) For any set of maximum-length IDOSs, S^- is the set of start states for maximum-length IDOSs.

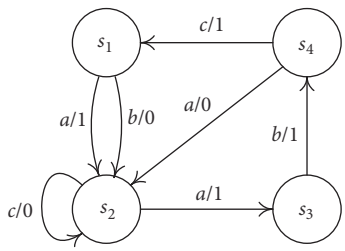


Figure 4: FSM M_2.

Table 3: UIO sequence of M_2.

No.	States	UIO(s)
Ex1	s_1	$b/0$ $(a/1)(c/0)$ $(a/1)(a/1)$
Ex2	s_2	$(a/1)(b/1)$ $c/0$
Ex3	s_3	$b/1$
Ex4	s_4	$a/0$ $c/1$

(ii) For any set of maximum-length IDOSs, S^+ is the set of end states for maximum-length IDOSs.

(iii) For any node $s \in S^-$, the number of maximum-length IDOSs starting from s is $d_{out>in}(s)$ in every set of maximum-length IDOSs.

(iv) For any node $s' \in S^+$, the number of maximum-length IDOSs ending at s' is $d_{in>out}(s')$ in every set of maximum-length IDOSs.

(v) The total number of maximum-length IDOSs is $\sum_{s\in S^-} d_{out>in}(s)$ which equals $\sum_{s'\in S^+} d_{in>out}(s')$ in each set of maximum-length IDOSs.

Step 2 (test sequence generation). With the set of maximum-length IDOSs, the same method as that of Bo Yang et al. is used to construct test sequences and the detail of the method is described in Algorithm 1. The core of the method is the rural symmetric augmentation over an FSM augmented by maximum-length IDOSs and the multiple UIO sequences for the end states of maximum-length IDOSs. The core of the symmetric augmentation is the states involved. According to the above properties, for any set of maximum-length IDOSs, there is no difference about the states involved in the symmetric augmentation such that a random set of maximum-length IDOSs will do for asymmetric FSMs with only invertible transitions when there is more than one set of maximum-length IDOSs.

M_2 in Figure 4 is an asymmetric FSM with only invertible transitions and its UIO sequences are shown in Table 3. The nonunique sets of maximum-length IDOSs for M_2 are listed as follows:

$\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(b/0)s_2, s_4(a/0)s_2\}$,
$\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(a/0)s_2, s_4(c/1)s_1(b/0)s_2\}$,
$\{s_1(b/0)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(a/1)s_2, s_4(a/0)s_2\}$,
$\{s_1(b/0)s_2(c/0)s_2(a/1)s_3(b/1)s_4(a/0)s_2, s_4(c/1)s_1(a/1)s_2\}$.

Clearly, all the sets of maximum-length IDOSs satisfy the properties in this section. The augmentation of M_2 using a random set of maximum-length IDOSs $\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(b/0)s_2, s_4(a/0)s_2\}$ is shown in Figure 5 and the associated test sequence of M_2 is $s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(b/0)s_2(a/1)s_3(b/1)s_4(a/0)s_2(c/0)s_2$.

3.2.3. FSMs with Noninvertible Transitions

Step 1 (maximum-length IDOSs generation). To address an FSM with noninvertible transitions in a similar way to the first two types of FSMs, noninvertible transitions are removed from the FSM and saved to a set. The remainder excluding noninvertible transitions is either an FSM or more than one connected component with only invertible transitions. And thus maximum-length IDOSs of the remainder excluding noninvertible transitions are generated in the same way as symmetric or asymmetric FSMs with only invertible transitions. Naturally, it comes to the same conclusion as the first two types of FSMs that a random set of maximum-length IDOSs will do when there are nonunique sets of maximum-length IDOSs. Next, a union of maximum-length IDOSs of the remainder excluding noninvertible transitions and all the noninvertible transitions is derived; moreover, whenever the start state of a maximum-length IDOS is the end state of a noninvertible transition, the maximum-length IDOS is concatenated with the noninvertible transition. If a maximum-length IDOS is an Euler tour then a node which is the end state of a noninvertible transition is preferred to be the start state of the tour. The set after all possible concatenations is a set of maximum-length IDOSs of the FSM with noninvertible transitions.

Given a union of maximum-length IDOSs of the remainder excluding noninvertible transitions as well as all the noninvertible transitions, there may be nonunique sets of maximum-length IDOSs for the FSM with noninvertible transitions because of different concatenation. The properties of nonunique sets of maximum-length IDOSs for FSMs with noninvertible transitions are described as follows:

(i) In any set of maximum-length IDOSs, for any state s which is the start state of a maximum-length IDOS, if there are j maximum-length IDOSs starting from s then there must be j maximum-length IDOSs starting from s in every other set of maximum-length IDOSs.

(ii) In any set of maximum-length IDOSs, for any state s which is the end state of a maximum-length IDOS, if there are k maximum-length IDOSs ending at s then there must be k maximum-length IDOSs ending at s in every other set of maximum-length IDOSs.

Step 2 (test sequence generation). For FSMs with noninvertible transitions, test sequence generation is in the same way as asymmetric FSMs with only invertible transitions, i.e., by means of rural symmetric augmentation over an FSM augmented by maximum-length IDOSs and the multiple UIO sequences for the end states of maximum-length IDOSs.

Require:
FSM M with n states $\{s_1, s_2, \ldots, s_n\}$ in which s_1 is the initial state;
Set of non-invertible transitions $P = \phi$;
Set of maximum-length IDOSs $Q = \phi$;
Set of end states for non-invertible transitions $S' = \phi$;
UIO sequences of M;
A minimum-length UIO sequence γ of s_1;
Ensure:
Reduced test sequence β of M;
(1) **if** M is an FSM with m non-invertible transitions **then**
(2) **for** i=1 to m **do**
(3) $P = P \cup t_i$ where t_i denotes a non-invertible transition;
(4) $M = M \setminus t_i$;
(5) $S' = S' \cup e_i$ where e_i denotes the end state of t_i;
(6) **end for**
(7) **for** each state whose out-degree > in-degree in M **do**
(8) Q=$Q \cup \alpha$ where α is a maximum-length IDOS of j transitions generated from the state;
(9) M=$M \setminus t_i$ $(1 \leqslant i \leqslant j)$;
(10) **end for**
(11) **for** each connected component with Euler tours **do**
(12) **if** States in S' can be found in an Euler tour t with k transitions **then**
(13) Choose a state in S' as the initial state of t;
(14) **end if**
(15) Q=$Q \cup t$;
(16) M=$M \setminus t_i$ $(1 \leqslant i \leqslant k)$;
(17) **end for**
(18) Q=$Q \cup P$;
(19) Concatenate non-invertible transitions and maximum-length IDOSs as long as the former's end state is the latter's start state;
(20) $M = M \cup V^*$ where V^* is a new state set;
(21) $M = M \cup T$ where T is a new transition set;
(22) $M = M \cup U$ where U is a new transition set;
(23) Construct a minimum-length rural postman tour over Q in the augmented M and extract a test sequence starting from s_1;
(24) Remove the transition sequence follows the UIO sequence of the maximum-length IDOS which is verified last in the minimum-length tour;
(25) **else**
(26) **if** M is a symmetric FSM with only invertible transitions **then**
(27) $Q = Q \cup t$ where t is an Euler tour starting from and ending at s_1;
(28) $\beta = t \cdot \gamma$;
(29) **else**
(30) execute lines (7) through (9);
(31) Execute lines (20) through (24);
(32) **end if**
(33) **end if**

Algorithm 1: Improved method of test sequence reduction.

The core of rural symmetric augmentation is still the states involved. It is known from the above properties that for any set of maximum-length IDOSs the states involved in the rural symmetric augmentation are the same such that a random concatenation will do.

Considering M_3 in Figure 6 with UIO sequences in Table 4, the following nonunique union of maximum-length IDOSs of the remainder excluding noninvertible transitions and noninvertible transitions confirm that a random union will do.

$\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(b/0)s_2, s_4(a/0)s_2, s_1(d/0)s_4, s_3(d/0)s_4\}$,

$\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(a/0)s_2, s_4(c/1)s_1(b/0)s_2, s_1(d/0)s_4, s_3(d/0)s_4\}$,

$\{s_1(b/0)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(a/1)s_2, s_4(a/0)s_2, s_1(d/0)s_4, s_3(d/0)s_4\}$,

$\{s_1(b/0)s_2(c/0)s_2(a/1)s_3(b/1)s_4(a/0)s_2, s_4(c/1)s_1(a/1)s_2, s_1(d/0)s_4, s_3(d/0)s_4\}$.

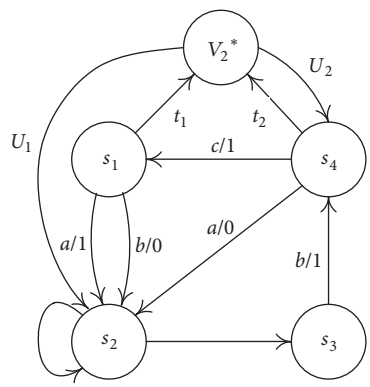


Figure 5: Augmentation of M_2.

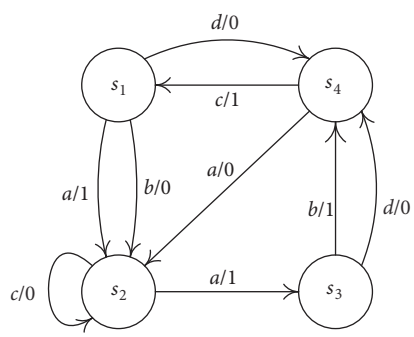


Figure 6: FSM M_3.

Table 4: UIO sequence of M_3.

No.	States	UIO(s)
Ex1	s_1	$b/0$ $(a/1)(c/0)$ $(a/1)(a/1)$
Ex2	s_2	$(a/1)(b/1)$ $c/0$
Ex3	s_3	$b/1$
Ex4	s_4	$a/0$ $c/1$

Take a random union of maximum-length IDOSs of the remainder excluding noninvertible transitions and noninvertible transitions $\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(b/0)s_2,\ s_4(a/0)s_2,\ s_1(d/0)s_4,\ s_3(d/0)s_4\}$; nonunique sets of maximum-length IDOSs from different concatenations $\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(b/0)s_2,\ s_3(d/0)s_4(a/0)s_2, s_1(d/0)s_4\}$ and $\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(b/0)s_2,\ s_1(d/0)s_4(a/0)s_2,\ s_3(d/0)s_4\}$ confirm that a random concatenation will do. The augmentation of M_3 using a randomly concatenated set of maximum-length IDOSs $\{s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)s_1(b/0)s_2,\ s_3(d/0)s_4(a/0)s_2,\ s_1(d/0)s_4\}$ is shown in Figure 7 and the resulting test sequence is $s_1(d/0)s_4(c/1)s_1(a/1)s_2(c/0)s_2(a/1)s_3(b/1)s_4(c/1)\ s_1(b/0)s_2(c/0)s_2(a/1)s_3(d/0)s_4(a/0)s_2(c/0)s_2$.

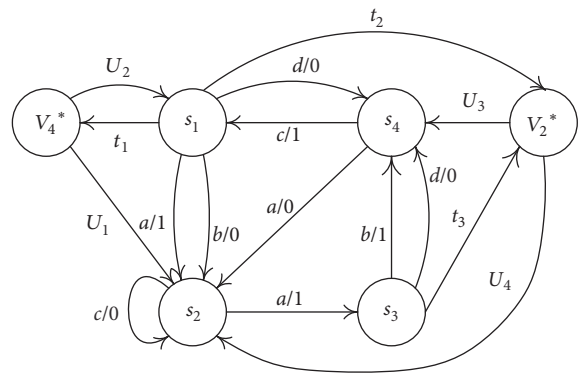


Figure 7: Augmentation of M_3.

Algorithm 1 describes the detail of the improved method for all types of FSMs. Note that self-loops are always given priority to traverse in the process of maximum-length IDOSs generation. When creating the new state set V^* in M, for the end state of every maximum-length IDOS, there is a state in V^*. When creating the new transition set T in M, for every maximum-length IDOS, there is a transition from the start state of the maximum-length IDOS to state v_i^* labeled with the corresponding label of the maximum-length IDOS. When creating a new transition set U in M, for every UIO sequence of the end state of every maximum-length IDOS, there is a transition from v_i^* to the end state of every UIO sequence labeled with the corresponding UIO sequence.

Theorem 5. *Given an FSM M, suppose that Q is a set of maximum-length IDOSs and β is a sequence over Q generated from Algorithm 1, then β is a reduced test sequence of M.*

Proof. The soundness of β is first proved; i.e., β is a test sequence of M. Then the effectiveness of β is assessed in terms of test generation and test execution cost, respectively. From a general standpoint, the notion of cost in the context of testing is complex and can be related to many factors. In our context, the effort required for generating test sequences is measured in terms of test sequence computational complexity. Test sequence execution cost is measured by the length of test sequences. Although these are clearly approximation methods, for practical reasons such methods have been commonly used in a number of testing studies [14–16]. □

(1) Soundness Analysis

(i) Check whether every transition defined in M is verified in the implementation

All the maximum-length IDOSs in Q are included in β since β is a sequence over Q. Algorithm 1 indicates that every maximum-length IDOS in β is followed by a UIO sequence that verifies the last transition of the sequence. According to Definition 2, i.e., for any maximum-length IDOS $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$, if $(s_j, s_{j+1}, i_j/o_j)$ is verified then every transition $(s_k, s_{k+1}, i_k/o_k)$ $(1 \leqslant k \leqslant j-1)$ is verified, it is inferred that transitions of all the

maximum-length IDOSs in Q are verified in β. According to Definition 4, i.e., every transition in M is included in one and only one IDOS of Q, it is concluded that every transition defined in M is verified in the implementation by β.

(ii) Check whether every state in M is defined in the implementation

M is supposed to be strongly connected such that for any state s of M there is at least one transition ending at s. It is proved that every transition defined in M is verified in β by identifying the end state of the transition; i.e., every state of M is checked in the implementation by β.

(2) Effectiveness Analysis

(i) Analysis of Computational Complexity. The test sequence β has the same computational complexity as those of Aho et al. and Hierons since all the three test sequence generation methods are based on the max flow/min cost problem and the networks used in every method are of the same order.

(ii) Analysis of Length Reduction. As mentioned above, transitions in a test sequence are of three kinds and the cost of a test sequence comes from the UIO sequences which have been appended for the purpose of transition verification and the transitions which have been appended for the purpose of concatenation of test subsequences to generate a minimum-length rural postman tour.

For symmetric FSMs with only invertible transitions, β is in the form of $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_1, i_j/o_j)\cdot\gamma$ where $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_1, i_j/o_j)$ is an Euler tour staring from and ending at s_1 and γ is a minimum-length UIO sequence of s_1; i.e., the cost of β only comes from one appended minimum-length UIO sequence of s_1.

For asymmetric FSMs with only invertible transitions, all the transitions are divided into a least number of maximum-length IDOSs; i.e., all the transitions are verified by a least number of appended UIO sequences such that the cost of β from the appended UIO sequences is minimal. For any maximum-length IDOS $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)$, there is a test subsequence $(s_1, s_2, i_1/o_1)(s_2, s_3, i_2/o_2)\ldots(s_j, s_{j+1}, i_j/o_j)\cdot\gamma$ where γ is a UIO sequence of s_{j+1}. β is obtained from a minimum-length rural postman tour covering all the test subsequences. The minimum-length rural postman tour is generated by the rural symmetric augmentation and thus the cost of β for concatenation comes from the replicated transitions during the rural symmetric augmentation. Benefiting from multiple UIO sequences, a minimum number of transition replications is reached by a sensible choice of UIO sequences during the rural symmetric augmentation; i.e., the cost of β from the transitions for concatenation is minimal.

For FSMs with noninvertible transitions, transitions excluding noninvertible ones are divided into a least number of IDOSs. According to Theorem 3, the acquired IDOSs concatenate with noninvertible transitions as much as possible such that a set of maximum-length IDOSs with a least number of maximum-length IDOSs is obtained; i.e., all the transitions of an FSM with noninvertible transitions are verified by a least number of appended UIO sequences such that the cost of β from the appended UIO sequences is minimal. Similarly, a minimum number of transition replications during the rural symmetric augmentation is reached by a sensible choice of UIO sequences; i.e., the cost of β from the transitions for concatenation is minimal.

In short, for all types of FSMs, the execution cost is reduced effectively without increasing the generation cost such that β is a reduced test sequence of M.

4. Case Study

We experiment with the aforementioned M_1, M_2, and M_3 which are random generation of different types of FSMs. M_4 and M_5 are also randomly generated FSMs; moreover, they are example FSMs used by Bo Yang et al. and Hierons, respectively. While the experimental results shed some light on how the improved method behaves with randomly generated FSMs, they bring no insight on the test sequence reduction of FSMs that are produced by software designers. For this reason, experiments on FSMs modeling INRES protocol [17], GUI for password modification in a property management system [18], page function of Gmail system [19] and the connection and release process of MAC layer in Zigbee protocol are carried out, and the FSM modeling page function of Gmail system is adjusted to satisfy the strong connectivity and UIO availability assumptions.

The FSMs used in the experiment are admittedly small. However, it is important to note that FSMs are mostly used to model the behavior of complex classes or class clusters, particularly complex control classes in protocols or reactive systems. They are rarely used to model entire systems which will result in large and unmanageable models for software engineers and testers. Completeness degree is a general factor to reveal the complexity of both large and small FSMs such that test sequence reduction of large FSMs can to some extent be learned through relatively small FSMs with the same distribution of completeness degree. Given an FSM with x inputs, y transitions and n states, completeness degree of M is denoted by $y/(x \times n)$; moreover, the higher the completeness degree is, the more complex the FSM is. In this section, completeness degrees of FSMs range from 0.24 to 1.

The associated data of the experiment is illustrated in Table 5, $|Input|$, $|State|$ and $|Transition|$ denote the number of inputs, states and transitions of every FSM. Completeness degree of every FSM is calculated and listed. $T_{FOTS+MUIO}$, T_{inv}, and $T_{inv+MUIO}$ are test sequences resulting from Bo Yang et al., Hierons, and the improved method, respectively. $|T_{FOTS+MUIO}|$, $|T_{inv}|$, and $|T_{inv+MUIO}|$ represent the length of the corresponding test sequences.

For symmetric FSMs with only invertible transitions, T_{inv} and $T_{inv+MUIO}$ are both Euler tours starting from the initial state followed by a minimum-length UIO sequence of the initial state. Thus, $|T_{inv}|$ and $|T_{inv+MUIO}|$ are always the same for symmetric FSMs with only invertible transitions. When the Euler tour conforms to the definition of fully overlapping transition sequences (FOTSs) [11], $|T_{FOTS+MUIO}|$ is the same as $|T_{inv}|$ and $|T_{inv+MUIO}|$; otherwise $|T_{FOTS+MUIO}|$ tends to be longer because of more appended UIO sequences as well as the possible extra transitions for concatenation.

Table 5: Experimental objects and associated data.

FSMs	Information of FSMs				Length of Test Sequences		
	$\lvert Input\rvert$	$\lvert State\rvert$	$\lvert Transition\rvert$	Completeness Degree	$\lvert T_{FOTS+MUIO}\rvert$	$\lvert T_{inv}\rvert$	$\lvert T_{inv+MUIO}\rvert$
M_1	2	3	4	0.67	8	5	5
M_2	3	4	7	0.58	10	12	10
M_3	4	4	9	0.56	15	15	13
M_4	3	5	11	0.73	17	17	16
M_5	2	5	10	1	33	21	20
INRES	5	4	16	0.8	13	12	12
GUI	7	8	25	0.45	74	31	31
Gmail	9	7	15	0.24	43	27	27
Zigbee	7	4	7	0.25	15	14	14

Table 6: Single sample K-S check.

	$\lvert T_{FOTS+MUIO}\rvert$	$\lvert T_{inv}\rvert$	$\lvert T_{inv+MUIO}\rvert$
N	9	9	9
Mean Value	25.333	17.111	16.444
Standard Deviation	21.535	8.054	8.263
$Kolmogorov-Smirnov\ \ Z$	.952	.517	.564
$Asympotic\ \ Significance\ \ (2-tailed)$	.325	.952	.908

Table 7: Paired samples statistics.

Pair	Mean	N	Standard Deviation	Standard Error Mean
$\lvert T_{FOTS+MUIO}\rvert$	25.333	9	21.535	7.178
$\lvert T_{inv+MUIO}\rvert$	16.444	9	8.263	2.754
$\lvert T_{inv}\rvert$	17.111	9	8.054	2.685
$\lvert T_{inv+MUIO}\rvert$	16.444	9	8.263	2.754

For asymmetric FSMs with only invertible transitions, $|T_{FOTS+MUIO}|$ and $|T_{inv+MUIO}|$ are the same if the maximum-length IDOSs comply with the definition of FOTS. Otherwise, $|T_{FOTS+MUIO}|$ tends to be longer because of more appended UIO sequences as well as the possible extra transitions for concatenation. T_{inv} is an Euler tour from a rural symmetric augmentation of an asymmetric FSM with only invertible transitions followed by a minimum-length UIO sequence of the initial state. The number of appended transitions in the rural symmetric augmentation is a deciding factor of $|T_{inv}|$.

For FSMs with noninvertible transitions, $|T_{FOTS+MUIO}|$ is longer than $|T_{inv+MUIO}|$ since every noninvertible transition is verified individually by an appended UIO sequence. $|T_{inv}|$ is the same as $|T_{inv+MUIO}|$ at best otherwise $|T_{inv}|$ tends to be longer than $|T_{inv+MUIO}|$.

As shown in Table 5, test sequences generated from different test methods conform to the above theoretical analysis. In this section, *two-sample t-test* is performed to compare test methods in terms of the length of the associated test sequences. *Single sample K-S check* in SPSS is used to verify the normality of the data studied since *t-test* is a parametric test and requires data to be normally distributed. Normality results are reported whenever samples deviate significantly from the normal distribution and the result of single sample *K-S* check in Table 6 shows that all the data in this experiment follow normal distribution. The paired samples statistics in Table 7 indicates that the average length of test sequences from the improved method is superior to those of the other two methods.

Vargha-Delaney effect size measure($\widehat{A}_{12}$) is also calculated to get more credible conclusions. When comparing two methods, $\widehat{A}_{12}$ measures the probability that one method would perform better than the other method. A value of 0.5 would mean that the two methods have equal probability of performing better than the other. The Vargha-Delaney effect size measure ($\widehat{A}_{12}$) from comparing $|T_{inv+MUIO}|$ to $|T_{FOTS+MUIO}|$ and $|T_{inv}|$ is shown in Table 8. The results show that $|T_{inv+MUIO}|$ is statistically 60.5% and 54.3% of the time significantly shorter than $|T_{FOTS+MUIO}|$ and $|T_{inv}|$, respectively.

5. Related Work

There are many works on IoT testing. Xiaoping Che et al. presented a logic-based approach to test the conformance and performance of XMPP protocol which is gaining momentum in IoT through real execution traces and formally specified

Table 8: Statistical results from $(\widehat{A}_{12})$.

	$\lvert T_{inv+MUIO}\rvert$ *versus*	$(\widehat{A}_{12})$
Ex1	$\lvert T_{FOTS+MUIO}\rvert$	0.395
Ex2	$\lvert T_{inv}\rvert$	0.457

properties [20]. Dimitrios Serpanos et al. introduced testing for security for IoT systems and especially fuzz testing, which is a successful technique to identify vulnerabilities in systems and network protocols [21]. Martin Tappler et al. presented a model-based approach to test IoT communication via active automata learning [22]. Combining Model-Based Testing (MBT) and a service-oriented solution, Abbas Ahmad et al. presented Model-Based Testing As A Service (MBTAAS) for testing data and IoT platforms [23]. Hiun Kim et al. introduced IoT testing as a Service-IoT-TaaS which is composed of remote distributed interoperability testing, scalable automated conformance testing, and semantics validation testing components adequate for testing IoT devices [24]. John Esquiagola et al. used the current version of their IoT platform to perform performance testing [25]. Daniel Kuemper et al. described how concepts for semantically described web services can be transferred into the IoT domain [26]. Philipp Rosenkranz et al. propose a testing framework which supports continuous integration techniques and allows for the integration of project contributors to volunteer hardware and software resources to the test system [27]. A connective and semantic similarity clustering algorithm (CSSCA) and a hierarchical combinatorial test model based on FSM are proposed by Kai Cui et al. [28].

Test sequence generation and reduction has long been an active research topic. Porto et al. used identification sets which are subsets of a characterizing set to identify states and obtained reduced test sequences [29]. Locating sequences are used to make sure that every element of a characterizing set is applied to the same state. Jourdan et al. generated shorter test sequences by means of reducing the number of locating sequences [30]. Baumgartner et al. proposed a mixed integer nonlinear programming (MINLP) model to formalize how the total cost of testing depends on the sequence and the parameters of the elementary test steps [31]. To provide an efficient formalization of the scheduling problem and avoid difficulties due to the evaluation of an objective function during the relaxation of the integer variables, the MINLP was formulated as a process network synthesis problem. Hierons et al. affirmed the importance of invertibility in test sequence reduction and considered three optimisation problems associated with invertible sequences [32]. Petrenko et al. addressed the problem of extending the checking experiment theory to cover a class of FSMs with symbolic extensions [33]. They also reported the results that further lift the theory of checking experiments for Mealy machines with symbolic inputs and symbolic outputs. Hierons et al. described an efficient parallel algorithm that uses many-core GPUs for automatically deriving UIOs from Finite State Machines [34]. The proposed algorithm uses the global scope of the GPU's global memory through coalesced memory access and minimizes the transfer between CPU and GPU memory. Song et al. introduced a practical conformance testing tool that generates high-coverage test input packets using a conformance test suite and symbolic execution. This approach can be viewed as the combination of conformance testing and symbolic execution [35]. Bokil et al. presented an automated black box test suite generation technique for reactive systems [36]. The technique is based on dynamic mining of specifications in form of an FSM from initial runs. The set of test cases thus produced contain several redundant test cases, many of which are eliminated by a simple greedy test suite reduction algorithm to give the final test suite.

6. Conclusions

Taking Zigbee protocol as an example, this paper introduces how FSM-based conformance testing works in wireless protocol conformance testing of IoT. An improved method in which both overlapping by invertibility and multiple UIO sequences are considered is proposed to achieve test sequence reduction for wireless protocol conformance testing of IoT. Based on invertibility, transitions of all types of FSMs are verified with as few appended UIO sequences as possible. Multiple UIO sequences contribute to generate a shorter test sequence by means of reducing transitions for concatenation of test subsequences in the test sequence. Moreover, test sequences can be further reduced by removing the transition sequence which follows the UIO sequence of the maximum-length IDOS that is verified last in the tour. Theory and experiment indicate that the execution cost is reduced effectively by the improved method under the premise of not increasing the generation cost. The improved method is also applicable to traditional protocol conformance testing as well as reactive systems. Numerous experimental data and practical examples about wireless protocols of IoT will be gathered in the future work to analyze the effectiveness of the method.

Data Availability

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

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 National Natural Science Foundation of China (Grant no. 61572306 and Grant no. 61502294), the IIOT Innovation and Development Special Foundation of Shanghai (Grant no. 2017-GYHLW-01037), and the CERNET Innovation Project (NGII20170513 and NGII20170206).

References

[1] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[2] L. Qi, X. Xu, X. Zhang et al., "Structural balance theory-based e-commerce recommendation over big rating data," *IEEE Transactions on Big Data*, vol. 4, no. 3, pp. 301–312, 2018.

[3] K. Zaniewski and W. Pedrycz, "A hybrid optimization approach to conformance testing of finite automata," *Applied Soft Computing*, vol. 23, pp. 91–103, 2014.

[4] S. J. S., K. H., and P. S., "Enhancing conformance testing using symbolic execution for network protocols," *IEEE Transactions on Reliability*, vol. 64, no. 3, pp. 1024–1037, 2015.

[5] P. Liu, H.-K. Miao, H.-W. Zeng, and Y. Liu, "FSM-based testing: Theory, method and evaluation," *Jisuanji Xuebao/Chinese Journal of Computers*, vol. 34, no. 6, pp. 965–984, 2011.

[6] R. Dorofeeva, K. El-Fakih, S. Maag, A. R. Cavalli, and N. Yevtushenko, "Experimental evaluation of FSM-based testing methods," in *Proceedings of the 3rd IEEE International Conference on Software Engineering and Formal Methods (SEFM '05)*, pp. 23–32, IEEE, Koblenz, Germany, September 2005.

[7] X. Zhang, M. Yang, J. Zhang, H. Shi, and W. Zhang, "A study on the extended unique input/output sequence," *Information Sciences*, vol. 203, pp. 44–58, 2012.

[8] I. Ahmad, F. M. Ali, and A. S. Das, "Synthesis of finite state machines for improved state verification," *Computers & Electrical Engineering*, vol. 32, no. 5, pp. 349–363, 2006.

[9] J. Shujuan, J. Xiaolin, W. Xingya, L. Haiyang, Z. Yamei, and L. Yingqi, "Measuring the importance of classes using uio sequence," *Chinese Journal of Electronics*, vol. 43, no. 10, pp. 2062–2068, 2015 (Chinese).

[10] É. F. D. Souza, V. A. D. Santiago Júnior, and N. L. Vijaykumar, "H-Switch Cover: a new test criterion to generate test case from finite state machines," *Software Quality Journal*, vol. 25, no. 2, pp. 373–405, 2017.

[11] B. Yang and H. Ural, "Protocol conformance test generation using multiple uio sequences with overlapping," *ACM SIGCOMM Computer Communication Review*, vol. 20, no. 4, pp. 118–125, 1990.

[12] R. M. Hierons, "Extending test sequence overlap by invertibility," *The Computer Journal*, vol. 39, no. 4, pp. 325–330, 1996.

[13] J. Bang-Jensen and G. Z. Gutin, *Digraphs: Theory, Algorithms and Applications*, Springer, New York, NY, USA, 2002.

[14] S. Mouchawrab, L. C. Briand, Y. Labiche, and M. Di Penta, "Assessing, comparing, and combining state machine-based testing and structural testing: a series of experiments," *IEEE Transactions on Software Engineering*, vol. 37, no. 2, pp. 161–187, 2011.

[15] K. El-Fakih, A. Simao, N. Jadoon, and J. C. Maldonado, "An assessment of extended finite state machine test selection criteria," *The Journal of Systems and Software*, vol. 123, pp. 106–118, 2017.

[16] A. T. Endo and A. Simao, "Evaluating test suite characteristics, cost, and effectiveness of FSM-based testing methods," *Information and Software Technology*, vol. 55, no. 6, pp. 1045–1062, 2013.

[17] J. F. Cutigi, A. Simao, and S. R. Souza, "Reducing fsm-based test suites with guaranteed fault coverage," *The Computer Journal*, vol. 59, no. 8, pp. 1129–1143, 2016.

[18] W.-W. Lin and H.-W. Zeng, "A chain algorithm for conformance testing based on uio sequences," in *Proceedings of the International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pp. 1–6, IEEE, 2015.

[19] P. Liu, H.-K. Miao, H.-W. Zeng, and J. Mei, "DFSM-based minimum test cost transition coverage criterion," *Ruan Jian Xue Bao/Journal of Software*, vol. 22, no. 7, pp. 1457–1474, 2011.

[20] X.-P. Che and S. Maag, "A passive testing approach for protocols in internet of things," in *Proceedings of the 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, pp. 678–684, IEEE, 2013.

[21] D. Serpanos and M. Wolf, "Security testing iot systems," *Internet-of-Things (IoT) Systems*, pp. 77–89, 2018.

[22] M. Tappler, B. K. Aichernig, and R. Bloem, "Model-Based Testing IoT Communication via Active Automata Learning," in *Proceedings of the 10th IEEE International Conference on Software Testing, Verification and Validation, ICST 2017*, pp. 276–287, IEEE, March 2017.

[23] A. Ahmad, F. Bouquet, E. Fourneret, F. L. Gall1, and B. Legeard, "Model-based testing as a service for iot platforms," in *Proceedings of the International Symposium on Leveraging Applications of Formal Methods*, pp. 727–742, Springer, 2016.

[24] H. Kim, A. Ahmad, J. Hwang et al., "IoT-TaaS: Towards a Prospective IoT Testing Framework," *IEEE Access*, vol. 6, pp. 15480–15493, 2018.

[25] J. Esquiagola, L. Costa, P. Calcina, G. Fedrecheski, and M. Zuffo, "Performance testing of an internet of things platform," in *Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security, IoTBDS 2017*, pp. 309–314, Portugal, April 2017.

[26] K. Daniel, E. Reetz, and R. Tönjes, "Test derivation for semantically described iot services," in *Proceedings of the 2013 Future Network & Mobile Summit*, pp. 1–10, 2013.

[27] P. Rosenkranz, M. Wählisch, E. Baccelli, and L. Ortmann, "A Distributed Test System Architecture for Open-source IoT Software," in *Proceedings of the 2015 Workshop on IoT challenges in Mobile and Industrial Systems*, pp. 43–48, ACM, Florence, Italy, May 2015.

[28] K. Cui, K.-j. Zhou, T. Qiu, M.-c. Li, and L.-m. Yan, "A hierarchical combinatorial testing method for smart phone software in wearable iot systems," *Computers and Electrical Engineering*, vol. 61, pp. 250–265, 2017.

[29] F. R. Porto, A. T. Endo, and A. Simao, "Generation of checking sequences using identification sets," in *Proceedings of the International Conference on Formal Engineering Methods*, pp. 115–130, Springer, 2013.

[30] G.-V. Jourdan, H. Ural, and H. Yenigun, "Reduced checking sequences using unreliable reset," *Information Processing Letters*, vol. 115, no. 5, pp. 532–535, 2015.

[31] J. Baumgartner, Z. Sle, B. Bertk, and J. Abonyi, "Test-sequence optimisation by survival analysis," *Central European Journal of Operations Research*, pp. 1–19, 2018.

[32] R. M. Hierons, M. R. Mousavi, M. K. Thomsen, and U. C. Trker, "Hardness of deriving invertible sequences from finite state machines," in *Proceedings of the International Conference on Current Trends in Theory and Practice of Informatics*, pp. 147–160, Springer, 2017.

[33] A. Petrenko, "Toward testing from finite state machines with symbolic inputs and outputs," *Software and Systems Modeling*, pp. 1–11, 2017.

[34] R. M. Hierons and U. C. Turker, "Parallel Algorithms for Testing Finite State Machines:Generating UIO Sequences," *IEEE Transactions on Software Engineering*, vol. 42, no. 11, pp. 1077–1091, 2016.

[35] J. Song, H. Kim, and S. Park, "Enhancing Conformance Testing Using Symbolic Execution for Network Protocols," *IEEE Transactions on Reliability*, vol. 64, no. 3, pp. 1024–1037, 2015.

[36] P. Bokil, P. Krishnan, and R. Venkatesh, "Achieving effective test suites for reactive systems using specification mining and test suite reduction techniques," *ACM SIGSOFT Software Engineering Notes*, vol. 40, no. 1, pp. 1–8, 2015.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 5198685, 19 pages
https://doi.org/10.1155/2018/5198685

Research Article

Adaptive DDoS Attack Detection Method Based on Multiple-Kernel Learning

Jieren Cheng,[1,2,3] Chen Zhang,[1] Xiangyan Tang,[1] Victor S. Sheng,[4] Zhe Dong,[1] and Junqi Li[1]

[1] *College of Information Science & Technology, Hainan University, Haikou 570228, China*
[2] *State Key Laboratory of Marine Resource Utilization in South China Sea, Haikou 570228, China*
[3] *Key Laboratory of Internet Information Retrieval of Hainan Province, Hainan University, Haikou 570228, China*
[4] *Department of Computer Science, University of Central Arkansas, Conway, AR 72035, USA*

Correspondence should be addressed to Chen Zhang; 314848554@qq.com

Received 11 July 2018; Accepted 19 September 2018; Published 16 October 2018

Guest Editor: Lianyong Qi

Distributed denial of service (DDoS) attacks has caused huge economic losses to society. They have become one of the main threats to Internet security. Most of the current detection methods based on a single feature and fixed model parameters cannot effectively detect early DDoS attacks in cloud and big data environment. In this paper, an adaptive DDoS attack detection method (ADADM) based on multiple-kernel learning (MKL) is proposed. Based on the burstiness of DDoS attack flow, the distribution of addresses, and the interactivity of communication, we define five features to describe the network flow characteristic. Based on the ensemble learning framework, the weight of each dimension is adaptively adjusted by increasing the interclass mean with a gradient ascent and reducing the intraclass variance with a gradient descent, and the classifier is established to identify an early DDoS attack by training simple multiple-kernel learning (SMKL) models with two characteristics including interclass mean squared difference growth (M-SMKL) and intraclass variance descent (S-SMKL). The sliding window mechanism is used to coordinate the S-SMKL and M-SMKL to detect the early DDoS attack. The experimental results indicate that this method can detect DDoS attacks early and accurately.

1. Introduction

In recent years, the security of computer networks, chips, virtual networks, and mobile devices has been of wide concern [1–3]. As an important platform for information exchange, computer network security has attracted much attention. In the security of computer network, distributed denial of service (DDoS) attack is yet to be settled in a long time. DDoS is a traditional network attack method. It controls a large number of zombie machines sending a large number of invalid network request packets to a target host. It consumes and meaninglessly occupies the resources of the server, causing normal users to be unable to use the normal services provided by the target host [4]. Although the DDoS attack mode is simpler, its destruction power to the network is far more than other network attacks. Moreover, this traditional attack method in recent years can still cause great damage to the Internet, and the frequency of launch, loss caused, complexity of DDoS, diversity of DDoS, and difficulty of defense have increased more than before [5]. In June 2016, an ordinary U.S. jewelry online sales website was flooded with 35,000 HTTP requests (spam requests) per second, making the site unable to provide normal services. In October, DynDNS, which provides dynamic DNS services in the United States, was subject to large-scale DDoS attacks, resulting in access problems for multiple websites using DynDNS services, including GitHub, Twitter, Airbnb, Reddit, Freshbooks, Heroku, SoundCloud, Spotify, and Shopify. Twitter has even appeared in nearly 24 hours with a zero-visit situation. The reason why DDoS attacks have such a great destructive power is that DDoS uses a large number of zombie machines to launch attacks on a certain target. Each zombie machine has powerful computing capability. Through the massive distributed processing capabilities of

zombie machines, it is easy for a server to no longer have the ability to provide services to normal users [6]. On the other hand, DDoS attacks are easy to implement. Unlike other network attacks, DDoS attacks require only a large number of zombie machines and a small amount of network security knowledge to launch an effective attack. This easy-to-grasp network attack method makes the DDoS attack more powerful.

At present, under the traditional network environment, methods for defense against DDoS attacks mainly include attack detection and attack response [7]. DDoS attack detection is based on attack signatures, congestion patterns, protocols, and source addresses as an important basis for detecting attacks, thereby establishing an effective detection mechanism. The detection model can be roughly divided into two categories: misuse-based detection and anomaly-based detection. Misuse-based detection is a technique based on feature-matching algorithms. It matches the collected and extracted user behavior features with the known feature database of DDoS attacks to identify whether an attack has occurred. Anomaly-based detection is adopted by monitoring systems. By establishing the target system and the user's normal behavior model, the monitoring systems can determine whether the states of the system and the user's activities deviate from the normal profile and can judge whether there is an attack. The attack response is to properly filter or limit the network traffic after the DDoS attack is initiated. The attack traffic to the attack target host is reduced as much as possible to mitigate the influence of the denial of a service attack.

With the rise of cloud computing technologies and software-defined networking (SDN) concepts, DDoS attack detection based on cloud computing environments and software-defined networks has received widespread attention [8, 9]. As a new computing model, cloud computing has powerful distributed computing capabilities, massive storage capabilities, and diverse service capabilities [10, 11]. It has become an important means of solving big data problems [12]. Therefore, establishing a cloud platform system is a necessary measure to effectively ensure cloud computing's reliability, stability, and security [13–15].

In recent years, machine learning has been applied to the field of security [17]. The method of constructing an attack detection model using machine learning has been widely used [18, 19]. The machine-learning method plays an important role in the traditional network environment, the cloud environment, and software-defined network architecture. The reason is that the machine-learning method can deeply mine the important information hidden behind the data and combine prior knowledge to discriminate and predict new data [20]. Therefore, compared with traditional detection methods, machine-learning methods can exhibit better detection accuracy [21–25]. In the above analysis of defense measures, it is known that the traditional network environment, cloud environment, and software-defined network architecture all involve attack detection for the defense mechanism of DDoS. Therefore, studying the use of machine-learning methods to identify DDoS attacks is of great significance. However, the data generated by the DDoS attack is often burst and diverse, and the background traffic size also has a greater impact on the detection model, thereby reducing the model's detection accuracy.

To solve the above problems, we propose a multiple-kernel learning DDoS attack detection method. The method uses the algorithm to extract five features and combines two multiple-kernel learning models with the adaptive feature weights to recognize attack flows and normal flows. For further improving the accuracy of DDoS attack detection, a sliding window mechanism is employed to coordinate two multiple-kernel learning models treating the detection results. Experiments show that our method can better distinguish DDoS attack flow from normal flow and can detect DDoS attacks earlier.

2. Related Work

DDoS attacks can cause tremendous damage to a network and often subject the attacked party to great economic losses. This is one of the main ways that hackers initiate cyberattacks.

To reduce the damage of DDoS attacks, researchers have proposed a large number of attack detection methods in recent years. According to the application scenario, these methods can be divided into three categories: the detection method in the conventional network environment, the detection method in the cloud environment, and the detection method in the software-defined network (SDN) environment.

(1) The conventional network environment refers to the Internet environment generally established on the Internet based on an open system interconnect reference model (OSI). In this regard, Saied et al. proposed a method for detecting known and unknown DDoS attacks using artificial neural networks [26]. Bhuyan et al. proposed an empirical evaluation method for the measurement of low-rate and high-rate DDoS attack detection information [27]. Tan et al. proposed a DDoS attack detection method based on multivariate correlation analysis [28]. Yu et al. proposed a DDoS attack detection method based on the traffic correlation coefficient [29]. Wang et al. conducted an in-depth analysis of the characteristics of DDoS botnets [30]. Kumar and others used the Jpcap API to monitor and analyze DDoS attacks [31]. Khundrakpam et al. proposed an application-layer DDoS attack detection method combining entropy and an artificial neural network [32].

(2) The cloud environment refers to the network service platform with cloud computing as the core technology. In this regard, Karnwal et al. proposed a defense method for XML DDoS and HTTP DDoS attacks under cloud computing platforms [33]; Sahi et al. proposed the check and defense method for TCP-flood DDoS attacks in the cloud environment [34]. Rukavitsyn et al. proposed a self-learning DDoS attack detection method in the cloud environment [35].

(3) Software-defined network refers to a new network architecture that adopts OpenFlow as the communication protocol and specifies the router as well as switch data exchange rules through the controller [36]. In this regard, Ashraf used machine-learning detection software to define

DDoS attacks under the network [37]. Mihai–Gabriel proposed an intelligent elastic risk assessment method based on the neural network and risk theory in the SDN environment [38]. Yan et al. proposed an effective controller scheduling method to reduce DDoS attacks in software-defined networks [39]. Chin et al. proposed a DDoS flood attack method for selective detection of packets under SDN [40]. Dayal et al. analyzed the behavioral characteristics of DDoS attacks under SDN [41]. Ye et al. proposed a method of using SVM to detect DDoS attacks under the SDN environment [42]. Except the above detection methods used to ensure the security of the system, some efficient cryptography techniques can be applied to achieve privacy of the system [43–46].

In summary, the core issue of DDoS attack detection research is the construction of feature extraction and classification models. The attack detection methods in the above three environments can effectively detect DDoS attacks corresponding to the environment. However, in the detection of early DDoS attack, these defense methods do not have a good detection effect. In addition, most of these methods use a single feature and do not consider the impact of multidimensional features on the classifier. Therefore, an adaptive DDoS attack detection method is proposed in this paper. Firstly, we design the algorithms to extract five features. Secondly, through an ensemble learning framework, the five features are used to train two multikernel learning models and obtain the adaptive feature weights with gradient method. Finally, the sliding window mechanism is used to coordinate the two models to improve the detection accuracy.

3. DDoS Attack Feature Extraction

3.1. Analysis of DDoS Attack Behavior. In the cloud environment, the botnets of DDoS attacks have distributed characteristics. Each zombie machine has the ability to independently calculate, send, and process data packets, and the source IP address of the packets can also be forged. The advantage of these DDoS attacks makes defense more difficult. However, under the background of time series, the characteristics of data packets generated by DDoS attacks are still quite different from those of normal users. The difference is reflected in the following three aspects.

(1) Asymmetry. DDoS attack is often caused by multiple zombie hosts sending a large number of packets to a host without the host's response. These useless packets quickly consume the host's service resources so that the host can no longer provide services to other users. With this feature, the DDoS attack behavior is such that there are a large amount of packets sent to the host from the zombie hosts, and there are no or a small amount of packets sent to the zombie hosts from the host. The IP data packet often presents a situation in which multiple-source IP addresses point to the same or several destination IP addresses, which is expressed as the asymmetry of the source IP as well as the destination IP in sending and receiving.

(2) Interactivity. It is assumed that there are A (zombie host) and B (attacked host). When an attack occurs, there are two main communication ways as follows: (1) A sends packets to B (denoted as A⟶B) and (2) A and B send packets to each other (denoted as A⇄B). And the packet amount sent with the way (A⟶B) is much more than those sent with the way (A⇄B). Therefore, the interactivity of DDoS attack flow has different states in communication direction and amount compared with normal flow.

(3) Distribution. According to the characteristics of DDoS attack, when an attack occurs, the number of the hosts that launch the attack is much larger than that of the attacked hosts. And the number of the source IP address is much larger than that of the destination IP address, so that the source address and the destination address have different distribution characteristics. In addition, because DDoS attacks generate useless requests, so compared to normal flows, the host ports accessed by the attack requests are more dispersed. Therefore, the distribution of the ports is different in normal flows and attack flows.

Due to the limited ability of a single feature to express data, it cannot fully reflect the characteristics of the DDoS attack. Therefore, to effectively express the characteristics of the DDoS attack, this paper selects five feature extraction methods based on the above characteristics as follows. The address correlation degree (ACD) combines the traffic burstiness, flow asymmetry, and source IP address distribution of DDoS attack; the IP flow features value (FFV) exploits the asymmetry of attack flows and the distribution of source IP addresses; the IP flow's interaction behavior feature (IBF) uses the different interactivity between normal flows and attack flows on the network; the IP flow multifeature fusion (MFF) exploits the different behavioral characteristics of normal flows as well as DDoS attack flows and integrates the multiple characteristics of DDoS attack flows; the IP flow address half interaction anomaly degree (HIAD) focuses on the characteristics of the aggregated attack flows that are mixture of a large number of normal background flows. In order to make the feature richer in representation, we refer to several articles and combine the five feature extraction algorithms, besides removing the less impactful parameters to form a multidimensional feature for DDoS attack detection [45–51].

3.2. DDoS Attack Feature Extraction. In the cloud environment, assume that network flow F is as follows: $\langle (t_1, s_1, d_1, p_1), (t_2, s_2, d_2, p_2), \ldots, (t_n, s_n, d_n, p_n) \rangle$ in a certain unit of time, where t_i, s_i, d_i, and p_i denote the time, source IP address, destination IP address, and the port of the $i(i = 1, 2, \ldots, n)$-th data packet, respectively. All data packets which contain source IP address A_i and destination IP address A_j are denoted as class $SD(A_i, A_j)$. All data packets with source IP address A_i are denoted as class $IPS(A_i)$. All data packets with destination IP address A_j are denoted as class $IPD(A_j)$. The packets with source IP address A_i which exist in the class $IPS(A_i)$ and class $IPD(A_i)$ are denoted as $IF(A_i)$. The packets with source IP address A_i which exist in class $IPS(A_i)$ and do not exist class $IPD(A_i)$ are denoted as $SH(A_i)$. The number of the different ports in $SH(A_i)$ is denoted as $Port(SH(A_i))$. The packets with the destination

IP address A_i which do not exist in class $IPS(A_i)$ and exist in class $IPD(A_i)$ are denoted as $DH(A_i)$. The number of the different ports in $DH(A_i)$ is denoted as $Port(DH(A_i))$.

Definition 1. If there are different destination IP addresses A_j and A_k, making classes $SD(A_i, A_j)$ and $SD(A_i, A_k)$ both non-null, then delete the class where all source IP address A_i packets reside.

Assume that the last remaining classes are denoted as $ACS_1, ACS_2, ..., ACS_m$ and are statistically calculated to gain the ACD. The detailed formulation is as follows.

$$ACD_F = \sum_{i=1}^{m} W(ACS_i) \tag{1}$$

In this part $W(ACS_i) = \theta_1 Port(ACS_i) + (1 - \theta_1)Packet(ACS_i)(0 < \theta_1 < 1)$, where $Port(ACS_i)$ is the number of different ports in class ACS_i, $Packet(ACS_i)$ is the number of data packets in class ACS_i, and θ_1 is the weighted value.

Definition 2. If all the packets whose destination IP address is A_j form the unique class $SD(A_i, A_j)$, delete the class where the packet with the destination IP address is A_j.

Assume that the last remaining classes are denoted as $SDS_1, SDS_2, \ldots, SDS_l$, all packets in these remaining classes with the destination IP address A_j are denoted as $SDD(A_j)$, and all the classes are denoted as $SDD_1, SDD_2, \ldots, SDD_m$. The FFV is defined as follows:

$$FFV_F = \left(\sum_{i=1}^{m} CIP(SDD_i) - m\right) \tag{2}$$

$CIP(SDD_i)$ in formula (2) is presented as follows:

$$\begin{aligned} CIP(SDD_i) &= Num(SDD_i) \\ &+ \theta_2 \sum_{j=1}^{Num(SDD_i)} OA\left(Pack\left(A_j\right)\right) \\ &+ (1-\theta_2)\left(OB\left(Port\left(SDD_i\right)\right) - 1\right). \end{aligned} \tag{3}$$

In this equation, $0 \le \theta_2 \le 1$, $Num(SDD_i)$ is the number of different source IP addresses in SDD_i;

$$\begin{aligned} OA\left(Pack\left(A_j\right)\right) = \Bigg\{ & Pack\left(A_j\right), \ \frac{Pack\left(A_j\right)}{\Delta t} \\ & > \theta_3; \ 0, \ \frac{Pack\left(A_j\right)}{\Delta t} \le \theta_3 \Bigg\}, \end{aligned} \tag{4}$$

$Pack(A_j)$ is the number of source IP addresses A_j in SDD_i, and θ_3 is the threshold of the number of packets:

$$\begin{aligned} OB\left(Port\left(SDD_i\right)\right) = \Bigg\{ & Port\left(SDD_i\right), \ \frac{Port\left(SDD_i\right)}{\Delta t} \\ & > \theta_4; \ 0, \ \frac{Port\left(SDD_i\right)}{\Delta t} \le \theta_4 \Bigg\}, \end{aligned} \tag{5}$$

$Port(SDD_i)$ is the number of different destination ports in SDD_i, θ_4 is the threshold of the number of ports, and Δt is the sampling time.

Definition 3. Assume that the IF flow is $IF_1, IF_2, \ldots, IF_M$, the SH class is denoted as $SH_1, SH_2, \ldots SH_S$, and the DH class is denoted as $DH_1, DH_2, \ldots, DH_M$. Then, define IBF as follows:

$$\begin{aligned} IBF = \frac{1}{M+1} \Bigg(& |S - D| + \sum_{i=1}^{S} over\left(Port\left(SH_i\right)\right) \\ & + \sum_{i=1}^{D} over\left(Port\left(DH_i\right)\right) \Bigg) \end{aligned} \tag{6}$$

$over(x) = \{x, x/\Delta t > \theta_5; 0, x/\Delta t \le \theta_5\}$, where θ_5 is the threshold of the amount of port. M in formula (6) is the number of IF flows within Δt, and $|S - D|$ is the absolute value of the difference value between the number of source IP addresses and the number of destination IP addresses for all SH and DH flows in Δt.

Definition 4. Assume that the resulting SD classes are $SD_1, SD_2, \cdots SD_l$ and IF classes are $IF_1, IF_2, \cdots IF_M$. The number of packets of source IP address A_i in class IF_i is denoted as Sn_i, where $i = 1, 2, ..., M$; the number of packets of all interworking flow classes is denoted as SN; and the source semi-interactive flow class is denoted as $SH_1, SH_2, \cdots SH_S$. The number of different ports in class SH_i is denoted as $Port(SH_i)$, where $i = 1, 2, ..., S$; the destination semi-interactive class is denoted as $DH_1, DH_2, \cdots DH_D$; and the number of different ports in class DH_i is denoted as $Port(DH_i)$, where $i = 1, 2, ..., D$.

The weighted value of all packets in SH class is defined as follows:

$$Weight_{SH} = \sum_{i=1}^{s} oversh\left(Packet\left(SH_i\right)\right) \tag{7}$$

The weighted value of all packets in SD classes is defined as follows:

$$Weight_{SD} = \sum_{i=1}^{L} oversd\left(Packet\left(SD_i\right)\right) \tag{8}$$

The weighted value of the number of packets of network flow F in unit time T is as follows:

$$\begin{aligned} Weight_{packet} = & \ flag\left(Weight_{SD}\right) Weight_{SD} \\ & + Weight_{SD} \end{aligned} \tag{9}$$

In these equations,

$$\begin{aligned} oversh(x) &= \left\{x, \ \frac{x}{\triangle t} > \theta_6; \ 0, \ \frac{x}{\triangle t} \le \theta_6\right\}, \\ oversd(x) &= \left\{x, \ \frac{x}{\triangle t} > \theta_7; \ 0, \ \frac{x}{\triangle t} \le \theta_7\right\}, \\ flag(x) &= \{0, \ x > 0; \ 1, \ x = 0\}, \end{aligned} \tag{10}$$

$\triangle t$ is sampling time, and θ_6 and θ_7 are SH-type packet number abnormality thresholds; $Packet(SD_i)$ is the number of packets in SD_i, $I = 1, 2, ..., n$. The weighted value of the number of different ports in the SH and DH classes is as follows:

$$Weight_{port} = \sum_{i=1}^{S} overp\left(Port\left(SH_i\right)\right) + \sum_{j=1}^{D} overp\left(Port\left(DH_j\right)\right) \tag{11}$$

where $overp(x) = \{x, x/\triangle t > \theta_8; 0, x/\triangle t \leq \theta_8\}$, $\triangle t$ is sampling time, and θ_8 is the SH-type port number abnormality threshold.

In this part we define the MFF as follows:

$$MFF_F = \frac{\mathrm{S} + Weight_{port} + Weight_{packet}}{M + 1} \tag{12}$$

where $f(x) = \{x, x \geq 1; 1, x \leq 1\}$.

Definition 5. The number of SH flows with different source IP addresses and the same destination IP address A_i is denoted as hn_i. The SH class with the same destination IP address A_i flow is denoted as $HSD(hn_i, A_i)$, where $i = 1, 2, ..., n$.

Assume that all HSD classes are $HSD_1, HSD_2, \cdots HSD_k$, and the number of different destination ports in the class HSD_i is expressed as $Port(HSD_i)$, where $i = 1, 2, ..., k$.

The HIAD is defined as follows:

$$HIAD_F = \left(\sum_{i=1}^{k}\left(hn_i + weight\left(Port\left(HSD_I\right)\right)\right)\right) \tag{13}$$

In (13), $weight(x) = \{x, x/\triangle t > \theta_9; 0, x/\triangle t \leq \theta_9\}$, $\triangle$t is sampling time, and θ_9 is the threshold for different destination ports.

4. The DDoS Attack Detection Model

The establishment of an attack detection model is an important part of the whole detection process. Based on the behavior of DDoS attack, we extract *ACD*, *IBF*, *MFF*, *HIAD*, and *FFV* features to express the inherent rules of attack flows. The disadvantages of the current DDoS attack detection models are summarized as follows: (1) some models highly depend on the selection of kernel function; (2) some models require data with highly stable value; (3) some models can only fit linear rules, but DDoS attack can generate linearly inseparable data due to abrupt, unstable, and stochastic characteristics. Considering that the multiple-kernel learning model has a low requirement for data stability and can be used for nonlinear fitting, and it can treat flexibly linear and nonlinear data, this paper proposes an adaptive DDoS attack detection method based on the ensemble learning framework.

4.1. The Multiple-Kernel Learning Model. The multiple-kernel learning (MKL) model is developed from the original single-kernel SVM. In single-kernel SVM, a SVM only uses one kernel function to map the sample to high-dimensional spaces. By comparison, the multiple-kernel learning model uses multiple-kernel functions with weight to map the sample to high-dimensional space. Therefore, it has higher flexibility and adaptability on heterogeneous data.

The multiple-kernel learning is defined as follows: given training set $\mathrm{T} = \{(x_1, y_1), (x_2, y_2) \cdots (x_n, y_n)\}$,

testing set $C = \{x_1', x_2' \cdots x_s'\}$, $x_\mathrm{i} \in R^d$, $x_\mathrm{k}' \in R^d$, $y_i \in (-1, +1)$, R is real-number set, d is data dimension, $i = 1, 2, \cdots, n$, $k = 1, 2, \cdots s$. $K_1(x, x'), K_2(x, x') \cdots K_M(x, x')$ are kernel functions in $R^d \times R^d$, and $\phi_1, \phi_2 \cdots \phi_M$ is a kernel mapping for each function. In the classic multiple-kernel learning SimpleMKL [52], the objective function of the hyperplane is as follows:

$$f(x) = \sum_{m=1}^{M}\left(\omega_m, \phi_m(x)\right) + b \tag{14}$$

where ω_m is the weight for each kernel function, and b is bias. The relaxation factor is ξ. According to the principle of minimum structure, the objective function can be optimized as follows:

$$\min \quad \psi\left(\omega_\mathrm{m}, b, \xi, d\right) = \frac{1}{2}\sum_{m-1}^{M}\frac{1}{d_m}\left\|\omega_\mathrm{m}\right\|_{\mathrm{H_m}}^2 + \mathrm{C}\sum_{\mathrm{i}-1}^{\mathrm{n}}\xi_\mathrm{i} \tag{15}$$

$$\begin{aligned} \text{s.t.} \quad & \mathrm{y_i}\sum_{\mathrm{m}=1}^{M}\omega_m \cdot \varphi\left(x_i\right) + \mathrm{y_i}b \geq 1 - \xi_i \\ & \sum_{m=1}^{M} d_m = 1, \quad d_m \geq 0 \\ & \xi_i \geq 0 \end{aligned} \tag{16}$$

By the two-order alternation optimization, the formula (15) can be converted to the optimization problem with d_m as the variable:

$$\min_{d \geq 0} \quad J(d), \sum_{m=1}^{M} d_m = 1 \tag{17}$$

$$\begin{aligned} \text{s.t.} \quad & \min_{\omega_m, b, \xi} = \frac{1}{2}\sum_{m-1}^{M}\frac{1}{d_m}\left\|\omega_\mathrm{m}\right\|_{\mathrm{H_m}}^2 + \mathrm{C}\sum_{\mathrm{i}-1}^{n}\xi_\mathrm{i} \\ & \mathrm{y_i}\sum_{\mathrm{m}=1}^{M}\omega_m \cdot \varphi\left(x_i\right) + \mathrm{y_i}b \geq 1 - \xi_i \\ & \xi_i \geq 0. \end{aligned} \tag{18}$$

The Lagrange function of $J(d)$ is as follows:

$$\begin{aligned} \mathrm{L} = & \frac{1}{2}\sum_{m-1}^{M}\frac{1}{d_m}\left\|\omega_\mathrm{m}\right\|_{\mathrm{H_m}}^2 + \mathrm{C}\sum_{\mathrm{i}=1}^{\mathrm{n}}\xi_\mathrm{i} \\ & + \sum_{\mathrm{i}=1}^{\mathrm{m}}\alpha_\mathrm{i}\left(1 - \xi_i - \mathrm{y_i}\sum_{\mathrm{m}=1}^{M}\omega_m \cdot \varphi_\mathrm{m}\left(x_i\right) + \mathrm{y_i}b\right) \\ & + \sum_{\mathrm{i}=1}^{\mathrm{n}}\nu_i\xi_i \end{aligned} \tag{19}$$

where α_i, ν_i are Lagrange operators. First, ω_m, b, ξ_i are calculated for partial derivatives. Then, the extremums are gained when the partial derivatives are "0." Finally, extremums are brought into the Lagrange function, which can be further changed to

$$\max \quad Q(\alpha) = -\frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j K_d\left(x_i, x_j\right) + \sum_{i=1}^{n}\alpha_i \tag{20}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{i}^{n}\alpha_i y_i = 0 \\ & C \geq \alpha_i \geq 0 \\ & K_d\left(x_i, x_j\right) = \sum_{m=1}^{M} d_m k_m\left(x_i, x_j\right). \end{aligned} \tag{21}$$

The gradient descent method is used to adjust $J(d)$ on d, update d, and optimize the d as well as a alternately. Then, an optimal solution is obtained:

$\alpha^* = (\alpha_1, \alpha_2, \cdots, \alpha_n)$; that is, the original objective function eventually turns into (22). The detailed formulation is as follows:

$$f(x) = \sum_{i=1}^{n}\alpha_i^* y_i \sum_{m=1}^{M} d_m K_d\left(x_i, x_j\right) + b \tag{22}$$

$x_j \in C$. When the test set data as x_j is inputted to $f(x)$, the object function can determine the category of test set data.

4.2. The Attack Detection Model Based on Multiple-Kernel Learning. The SimpleMKL model can be suitable for all the dimension weight values with "1". But it cannot fully exert the different features. This paper uses the feature weights to control the effect of different features on the model. To gain the appropriate feature weights in the SimpleMKL model, we combine the gradient method to optimize the weight parameters, so that the detection accuracy is further improved.

We marked ACD as x_1, IBF as x_2, MFF as x_3, HIAD as x_4, and FFV as x_5, then the feature value vector is $F = (x_1, x_2, x_3, x_4, x_5)$, and the marked weight vector is $W = (w_1, w_2, w_3, w_4, w_5)$. Combinatorial features are $CF = F * W^T$, and the mean value of each dimension of normal flow is $u_{11}, u_{12}, u_{13}, u_{14}$, or u_{15}. Note the mean value of each dimension of the attack flow is $u_{21}, u_{22}, u_{23}, u_{24}$, or u_{25}.

The interclass mean squared difference is expressed as follows:

$$\begin{aligned} M = {} & \left[w_1 * \left(u_{11} - u_{21}\right)\right]^2 + \left[w_2 * \left(u_{12} - u_{22}\right)\right]^2 \\ & + \left[w_3 * \left(u_{13} - u_{23}\right)\right]^2 + \left[w_4 * \left(u_{14} - u_{24}\right)\right]^2 \\ & + \left[w_5 * \left(u_{15} - u_{25}\right)\right]^2 \end{aligned} \tag{23}$$

The normal intraclass variance is denoted:

$$\begin{aligned} S_1 = {} & \sum_{i=1}^{n}\left[w_1 * \left(x_{i1} - u_{11}\right)\right]^2 + \left[w_2 * \left(x_{i2} - u_{12}\right)\right]^2 \\ & + \left[w_3 * \left(x_{i3} - u_{13}\right)\right]^2 + \left[w_4 * \left(x_{i4} - u_{14}\right)\right]^2 \\ & + \left[w_5 * \left(x_{i5} - u_{15}\right)\right]^2 \end{aligned} \tag{24}$$

The attack intraclass variance is denoted:

$$\begin{aligned} S_2 = {} & \sum_{i=1}^{n}\left[w_1 * \left(x_{i1} - u_{21}\right)\right]^2 + \left[w_2 * \left(x_{i2} - u_{22}\right)\right]^2 \\ & + \left[w_3 * \left(x_{i3} - u_{23}\right)\right]^2 + \left[w_4 * \left(x_{i4} - u_{24}\right)\right]^2 \\ & + \left[w_5 * \left(x_{i5} - u_{25}\right)\right]^2 \end{aligned} \tag{25}$$

The intraclass variance is $S = S_1 + S_2$. To improve classification accuracy and ensure a rapid convergence of functions, on the one hand, we should try to improve the mean difference between positive and negative samples, so that the two kinds of samples are far away from each other; that is, we should increase the M value. On the other hand, we should minimize the differences between samples. The variance corresponding to each dimension should be as small as possible, thus reducing the S value. Therefore, the classification model needs to train two different classifiers to classify the samples. One classifier is interclass mean squared difference growth (M-SMKL) and the other classifier is intraclass variance descent (S-SMKL). In combination with the SimpleMKL framework formula (15), the above problems can be transformed into (26). The detailed formulation is as follows:

$$\begin{aligned} & \max_{x_{ij}\in F} \quad \alpha M + \min_{x_{ij}\in F} \beta S \\ & \min \quad \psi\left(\omega_{\mathrm{m}}, b, \xi, d, w\right) = \frac{1}{2}\sum_{m-1}^{M}\frac{1}{d_m}\left\|\omega_{\mathrm{m}}\right\|_{\mathrm{H_m}}^2 + \mathrm{C}\sum_{\mathrm{i-1}}^{\mathrm{n}}\xi_{\mathrm{i}} \end{aligned} \tag{26}$$

$$\begin{aligned} \text{s.t.} \quad & \mathrm{y_i}\sum_{m=1}^{M}\omega_m \cdot \varphi\left(wx_i\right) + y_i b \geq 1 - \xi_i \\ & \sum_{m=1}^{M} d_m = 1, \quad d_m \geq 0 \\ & \xi_i \geq 0 \\ & M < \sigma_1 \\ & S > \sigma_1. \end{aligned} \tag{27}$$

If $\alpha \gg \beta$, the objective function is M-SMKL. If $\beta \gg \alpha$, the objective function is S-SMKL. α and β are converted to the learning rate of formula (35).

To solve the above problems, we use the way of updating iterative weights to get the objective function. The details are as follows. Firstly, the weights of each feature are assigned

initial values. Secondly, they are combined with (26) and (27) to gain optimal function of this time. The mathematical form is expressed as follows:

$$\max \quad Q(\alpha) = -\frac{1}{2}\sum_{\mathrm{i,j}=1}^{m}\alpha_i\alpha_j y_i y_j K_d\left(wx_i, wx_j\right) + \sum_{i=1}^{n}\alpha_i \tag{28}$$

$$\begin{aligned} \text{s.t.} \quad & \sum_{\mathrm{i}}^{n}\alpha_i y_i = 0 \\ & C \geq \alpha_i \geq 0 \\ & K_d\left(wx_i, wx_j\right) = \sum_{m=1}^{M} d_m k_m\left(wx_i, wx_j\right) \end{aligned} \tag{29}$$

The optimal equation obtained using (28) and (29) is as follows:

$$f(x) = \sum_{i=1}^{n}\alpha_i^* y_i \sum_{m=1}^{M} d_m K_d\left(wx_i, wx_j\right) + b. \tag{30}$$

To further determine whether the optimal equation has achieved good results, this paper sets two constraint conditions for M-SMKL and S-SMKL, respectively, without conflict with formula (27) constraint conditions. These constraint conditions are expressed as follows.

The constraint conditions of M-SMKL are as follows:

$$\begin{aligned} & t_1 < \left|M_{i+1} - M_i\right| < t_2 \\ & \quad < \left|M_{\mathrm{i}} - M_{i-1}\right| < t_3 \\ & \frac{f(M_i)}{f(S_i)} - \frac{f(M_{i-1})}{f(S_{i-1})} > p_1 \\ & \frac{f(M_i)}{f(S_i)} - \frac{f(M_{i+1})}{f(S_{i+1})} > p_2 \end{aligned} \tag{31}$$

The constraint conditions of S-SMKL are as follows:

$$\begin{aligned} & t_4 < \left|S_{\mathrm{i}} - S_{i-1}\right| < t_5 < \left|S_{i+1} - S_i\right| \\ & \quad < t_6 \\ & \frac{f(M_i)}{f(S_i)} - \frac{f(M_{i-1})}{f(S_{i-1})} > p_3 \\ & \frac{f(M_i)}{f(S_i)} - \frac{f(M_{i+1})}{f(S_{i+1})} > p_4, \end{aligned} \tag{32}$$

where the values of p_1, p_2, p_3, and p_4 are close to "0"; the values of t_1, t_2, and t_3 are close to "1"; the values of t_4, t_5, and t_6 are close to "7.5". If the constraint condition is satisfied, the algorithm will be stopped and formula (30) will become the optimal function; otherwise, each dimension weight will be updated iteratively. The gradient of M and S corresponding to each dimension weight is as follows:

$$\begin{aligned} & \frac{\partial M}{\partial w_1} = 2w_1\left(u_{11} - u_{21}\right)^2 \\ & \frac{\partial M}{\partial w_2} = 2w_2\left(u_{12} - u_{22}\right)^2 \\ & \frac{\partial M}{\partial w_3} = 2w_3\left(u_{13} - u_{23}\right)^2 \\ & \frac{\partial M}{\partial w_4} = 2w_4\left(u_{14} - u_{24}\right)^2 \\ & \frac{\partial M}{\partial w_5} = 2w_5\left(u_{15} - u_{25}\right)^2 \end{aligned} \tag{33}$$

$$\begin{aligned} & \frac{\partial S}{\partial w_1} = 2\left[w_1\left(\sum_{i=1}^{n_1} x_{11}^2 - n_1 u_{11}^2\right) + w_1\left(\sum_{i=1}^{n_2} x_{21}^2 - n_2 u_{21}^2\right)\right] \\ & \frac{\partial S}{\partial w_2} = 2\left[w_2\left(\sum_{i=1}^{n_1} x_{12}^2 - n_1 u_{12}^2\right) + w_2\left(\sum_{i=1}^{n_2} x_{22}^2 - n_2 u_{22}^2\right)\right] \\ & \frac{\partial S}{\partial w_3} = 2\left[w_3\left(\sum_{i=1}^{n_1} x_{13}^2 - n_1 u_{13}^2\right) + w_3\left(\sum_{i=1}^{n_2} x_{23}^2 - n_2 u_{23}^2\right)\right] \\ & \frac{\partial S}{\partial w_4} = 2\left[w_4\left(\sum_{i=1}^{n_1} x_{14}^2 - n_1 u_{14}^2\right) + w_4\left(\sum_{i=1}^{n_2} x_{24}^2 - n_2 u_{24}^2\right)\right] \\ & \frac{\partial S}{\partial w_5} = 2\left[w_5\left(\sum_{i=1}^{n_1} x_{15}^2 - n_1 u_{15}^2\right) + w_5\left(\sum_{i=1}^{n_2} x_{25}^2 - n_2 u_{25}^2\right)\right] \end{aligned} \tag{34}$$

where n_1 is the number of the normal flow feature of the training sample; n_2 is the number of the attack flow feature of

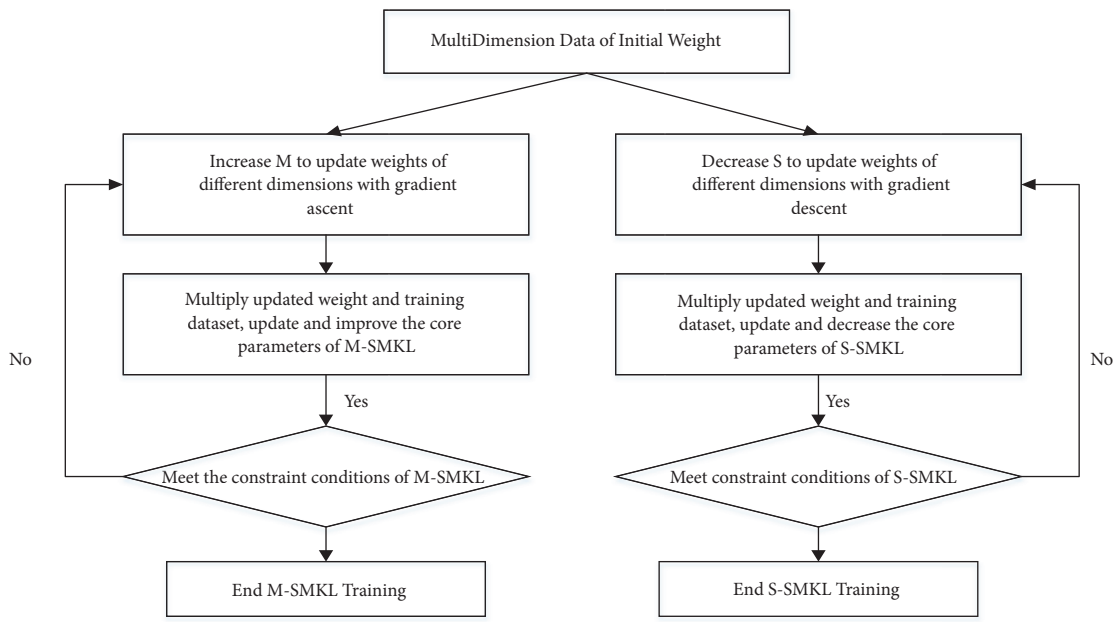


Figure 1: Flow chart of multiple-kernel learning training process based on ensemble learning.

the training sample. According to gradients in (33) and (34), the weight of each dimension is updated as follows (35):

$$\begin{aligned} w_1 &= w_1 + 2 * lr_1 * \frac{\partial M}{\partial w_1} - 2 * lr_2 * \frac{\partial S}{\partial w_1} \\ w_2 &= w_2 + 2 * lr_1 * \frac{\partial M}{\partial w_2} - 2 * lr_2 * \frac{\partial S}{\partial w_2} \\ w_3 &= w_3 + 2 * lr_1 * \frac{\partial M}{\partial w_3} - 2 * lr_2 * \frac{\partial S}{\partial w_3} \\ w_4 &= w_4 + 2 * lr_4 * \frac{\partial M}{\partial w_4} - 2 * lr_2 * \frac{\partial S}{\partial w_4} \\ w_5 &= w_5 + 2 * lr_1 * \frac{\partial M}{\partial w_5} - 2 * lr_2 * \frac{\partial S}{\partial w_5} \end{aligned} \tag{35}$$

where lr_1 is the learning rate of gradient ascent; lr_2 is the learning rate of gradient descent. lr_1 has the same function as α and lr_2 has the same function as β. Each updated weight is multiplied by each original feature accordingly and the next round of iteration is carried out.

4.3. Framework of Multiple-Kernel Learning Detection Based on Ensemble Learning. We input the multidimensional data with weight and set the learning rate. Then two different classifiers are trained. M-SMKL is trained by increasing the M value mainly with reducing the S value secondarily and the S-SMKL is trained by reducing the S value mainly with increasing the M value secondarily. During the training process, the M value and the S value are constantly updated with the method of gradient rising and descending until the constraint conditions are met. The flowchart is provided in Figure 1.

The detection process is as follows: firstly, the test data is multiplied with two different weight vectors which are trained earlier; secondly, the calculated data are inputted to the corresponding M-SMKL and S-SMKL model; finally, we use the sliding window mechanism to coordinate two kinds of models. The sliding window mechanism is described as follows. Firstly, a sliding window with a size of n is created. Secondly, the trained M-SMKL classifies the test data and obtains the first classification results; the trained S-SMKL classifies the test data and obtains the second classification results. Finally, four ways are used to cooperatively treat the first classification results and the second classification results; the details are as follows: (1) if M-SMKL and S-SMKL identify that the current data category is both normal, the current data category is judged to be normal; (2) if M-SMKL and S-SMKL identify that the current data category is both attack, the current data category is judged to be attack; (3) if M-SMKL identifies that the current data category is normal but S-SMKL identifies that the current data category is attack, the current data category is judged to be attack; (4) if M-SMKL identifies that the current data category is attack but S-SMKL identifies that the current data category is normal, then consider the following. *Step 1.* Move the starting point of the sliding window to the current position of the test data in the first classification result, and map the end point of the sliding window to the n-1 position of the first classification results. *Step 2.* If the results in the sliding window are all attack, the current data category is judged to be attack; otherwise, the current data category is judged to be normal. The flow chart is provided in Figure 2.

The reason for the training of two kinds of SMKL is that S-SMKL focuses on reducing the difference between the data of each dimension and can assemble the two types of samples in their respective central positions. However,

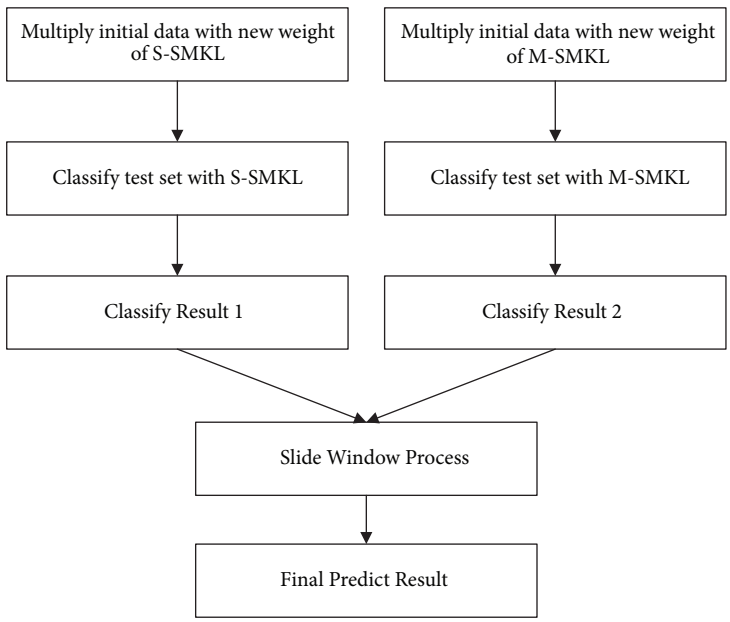


Figure 2: Flow chart of multiple-kernel learning detection process based on ensemble learning.

S-SMKL does not consider the location of the two sample-center points. Although a better classification feature can be maintained on the whole, it is impossible to identify DDoS attacks earlier because the center distance of the normal flow and attack flow is small. M-SMKL focuses on the difference between the two types of data centers and maximizes the sample centers distance between the two types of sample centers, making the two samples as separate as possible. M-SMKL can expand the distance of different class so that the attack flow can be identified earlier but it makes intraclass data dispersed, causing default results. Therefore, the sliding window mechanism is adopted to coordinate the two models to detect early DDoS accurately.

5. Experimental Analysis

5.1. Experimental Data Sets and Evaluation Standards. The data set used for this experiment is the CAIDA "DDoS Attack 2007" data set [53]. This data set contains an [L1] Distributed Denial of Service (DDoS) anonymous traffic attack for approximately one hour on August 4, 2007. The total size of the data set is 21 GB, which accounts for approximately one hour (20:50:08 UTC–21:56:16 UTC). Attacks began around 21:13, causing the network load to grow rapidly (in minutes) from approximately 200 kbits/s to 80 megabits/s. One hour of attack traffic is divided into 5 minutes of files and stored in PCAP format. The contents of this data set are TCP network traffic packets. Each TCP packet contains the source address, destination address, source port, destination port, packet size, and protocol type. The duration of normal flow data used in this paper is 2 minutes in total, and the duration of attack data is 5 minutes in total.

The hardware equipment adopted is 8 GB memory, Intel Core i7 processor, and a computer with a Windows 10 64-bit system; the development environment is MATLAB 2014a and Codeblocks 10.05. The evaluation criteria used in this paper consist of the detection rate (DR), the false alarm rate (FR), and total error rate (ER).

Assume that TP indicates that the number of normal test samples is properly marked, FP indicates the number of normal test samples that have been incorrectly marked, TN indicates the number of attack test samples that are correctly marked, and FN indicates the number of attack test samples that have been incorrectly marked:

$$DR = \frac{TN}{TN + FN}$$
$$FR = \frac{FP}{TP + FP} \tag{36}$$
$$ER = \frac{FN + FP}{TP + FP + TN + FN}.$$

We used the above five feature extraction algorithms to extract features from the data set. The extracted feature values are normalized and used as a training set. The data in the training set can be regarded as the regularity of the change in network traffic. The network traffic has an abrupt and volatile nature. Therefore, although the collected network data have similarities with the conventional ones, they still have a certain degree of difference. To simulate this phenomenon for verifying the effectiveness of the presented method, three types of data are generated as follows. (1) Normal flow feature values and attack flow feature values are multiplied by random number; (2) only the attack flow feature values are multiplied by random number; and (3) only the normal flow feature values are multiplied by random number.

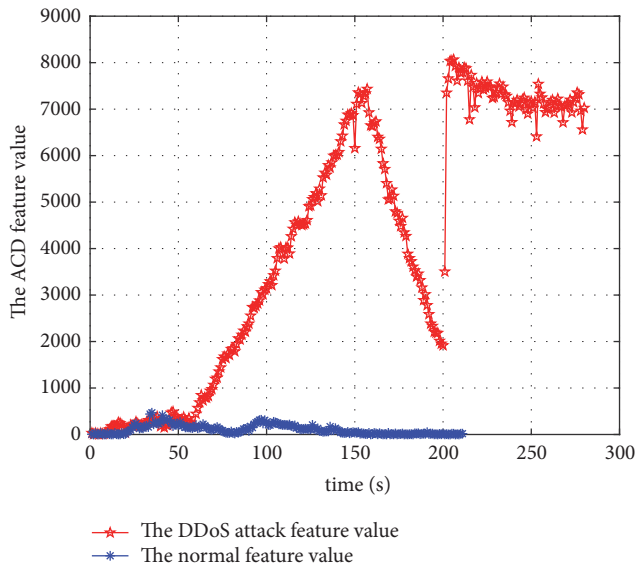


Figure 3: The ACD feature graph of DDoS attack flow and normal flow.

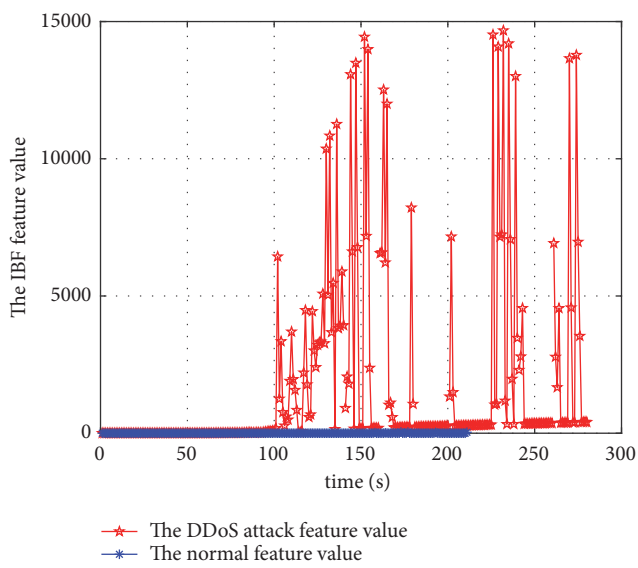


Figure 4: The IBF feature graph of DDoS attack flow and normal flow.

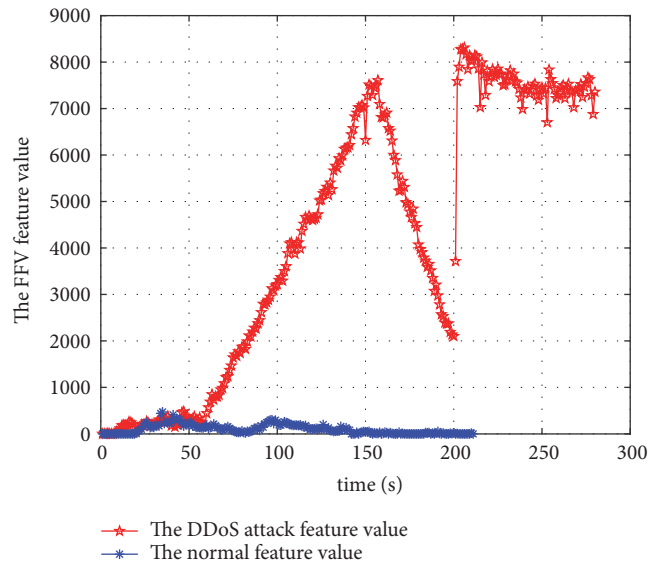


Figure 5: The FFV feature graph of DDoS attack flow and normal flow.

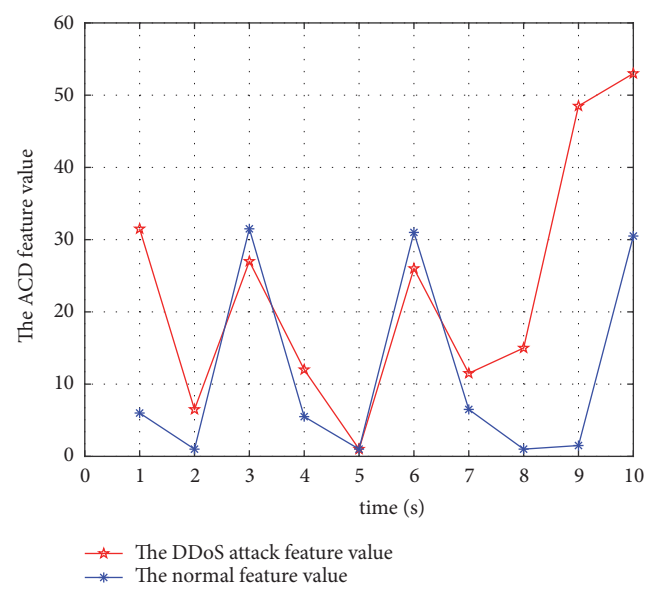


Figure 6: The ACD feature graph of DDoS attack flow and normal flow in the first 10 seconds.

5.2. Experimental Results and Analysis. Five features are used to extract feature data from attack data and normal data, and positive as well as negative sample sets are obtained. The sampling time is set to 1 s, and the remaining parameters of the five feature extraction methods are set as follows: $\theta_1 = 0.5$, $\theta_2 = 0.5$, $\theta_3 = 3$, $\theta_4 = 3$, $\theta_5 = 3$, $\theta_6 = 3$, $\theta_7 = 3$, $\theta_8 = 3$, and $\theta_9 = 3$. The total of normal feature values is 211 and the total of attack feature values is 280. Figures 3–9 illustrate the feature values extracted by the five algorithms.

As illustrated in Figure 3, the early attack feature values of DDoS attack are close to the normal feature values. This is because there are a large number of bidirectional flows in the early stage of the DDoS attack and these bidirectional flows gradually decrease with the increase of the attack degree. Therefore, using the ACD as a feature after 70 seconds can significantly reflect the difference between the attack flow and the normal flow. ACD can reflect the difference between normal flow and attack flow the earliest.

As illustrated in Figure 4, compared with ACD, although IBF does not recognize the attack flow earlier, the distribution range of its feature values is more uniform and presents a certain degree of volatility. This makes the feature less susceptible to individual outliers.

As illustrated in Figure 5, the FFV feature is very similar to the ACD, but as illustrated in Figures 6 and 7, in the initial stage, the FFV is more capable of reflecting the difference between the attack flow and the normal flow than the ACD is.

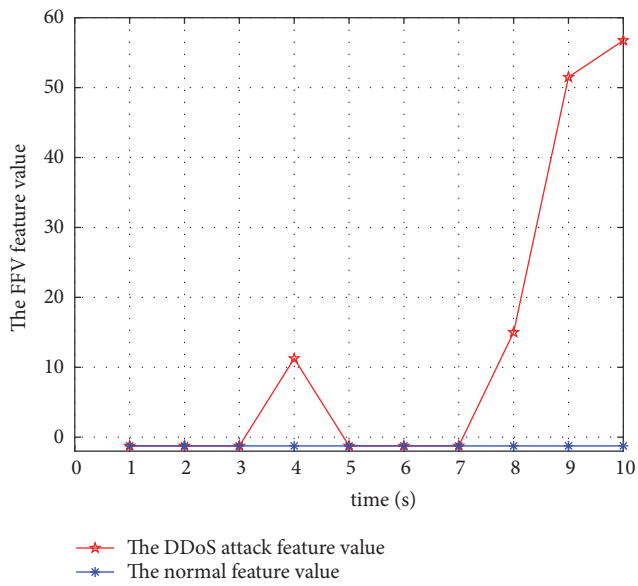


Figure 7: The FFV feature graph of DDoS attack flow and normal flow in the first 10 seconds.

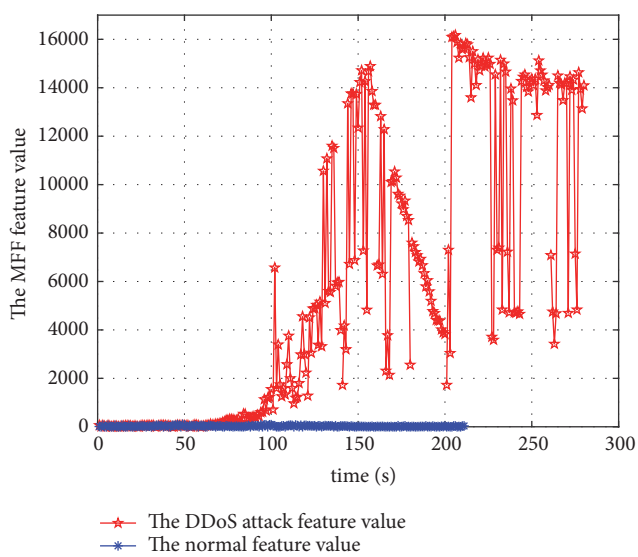


Figure 8: The MFF feature graph of DDoS attack flow and normal flow.

As illustrated in Figure 8, although the MFF feature cannot determine the attack flow and the normal flow as early as possible, it can make the feature values of the attack stage more stable, so that it can avoid the outliers of attack flows.

As illustrated in Figure 9, it can be seen from the value of the ordinate that the HIAD best reflects the difference between the normal flow and the attack flow while having better stability in the latter half of the attack flow. After the early data, this feature can greatly distinguish between normal flow and abnormal flow, influence the classifier more, and make better decisions.

In summary, all five features have their own unique characteristics. To make full use of the characteristics of each feature, the feature values extracted by these five algorithms

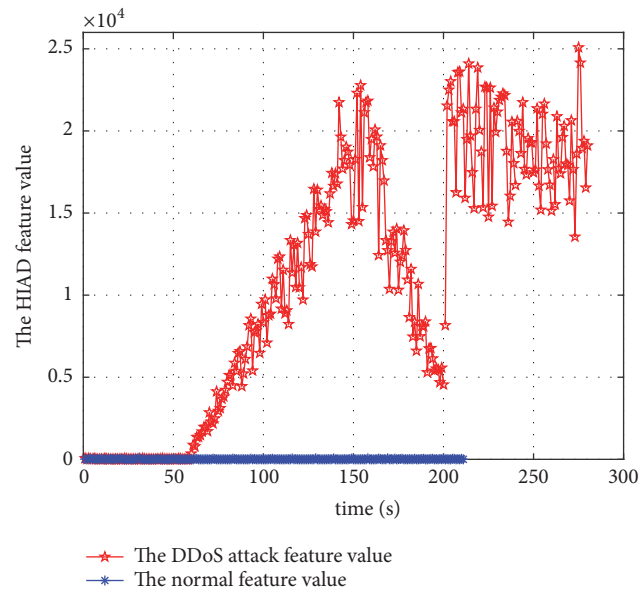


Figure 9: The HIAD feature graph of DDoS attack flow and normal flow.

are each used as a five-dimensional-feature data set. Using these five feature values as training sets, two multiple-kernel learning models dominated by gradient ascent and gradient descent are trained into the algorithm, and corresponding five-dimensional feature weight vectors are obtained. Finally, according to the framework of Figure 2, the classification results of test set are obtained and are used to verify the effectiveness of method. The parameters of M-SMKL are set as follows: $l_{r_1} = 2 * 10^{-5}$, $l_{r_2} = 2 * 10^{-3}$, $t_1 = 1.002$, $t_2 = 1.0065$, $t_3 = 1.007$, $p_1 = 0.000084$, and $p_2 = 0.000001$. The parameters of S-SMKL are set as follows: $l_{r_1} = 2 * 10^{-5}$, $l_{r_2} = 2 * 10^{-2}$, $t_4 = 7.3425$, $t_5 = 7.8340$, $t_6 = 7.8350$, $p_3 = 0.000775$, and $p_4 = 0.000680$. The size of the sliding window is 8. The parameters for multiple-kernel learning are all default values, and the kernel function includes two Gaussian functions and two polynomial functions. The SVM parameters are all default values, and the kernel function is linear function. The experimental results are illustrated in Figures 10–18.

As shown in Figures 10–18, under the three types of experiments, according to the three evaluation criteria, the overall performance of the algorithms from the highest to the lowest is the ADADM, the SVM method, the SMKL method, and Nezhad et al.'s method [16].

This is because although the method described by Nezhad et al. [16] is visibly superior to other methods in terms of DR indicators, it is far worse than other methods with respect to other indicators. The reason is that the Nezhad et al. [16] method relies excessively on the first reference point. When the first reference point fluctuates, this method recognizes easily some normal samples as attack samples.

Although the classification accuracy of the attack samples is high, a large number of normal samples are misjudged, so this method is superior in terms of DR and its other indicators are inferior to those of other methods. This is

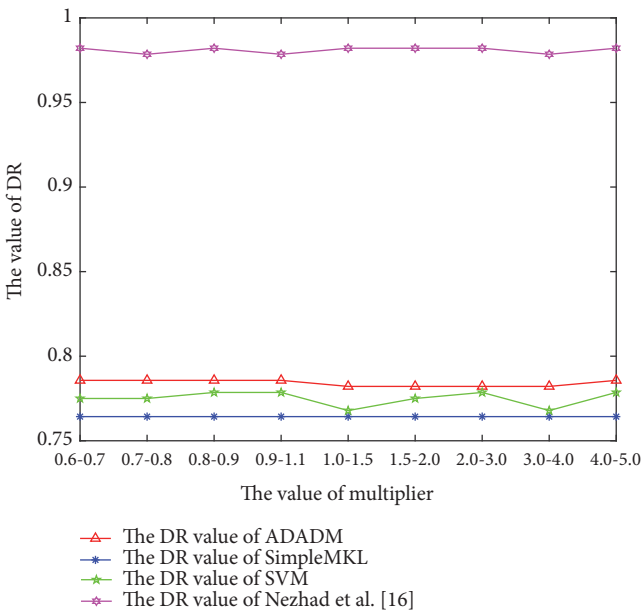


Figure 10: The DR contrast diagram of four algorithms for scaling attack flow and normal flow.

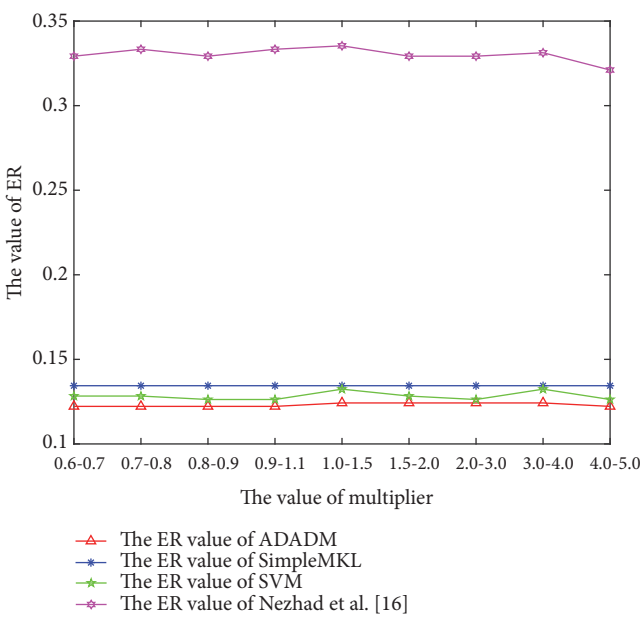


Figure 11: The ER contrast diagram of four algorithms for scaling attack flow and normal flow.

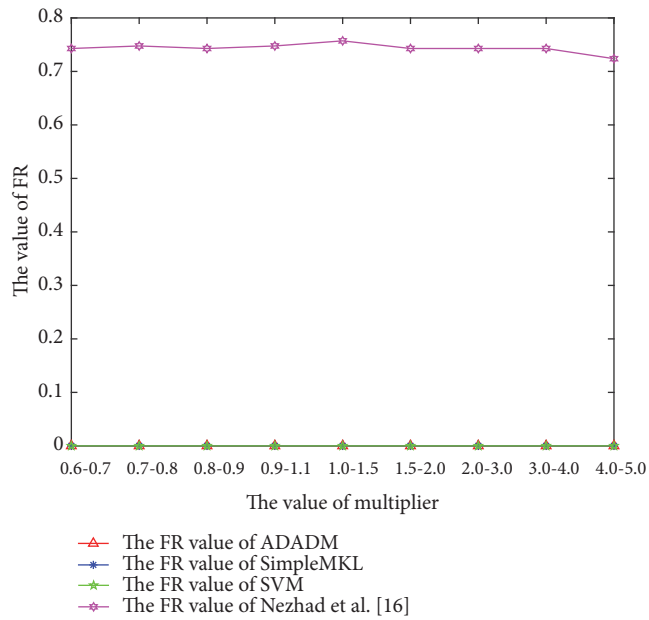


Figure 12: The FR contrast diagram of four algorithms for scaling attack flow and normal flow.

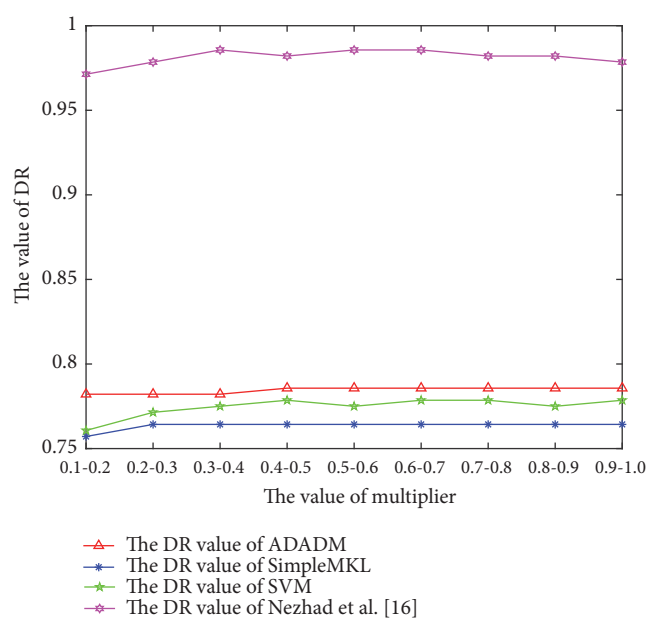


Figure 13: The DR contrast diagram of four algorithms for narrowing the attack flow.

why, in this case, the Nezhad et al. [16] method performs the worst. The effect of SVM is generally better than that of the SMKL method because although the SMKL method coordinates multiple-kernel functions to map the sample to a high-dimensional Hilbert space, the linear kernel function is obviously more suitable for the sample. Using the linear kernel SVM can establish a better hyperplane than the SMKL method to identify the data containing early DDoS attacks. However, although the multiple-kernel learning method does not use a linear kernel function that is more suitable for the sample space, it can still maintain high accuracy, indicating that multiple-kernel learning has a lower dependence on the selection of kernel functions than the single-kernel SVM.

We compared the ADADM to the SVM method. The ADADM method uses the same kernel function as SMKL method. Because the multikernel learning method is flexible and adaptable, it is possible to continuously optimize the hyperplane by adjusting the weights of the feature of each dimension to recognize the DDoS as early as possible. Attack

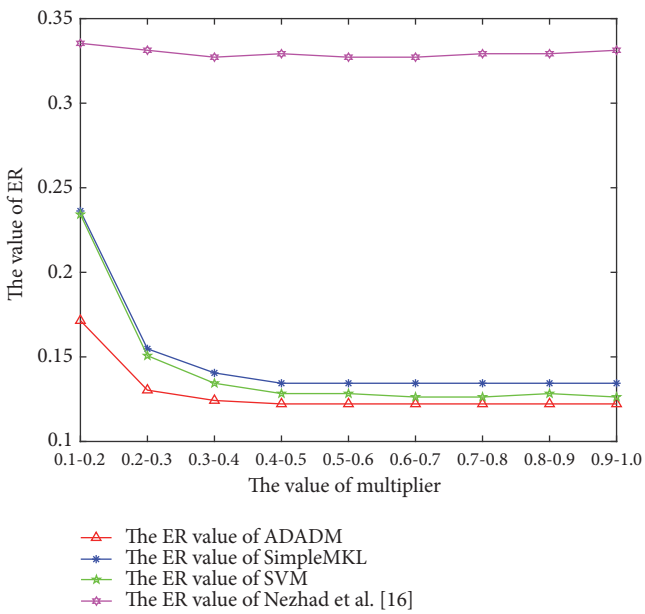


Figure 14: The ER contrast diagram of four algorithms for narrowing the attack flow.

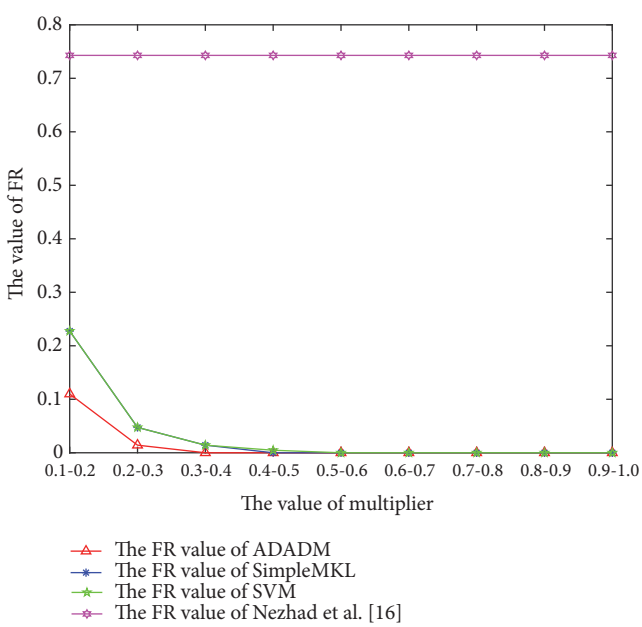


Figure 15: The FR contrast diagram of four algorithms for narrowing the attack flow.

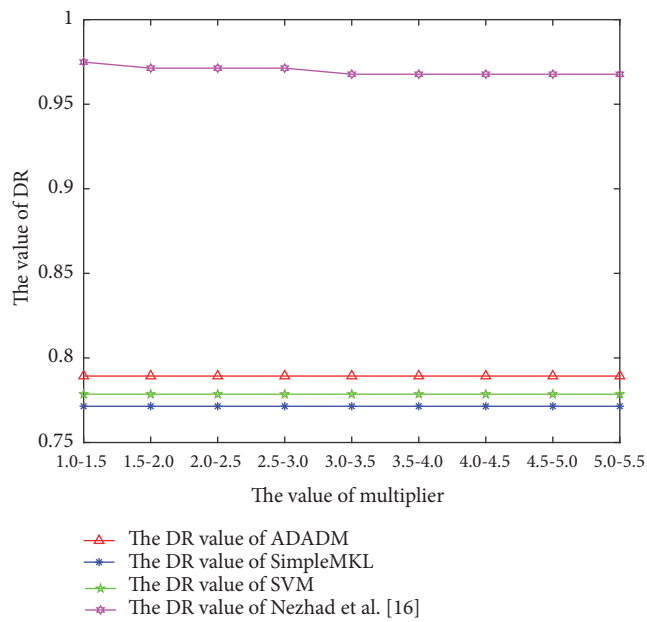


Figure 16: The DR contrast diagram of four algorithms for amplifying the normal flow.

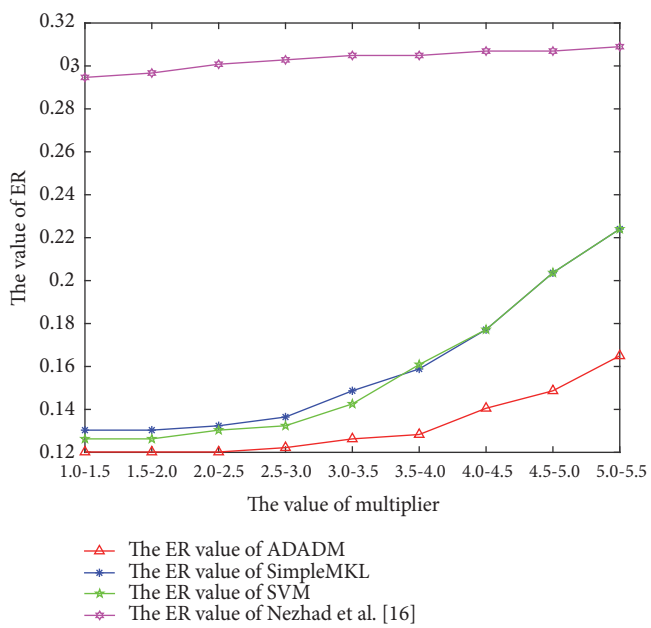


Figure 17: The ER contrast diagram of four algorithms for amplifying the normal flow.

flow data and normal flow data are located on both sides of the hyperplane.

In addition, using the idea of ensemble learning to train two different classifiers and using the sliding window mechanism to further synthesize the advantage of each classifier improves the algorithm's performance in the three types of experiments. This method we propose outperforms not only the SVM method but also other methods of DDoS attack detection. The experimental data are presented in Tables 1, 2, and 3.

6. Conclusion

In this paper, five-dimensional features are defined for describing the burstiness of DDoS attack flows, the distribution of IP source addresses, and the interactivity of DDoS attack flows. Based on the five-dimensional

Table 1: Comparison results of four algorithms for scaling attack flow and normal flow.

		The value of the random multiplier								
		0.6–0.7	0.7–0.8	0.8–0.9	0.9–1.1	1.0–1.5	1.5–2.0	2.0–3.0	3.0–4.0	4.0–5.0
ADADM method	DR (%)	78.57	78.57	78.57	78.57	78.21	78.21	78.21	78.21	78.57
	FR (%)	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	ER (%)	12.22	12.22	12.22	12.22	12.42	12.42	12.42	12.42	12.22
SimpleMKL method	DR (%)	76.43	76.43	76.43	76.43	76.43	76.43	76.43	76.43	76.43
	FR (%)	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	ER (%)	13.44	13.44	13.44	13.44	13.44	13.44	13.44	13.44	13.44
SVM method	DR (%)	77.50	77.50	77.86	77.86	76.79	77.50	77.86	76.79	77.86
	FR (%)	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	ER (%)	12.83	12.83	12.63	12.63	13.24	12.83	12.63	13.24	12.63
Nezhad et al.'s [16] method	DR (%)	98.21	97.85	98.21	97.85	98.21	98.21	98.21	97.85	98.21
	FR (%)	74.29	74.76	74.29	74.76	75.71	74.29	74.29	74.29	72.38
	ER (%)	32.92	33.33	32.92	33.33	33.54	32.92	32.92	33.13	32.11

Table 2: Comparison results of four algorithms for narrowing the attack flow.

		The value of the random multiplier								
		0.1–0.2	0.2–0.3	0.3–0.4	0.4–0.5	0.5–0.6	0.6–0.7	0.7–0.8	0.8–0.9	0.9–1.0
ADADM method	DR (%)	78.21	78.21	78.21	78.57	78.57	78.57	78.57	78.57	78.57
	FR (%)	10.99	1.42	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	ER (%)	17.15	13.04	12.42	12.22	12.22	12.22	12.22	12.22	12.22
SimpleMKL method	DR (%)	75.71	76.43	76.43	76.43	76.43	76.43	76.43	76.43	76.43
	FR (%)	22.75	4.74	1.42	0.01	0.01	0.01	0.01	0.01	0.01
	ER (%)	23.63	15.48	14.05	13.44	13.44	13.44	13.44	13.44	13.44
SVM method	DR (%)	76.07	77.14	77.50	77.86	77.50	77.86	77.86	77.50	77.86
	FR(%)	22.75	4.74	1.42	0.47	0.01	0.01	0.01	0.01	0.01
	ER(%)	23.42	15.07	13.44	12.83	12.83	12.63	12.63	12.83	12.63
Nezhad et al.'s [16] method	DR(%)	97.13	97.85	98.57	98.21	98.57	98.57	98.21	98.21	97.85
	FR(%)	74.29	74.29	74.29	74.29	74.29	74.29	74.29	74.29	74.29
	ER (%)	33.54	33.13	32.72	32.92	32.72	32.72	32.92	32.92	33.13

Table 3: Comparison results of four algorithms for amplifying the normal flow.

		The value of random multiplier								
		1.0–1.5	1.5–2.0	2.0–2.5	2.5–3.0	3.0–3.5	3.5–4.0	4.0–4.5	4.5–5.0	5.0–5.5
ADADM method	DR (%)	78.93	78.93	78.93	78.93	78.93	78.93	78.93	78.93	78.93
	FR (%)	0.01	0.01	0.01	0.47	1.42	1.90	4.74	6.64	10.43
	ER (%)	12.02	12.02	12.02	12.22	12.63	12.83	14.05	14.87	16.50
SimpleMKL method	DR (%)	77.14	77.14	77.14	77.14	77.14	77.14	77.14	77.14	77.14
	FR (%)	0.01	0.01	0.47	1.42	4.27	6.64	10.90	17.06	21.80
	ER (%)	13.04	13.04	13.24	13.65	14.87	15.89	17.72	20.37	22.40
SVM method	DR (%)	77.86	77.86	77.86	77.86	77.86	77.86	77.86	77.86	77.86
	FR (%)	0.01	0.01	0.95	1.42	3.79	8.06	11.85	18.01	22.75
	ER (%)	12.63	12.63	13.04	13.24	14.26	16.09	17.72	20.37	22.40
Nezhad et al.'s [16] method	DR (%)	97.49	97.13	97.13	97.13	96.77	96.77	96.77	96.77	96.77
	FR (%)	65.24	65.24	66.19	66.67	66.67	66.67	67.14	67.14	67.62
	ER (%)	29.47	29.67	30.08	30.28	30.49	30.49	30.69	30.69	30.90

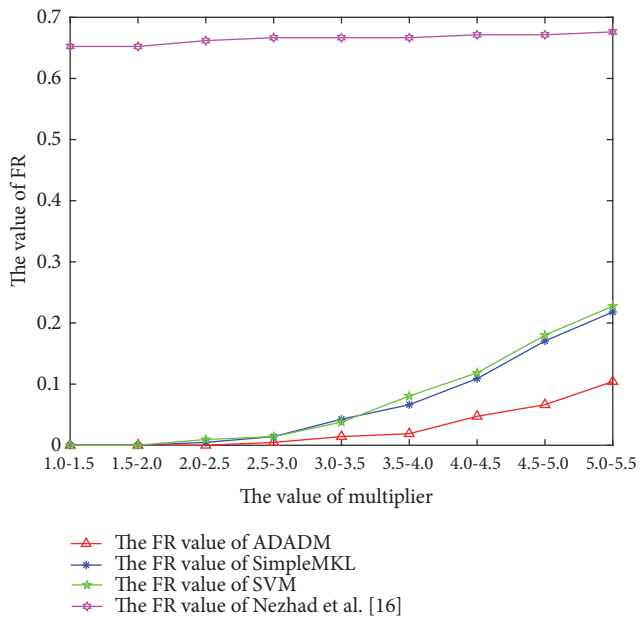


Figure 18: The FR contrast diagram of four algorithms for amplifying the normal flow.

features and the ensemble learning framework, adaptive feature weights are obtained and the M-SMKL and S-SMKL multiple-kernel learning models are trained to detect DDoS attack. For identifying early attacks effectively, the sliding window mechanism is used to coordinate the S-SMKL and the M-SMKL to deal with the detection results. Experimental results show that, compared with similar methods, our method, can produce more accurate results for detecting early DDoS attack.

In the follow-up work, we will further study how to transform the multidimensional weight adaptive problem based on multiple-kernel learning into a convex optimization problem and improve the detection rate and convergence speed of the method.

Data Availability

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

Conflicts of Interest

There are no conflicts of interest in this paper.

Acknowledgments

This work was supported by the Hainan Provincial Natural Science Foundation of China [2018CXTD333, 617048]; the National Natural Science Foundation of China [61762033, 61702539]; Hainan University Doctor Start Fund Project [kyqd1328]; and Hainan University Youth Fund Project [qnjj1444].

References

[1] Z. Cai, Z. Wang, K. Zheng, and J. Cao, "A Distributed TCAM coprocessor architecture for integrated longest prefix matching, policy filtering, and content filtering," *IEEE Transactions on Computers*, vol. 62, no. 3, pp. 417–427, 2013.

[2] J. H. Cui, Y. Y. Zhang, Z. P. Cai et al., "Securing display path for security-sensitive applications on mobile devices," *Computer, Materials & Continua*, vol. 55, no. 1, pp. 17–35, 2018.

[3] S. Liu, Z. Cai, H. Xu, and M. Xu, "Towards security-aware virtual network embedding," *Computer Networks*, vol. 91, pp. 151–163, 2015.

[4] A. S. Pimpalkar and A. R. Bhagat Patil, "Detection and defense mechanisms against DDoS attacks," in *Proceedings of the International Conference on Innovations in Information, Embedded and Communication Systems*, pp. 1–6, IEEE, 2015.

[5] J. R. Cheng, R. M. Xu, and X. Y. Tang, "An Abnormal Network Flow Feature Sequence Prediction Approach for DDoS Attacks Detection in Big Data Environment," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 95–119, 2018.

[6] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, "Botnet in DDoS attacks: trends and challenges," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2242–2270, 2015.

[7] K. Zeb, O. Baig, and M. K. Asif, "DDoS attacks and countermeasures in cyberspace," in *Proceedings of the 2015 2nd World Symposium on Web Applications and Networking, WSWAN '15*, pp. 1–6, IEEE, 2015.

[8] 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.

[9] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: a comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[10] Z. Zhou, M. Dong, K. Ota, G. Wang, and L. T. Yang, "Energy-efficient resource allocation for d2d communications underlaying cloud-ran-based lte-a networks," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 428–438, 2016.

[11] W. Lin, S. Xu, L. He, and J. Li, "Multi-resource scheduling and power simulation for cloud computing," *Information Sciences*, vol. 397-398, pp. 168–186, 2017.

[12] W. Jiang, G. Wang, M. Z. A. Bhuiyan, and J. Wu, "Understanding graph-based trust evaluation in online social networks: Methodologies and challenges," *ACM Computing Surveys*, vol. 49, no. 1, 2016.

[13] E. Luo, Q. Liu, and G. Wang, "Hierarchical Multi-Authority and Attribute-Based Encryption Friend Discovery Scheme in Mobile Social Networks," *IEEE Communications Letters*, vol. 20, no. 9, pp. 1772–1775, 2016.

[14] S. Peng, A. Yang, L. Cao, S. Yu, and D. Xie, "Social influence modeling using information theory in mobile social networks," *Information Sciences*, vol. 379, pp. 146–159, 2017.

[15] T. Peng, Q. Liu, D. Meng, and G. Wang, "Collaborative trajectory privacy preserving scheme in location-based services," *Information Sciences*, vol. 387, pp. 165–179, 2017.

[16] S. M. T. Nezhad, M. Nazari, and E. A. Gharavol, "A Novel DoS and DDoS Attacks Detection Algorithm Using ARIMA Time Series Model and Chaotic System in Computer Networks," *IEEE Communications Letters*, vol. 20, no. 4, pp. 700–703, 2016.

[17] R. H. Meng, S. G. Rice, and J. Wang, "A Fusion Steganographic Algorithm Based on Faster R-CNN," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 1–16, 2018.

[18] 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 '16)*, pp. 1–8, Fajardo, Puerto Rico, USA, October 2016.

[19] C. Yuan, X. Li, Q. M. J. Wu et al., "Fingerprint Liveness Detection from Different Fingerprint Materials Using Convolutional Neural Network and Principal Component Analysis," *CMC: Computers, Materials & Continua*, vol. 53, no. 3, pp. 357–371, 2017.

[20] M. I. Jordan and T. M. Mitchell, "Machine learning: trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

[25] 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.

[26] A. Saied, R. E. Overill, and T. Radzik, "Artificial neural networks in the detection of known and unknown DDoS attacks: proof-of-concept," in *Proceedings of PAAMS 2014 International Workshops (PAAMS '14)*, Salamanca, Spain, 2014.

[27] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection," *Pattern Recognition Letters*, vol. 51, pp. 1–7, 2015.

[28] Z. Y. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu, "A system for denial-of-service attack detection based on multivariate correlation analysis," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 2, pp. 447–456, 2014.

[29] S. Yu, W. Zhou, W. Jia, S. Guo, Y. Xiang, and F. Tang, "Discriminating DDoS attacks from flash crowds using flow correlation coefficient," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 6, pp. 1073–1080, 2012.

[30] A. Wang, A. Mohaisen, W. Chang, and S. Chen, "Delving into Internet DDoS Attacks by Botnets: Characterization and Analysis," in *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN '15*, pp. 379–390, 2015.

[31] G. D. Kumar, C. V. G. Rao, M. K. Singh, and F. Ahmad, "Using Jpcap API to monitor, analyze, and report network traffic for DDoS attacks," in *Proceedings of the 14th International Conference on Computational Science and Its Applications, ICCSA '14*, vol. 39, p. 35, 2014.

[32] K. J. Singh, K. Thongam, and T. De, "Entropy-based application layer DDoS attack detection using artificial neural networks," *Entropy*, vol. 18, no. 10, 2016.

[33] A. Rukavitsyn, K. Borisenko, and A. Shorov, "Self-learning method for DDoS detection model in cloud computing," in *Proceedings of the 2017 IEEE Russia Section Young Researchers in Electrical and Electronic Engineering Conference, ElConRus '17*, pp. 544–547, 2017.

[34] H. Zhang, Z. Cai, Q. Liu et al., "A Survey on Security-Aware Measurement in SDN," *Security and Communication Networks*, vol. 2018, Article ID 2459154, 14 pages, 2018.

[35] J. Ashraf and S. Latif, "Handling intrusion and DDoS attacks in Software Defined Networks using machine learning techniques," in *Proceedings of the 2014 National Software Engineering Conference, NSEC '14*, pp. 55–60, Karachi, Pakistan, 2014.

[36] I. Mihai-Gabriel and P. Victor-Valeriu, "Achieving DDoS resiliency in a software defined network by intelligent risk assessment based on neural networks and danger theory," in *Proceedings of the 15th IEEE International Symposium on Computational Intelligence and Informatics, CINTI '14*, pp. 319–324, Hungary, 2014.

[37] Q. Yan, Q. Gong, and F. R. Yu, "Effective software-defined networking controller scheduling method to mitigate DDoS attacks," *IEEE Electronics Letters*, vol. 53, no. 7, pp. 469–471, 2017.

[38] T. Chin, X. Mountrouidou, X. Li, and K. Xiong, "Selective packet inspection to detect DoS flooding using software defined networking (SDN)," in *Proceedings of the 2015 35th IEEE International Conference on Distributed Computing Systems Workshops, ICDCSW '15*, pp. 95–99, 2015.

[39] N. Dayal and S. Srivastava, "Analyzing behavior of DDoS attacks to identify DDoS detection features in SDN," in *Proceedings of the 9th International Conference on Communication Systems and Networks, COMSNETS '17*, pp. 274–281, 2017.

[40] J. Ye, X. Cheng, J. Zhu, L. Feng, and L. Song, "A DDoS Attack Detection Method Based on SVM in Software Defined Network," *Security and Communication Networks*, vol. 2018, Article ID 9804061, 8 pages, 2018.

[41] 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.

[42] 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.

[43] 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, no. 1, pp. 20632–20640, 2018.

[44] 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.

[45] J. Cheng, X. Tang, and J. Yin, "A change-point DDoS attack detection method based on half interaction anomaly degree," *International Journal of Autonomous and Adaptive Communications Systems*, vol. 10, no. 1, pp. 38–54, 2017.

[46] J. R. Cheng, J. Yin, Y. Liu, Z. Cai, and C. Wu, "DDoS Attack Detection Using IP Address Feature Interaction," in *Proceedings of the 2009 International Conference on Intelligent Networking and Collaborative Systems (INCOS '09)*, pp. 113–118, 2009.

[47] J. Cheng, B. Zhang, J. Yin et al., "DDoS Attack Detection Using Three-State Partition Based on Flow Interaction," *Communications in Computer & Information Science*, vol. 29, no. 4, pp. 176–184, 2009.

[48] J. Cheng, J. Yin, Y. Liu, Z. Cai, and C. Wu, "Detecting Distributed Denial of Service Attack Based on Multi-feature Fusion," in *Security Technology*, vol. 58, pp. 132–139, 2009.

[49] J. R. Cheng, X. Tang, X. Zhu, and J. Yin, "Distributed denial of service attack detection based on IP Flow Interaction," in *Proceedings of the 2011 International Conference on E-Business and E-Government (ICEE '11)*, pp. 1–4, 2011.

[50] J. Cheng, J. Yin, and L. Yun, "Detecting Distributed Denial of Service Attack Based on Address Correlation Value," *Journal of Computer Research & Development*, vol. 46, no. 8, pp. 1334–1340, 2009.

[51] J. Cheng, J. Zhou, X. Tang, and J. Shi, "A distributed denial of service attack detection method based on time series prediction model," *Network Security Technology and Application*, vol. 10, pp. 71–89, 2016.

[52] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet, "Simple MKL," *Journal of Machine Learning Research*, vol. 9, no. 3, pp. 2491–2521, 2008.

[53] The Cooperative Association for Internet Data Analysis, *The Caida Ucsd DDoS Attack*, 2007, http://www.caida.org/data/passive/ddos-20070804_dataset.xml.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 2863793, 12 pages
https://doi.org/10.1155/2018/2863793

Research Article

Scheduling Parallel Intrusion Detecting Applications on Hybrid Clouds

Yi Zhang,[1] Jin Sun,[1] Zebin Wu,[1,2] Shuangyu Xie,[1] and Ruitao Xu[1]

[1] *School of Computer Science and Engineering, Nanjing University of Science and Technology, No. 200 Xiaolingwei Street, Nanjing, China*
[2] *Lianyungang E-Port Information Development Co. Ltd., Lianyungang, China*

Correspondence should be addressed to Yi Zhang; yzhang@njust.edu.cn

Received 29 April 2018; Accepted 5 July 2018; Published 16 October 2018

Academic Editor: Xuyun Zhang

Recently, Parallel Intrusion Detection (PID) becomes very popular and its procedure of the parallel processing is called a PID application (PIDA). This PIDA can be regarded as a Bag-of-Tasks (BoT) application, consisting of multiple tasks that can be processed in parallel. Given multiple PIDAs (i.e., BoT applications) to be handled, when the private cloud has insufficiently available resources to afford all tasks, some tasks have to be outsourced to public clouds with resource-used costs. The key challenge here is how to schedule tasks on hybrid clouds to minimize makespan given a limited budget. This problem can be formulated as an Integer Programming model, which is generally NP-Hard. Accordingly, in this paper, we construct an Iterated Local Search (ILS) algorithm, which employs an effective heuristic to obtain the initial task sequence and utilizes an insertion-neighbourhood-based local search method to explore better task sequences with lower makespans. A swap-based perturbation operator is adopted to avoid local optimum. With the objective of improving the proposal's efficiency without loss of any effectiveness, to calculate task sequences' objectives, we construct a Fast Task Assignment (FTA) method by integrating an existing Task Assignment (TA) method with an acceleration mechanism designed through theoretical analysis. Accordingly, the proposed ILS is named FILS. Experimental results show that FILS outperforms the existing best algorithm for the considered problem, considerably and significantly. More importantly, compared with TA, FTA achieves a 2.42x speedup, which verifies that the acceleration mechanism employed by FTA is able to remarkably improve the efficiency. Finally, impacts of key factors are also evaluated and analyzed, exhaustively.

1. Introduction

Cloud computing is a novel service-based paradigm that delivers large-scale computational resources in the form of a pay-as-you-go model. Recently, some innovative providers (e.g., VMware partnered with IBM) deliver hybrid cloud construction solutions (e.g., VMware Cloud Foundation (http://www.vmware.com/products/cloud-foundation.html)), which enable creating an extension of a private cloud on public clouds (as seen in Figure 1). As a result, administrators/programs (e.g., application/task schedulers) of the private cloud are able to use resources of public clouds seamlessly and transparently through unified tools/interfaces, since both the private cloud and its extension use the same virtualization technique provided by hybrid cloud construction solutions. In other words, these administrators/programs can perform actions on public clouds just like on their own private cloud. For example, an administrator wants to create an instance of a small VM type for executing a task. When the private cloud has insufficient resources, the administrator can create an instance of the same small VM type on a public cloud to handle the task.

Intrusion Detection (ID) has been widely used to protect computer/network systems from diverse attacks. Recently, taking advantage of distributed computing technologies (e.g., cloud computing), Parallel Intrusion Detection (PID) becomes very popular because of its high efficiency [1, 2]. PID is an ID whose critical part can be processed in parallel. For instance, in an ID using data mining methods, the data can be divided into multiple partitions. As a result, the entire mining job on all the data is divided into multiple subjobs (or called tasks), which only perform mining work on partitions

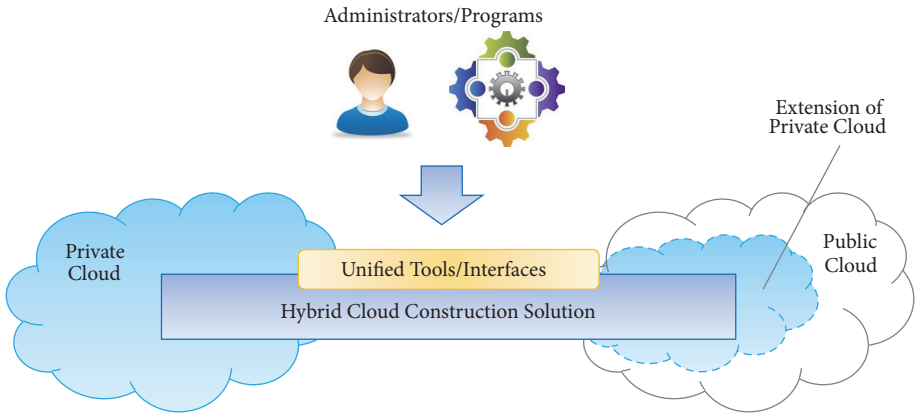


Figure 1: Administrators/programs of private cloud use resources of public clouds through unified tools/interfaces provided by hybrid cloud construction solutions [6].

and can accordingly be executed in parallel. Besides, an ID applying deep learning methods whose time-consuming training procedures can be performed in parallel is also a typical PID [3]. The procedure of the parallel processing in PID is called a PID application (PIDA) in this paper. Actually, these PIDAs can be considered as Bag-of-Tasks (BoT) applications, consisting of many independent tasks processed in parallel without synchronization or communication [4]. Theoretically, a cloud computing environment is the ideal platform to execute BoT applications, since it delivers cloud resources in a pay-as-you-go manner [5]. This is the reason why customers are willing to execute BoT applications on clouds.

Actually, customers may have private clouds, whose resources are free to use. Given multiple PIDAs to be processed for protecting different types of computer/network systems, these customers have to outsource some tasks to public clouds with additional costs, when their private clouds cannot afford all applications' tasks. Technically, tasks outsourced to public clouds can be achieved easily by the aforementioned hybrid cloud construction solutions. The key issue here is, given a limited budget, how to schedule tasks on hybrid clouds to minimize the total execution time (a.k.a. makespan).

This paper aims to schedule PIDAs on hybrid clouds, which is actually BoT Scheduling Problem (BTSP) with resource demands and budget constraints on hybrid clouds to minimize the makespan. In our previous work [6], this problem was formulated as an Integer Programming (IP) model, which is generally NP-Hard [7]. Accordingly, we also proposed an Effective Heuristic (EH) to solve the problem. EH starts from a task sequence generated by Longest Task First method (LTF) and uses a Task Assignment (TA) method to schedule all tasks in the obtained sequence to calculate the makespan. Although EH was verified to outperform the well-known RoundRobin method, we observe that the quality of the task schedule output by TA depends on its input task sequence, significantly. In order to further improve the schedule's quality, in this paper, we construct an Iterated Local Search (ILS) algorithm, which employs LTF to obtain the initial task sequence and utilizes an insertion-neighbourhood-based local search method to explore better task sequences with lower makespans. A swap-based perturbation operator is adopted to avoid local optimum. With the objective of improving the proposal's efficiency without loss of any effectiveness, instead of using TA to calculate task sequences' objectives, we construct a Fast TA (FTA) method by integrating TA with an acceleration mechanism designed through theoretical analysis. Accordingly, the proposed ILS is named as FILS. Experimental results show that FILS outperforms the existing best algorithm EH, considerably and significantly. More importantly, compared with TA, FTA achieves a 2.42x speedup and identical effectiveness, which verifies that the acceleration mechanism employed by FTA is able to remarkably improve the efficiency without losing effectiveness. The contributions of this paper are summarized below.

(i) We regard PIDA scheduling as BTSP, which can be formulated as an IP model.

(ii) We establish an effective algorithm FILS to solve the problem.

(iii) We propose an efficient heuristic FTA, which includes an acceleration method designed by theoretical analysis, to improve the efficiency.

(iv) We perform exhausted experiments to verify the proposed algorithms' effectiveness and efficiencies.

The rest of this paper is organized as follows. Section 2 discusses related works. Section 3 presents the problem description. The proposed FILS is described in Section 4. A full performance evaluation is shown in Section 5. Conclusions are given in Section 6 finally.

2. Related Works

In the literature, many efforts have been made to study BTSP in cloud environments, such as [6, 8–30]. Reference [4] presented a thoroughly comprehensive review on the state-of-the-art. As this paper considers budget-constrained BTSP

on hybrid clouds, we detail the two contributions and eight works related to budget constraints and hybrid clouds.

The following two contributions tackled BTSP with budget constraints in the environment of multiple public clouds. Reference [17] presented an algorithm for solving BTSP with either budget or deadline constraints. In their proposed algorithm, all the VM instances are initialized in the same type and iteratively replaced to be other different VM types. As a result, the objective (the total cost if a deadline constraint is considered or the makespan if a budget constraint is given) can be reduced without violating the constraint. Reference [18] proposed an approach to scale cloud resources for solving BTSP with both deadline and budget constraints while minimizing the total cost. They formulated the considered problem as an Integer Programming problem and developed a policy to determine the number of each VM type that can meet both constraints. In comparison with the problem tackled in these two papers, our considered problem has a different environment.

In the literature, many efforts have been made to study BTSP in cloud environments. Reference [4] presented a thoroughly comprehensive review on the state-of-the-art. As this paper considers BTSP on hybrid clouds, we detail the eight works related to hybrid clouds. Van den Bossche et al. [22] considered and formulated deadline-constrained BTSP as an IP and used the IBM CPLEX to obtain solutions. Later, the same authors [23] proposed two cost-efficient heuristics considering both computational and data-transferred costs. Similar heuristics are presented in [24] and a comprehensive analysis is performed through simulation experiments to show the effectiveness. Reference [26] tackled a similar deadline-constrained problem, in which physical machines on the private cloud are taken into account. The authors proposed a greedy heuristic that dispatches tasks to available physical machines on the private cloud and assigns them to public clouds while there are no available ones. Reference [27] solved deadline-constrained BTSP with the variation of tasks' runtimes. Thus, the authors constructed a method to estimate tasks' runtimes so that the scheduling plan can be updated accordingly. Reference [28] considered deadline-constrained BTSP on cloud federations (a term of hybrid clouds) and formulated it as an IP. The CPLEX was used to solve the problem with the results showing that the cloud federations benefit customers, compared with single cloud provider. Different from the three aforementioned works assuming that each task can be executed in an instance of any VM type, [29, 30] considered computation-intensive BTSP with resource demands and deadline constraints on hybrid clouds from the perspective of cloud providers with an objective of maximizing profit. Both papers employed Particle Swarm Optimization algorithms to solve the considered problems. Obviously, compared with the problems handled in the aforementioned eight papers (i.e., *deadline*-constrained BTSP with *cost minimization*), our considered problem (i.e., *budget*-constrained BTSP with *makespan minimization*) shares neither constraints nor objectives.

Our previous work [6] formulated the considered problem as an IP and proposed EH to solve it. EH uses LTF to generate the initial task sequence and employs TA to schedule all tasks in the obtained sequence to calculate the makespan. In this paper, we establish FILS that is demonstrated to outperform EH by experiments. As FTA is used rather than TA to calculate task sequences' objectives, we achieve a 2.42x speedup without loss of any effectiveness.

3. Problem Description

The formulation of the considered problem was given in our previous work [6]. For the completeness, we also introduce the formulation in this paper, briefly. We use $CP_0, CP_1, \ldots, CP_m$ to denote the $m + 1$ cloud providers. CP_0 represents the private cloud and the others are m public clouds. The private cloud provides k VM types $VM_1, VM_2, \ldots, VM_k$. Each VM type $VM_q (q = 1, 2, \ldots, k)$ has two performance parameters CPU_q and Mem_q that denote the number of CPUs and the amount of memory, respectively. When a task is outsourced to a public cloud, an instance of the VM type demanded by the task's application should be created to tackle the task on the public cloud. As aforementioned, technically, this procedure can be achieved easily by the above mentioned hybrid cloud construction solutions. Therefore, we can equivalently regard that the m public clouds also provide the k VM types. Additionally, $p_{qh} (q = 1, 2, \ldots, k,\ h = 1, 2, \ldots, m)$ represents the price (per time unit) for using an instance of VM_q provided by a public cloud $CP_h (h = 1, 2, \ldots, m)$. The private cloud's resources are free to use.

There are n applications. Each application $a_i (i = 1, 2, \ldots, n)$ requires a user-specified VM type. We use a binary variable x_{iq} to denote this relationship. $x_{iq} = 1$ means a_i demands VM_q; $x_{iq} = 0$ otherwise. Meanwhile, each application consists of T_i tasks $t_{i1}, t_{i2}, \ldots, t_{iT_i}$. Like most of existing contributions (such as [22–24, 29]), in our considered problem, one task is executed in one VM instance exclusively at a time, and each task is executed consecutively (i.e., no preemption is allowed). Those problems where one VM instance can run multiple tasks simultaneously or tasks can be executed preemptively are beyond the scope of this paper. A task $t_{ij} (i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, T_i)$ has a runtime $r_{ij} (i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, T_i)$. In other words, r_{ij} is the execution duration when t_{ij} is executed in an instance of the VM type required by its application.

Like [22–24, 29], setup times for VM instances (such as VM image loading, software installing, and network configuration) are regarded to be zero. Actually, there are cloud providers that are able to deliver VM instances in minutes (e.g., Amazon EC2) and even in seconds (e.g., qingcloud (https://www.qingcloud.com/)). However, in our considered problem, tasks' runtimes are longer than one hour at least. Setup times for VM instances are negligible compared with tasks' runtimes and are thus assumed to be zero. Additionally, though some traditional cloud providers (e.g., Amazon EC2) charge VM instances in hours, there are some innovative ones delivering resources in minutes (e.g., Microsoft Azure) or even in seconds (e.g., qingcloud and TecentCloud (https://www.qcloud.com/)). Obviously, users prefer using resources provided by these providers since they do not need to pay for an entire hour while only fraction

Table 1: Notations for problem description.

Notations	Indications	Notations	Indications
CP_0	Private cloud	CP_h	h-th public cloud ($h = 1, 2, \ldots, m$)
m	Total number of public clouds	VM_q	q-th VM type
k	Total number of VM types	CPU_q	Amount of CPU of VM_q
Mem_q	Amount of Memory of VM_q	p_{qh}	Price of VM_q provided by CP_h
a_i	i-th application	n	Total number of applications
T_i	Task number of a_i	c_i	completion time of a_i
t_{ij}	j-th task of a_i	r_{ij}	Runtime of t_{ij}
st_{ij}	Start time of t_{ij}	c_{ij}	completion time of t_{ij}
CPU^*/Mem^*	Capacity of CPU/Memory in CP_0	s	Time slot
B	Budget	x_{iq}	A variable $x_{iq} = 1$: a_i requires VM_q $x_{iq} = 0$: otherwise
y_{ijh}	A decision variable $y_{ijh} = 1$: t_{ij} is dispatched to CP_h $y_{ijh} = 0$: otherwise	z_{ijs}	A decision variable $z_{ijs} = 1$: t_{ij} is executed at slot s on CP_0 $z_{ijs} = 0$: otherwise

of this hour is used. Accordingly, in this paper, we regard resources are charged in seconds; i.e., the time unit is set as a second. As a result, it is not necessary to consider how to make use of an entire hour when we formulate the problem. The time axis is divided into several slots with the granularity of a second.

The private cloud CP_0 has limited number of available resources. The capacities of CPU and memory are denoted as CPU^* and Mem^*, respectively. In other words, for any time slot $s(s = 0, 1, \ldots)$, the amount of consumed resources cannot exceed CPU^* and Mem^*. All the m public clouds are regarded to have infinite resources. Let c_{ij} and c_i be the completion time of a task t_{ij} and the application's completion time, respectively. We have

$$c_i = \max\left\{c_{ij} \ \left(j = 1, 2, \ldots, T_i\right)\right\} \tag{1}$$

Accordingly, we can define the maximum of time slots S satisfying $S \geq \max\{c_i(i = 1, 2, \ldots, n)\}$. Let st_{ij} be the start time of the task t_{ij}. We can calculate c_{ij} by

$$c_{ij} = st_{ij} + r_{ij} \tag{2}$$

Let $y_{ijh}(i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, T_i,\ h = 0, 1, \ldots, m)$ and $z_{ijs}(i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, T_i,\ s = 0, 1, \ldots, S)$ be two decision variables. $y_{ijh} = 1$ means t_{ij} is dispatched to CP_h and $y_{ijh} = 0$ otherwise. $z_{ijs} = 1$ indicates t_{ij} is in execution at time slot s on CP_0 and $z_{ijs} = 0$ otherwise. Obviously, if a task t_{ij} is dispatched to the private cloud (i.e., $y_{ij0} = 1$), its start time $st_{ij} = \operatorname{argmin}\{s \mid z_{ijs} = 1\}$; otherwise, $st_{ij} = 0$ (like [29], we also focus on computation-intensive BoT applications which require tiny amounts of data and the short duration of transferring these tiny-amount data can be negligible compared with tasks' runtimes. As setup times for VM instances have been reasonably assumed to be zero, we can regard that tasks can be started at time slot 0 on public clouds). With the consideration of the variables defined above, the total cost $Cost$ can be calculated by

$$Cost = \sum_{i=1}^{n}\sum_{j=1}^{T_i}\sum_{h=1}^{m}\sum_{q=1}^{k} x_{iq} y_{ijh} r_{ij} p_{qh} \tag{3}$$

Let B be the budget and c_{max} be the objective makespan. All the notations for problem description are listed in Table 1. The problem can be formulated as an Integer Programming (IP) model given below.

Minimize the makespan c_{max}:

$$c_{max} = \max\left\{c_i \ \ (i = 1, 2, \ldots, n)\right\} \tag{4}$$

s.t.

$$Cost \leq B \tag{5}$$

$$\sum_{h=0}^{m} y_{ijh} = 1, \qquad i = 1, 2, \ldots, n,\ \ j = 1, 2, \ldots, T_i \tag{6}$$

$$\sum_{i=1}^{n}\sum_{q=1}^{k}\sum_{j=1}^{T_i} z_{ijs} x_{iq} CPU_q \leq CPU^*, \quad s = 0, 1, \ldots, S \tag{7}$$

$$\sum_{i=1}^{n}\sum_{q=1}^{k}\sum_{j=1}^{T_i} z_{ijs} x_{iq} Mem_q \leq Mem^*, \quad s = 0, 1, \ldots, S \tag{8}$$

Equation (4) is the objective. Equation (5) guarantees that the total cost is not beyond the budget. Equation (6) ensures that a task is assigned to a unique cloud. Equations (7) and (8) make sure that the consumption of CPU and memory of the private cloud at any time slot cannot exceed CPU^* and Mem^*, respectively.

```
Initialize the solution and regard it as the current solution;
Regard the initial solution as the best solution;
while (termination criterion is not met) do
    Perform a local search method on the current solution;
    Update the best solution if a new one is found;
    Perform a perturbation operator on the best solution and regard the obtained solution as the current solution;
end while
return The best solution;
```

Algorithm 1: Framework of general ILS.

4. Fast Iterated Local Search Algorithm (FILS)

The framework of general ILS is given in Algorithm 1. We can see that an ILS starts with an initial solution. If the termination criterion is not met, a local search method is performed on the current solution to explore new good solutions, and a perturbation operator is used to avoid local optimum. In this paper, we proposed a FILS, in which task sequences are considered solutions. LTF is used to generate the initial solution. An Insertion-Based Local Search Method (ILSM) is employed to explore better task sequences with lower makespans. A Swap-Based Perturbation Operator (SPO) is used to perturb the current solution. FTA is constructed to calculate makespans of task sequences. Details are given in this section.

4.1. Longest Task First (LTF). In our previous work [6], four heuristics including Highest Lowest Public Cost (LPC) First (HLPCF), Lowest LPC First (LLPCF), Longest Task First (LTF), and Shortest Task First (STF) were examined by experiments with the results showing that LTF is the best and helps the proposed EH to achieve good effectiveness. Accordingly, we also use LTF to generate the initial task sequence of FILS. LTF arranges all tasks by their runtimes in the nonascending order.

Meanwhile, it is worth introducing LPC, which will be used to describe FILS in Section 4.4. Like [6], the costs of executing tasks on public clouds are defined as public costs. A task's LPC can be defined as the minimum of all its public costs. Given a task t_{ij}, assume that the index of the VM type demanded by its application is u; i.e., $x_{iu} = 1$. The task's LPC can be calculated by

$$LPC_{ij} = \min\left\{r_{ij}p_{uh} \ \ (h = 1, 2, \ldots, m)\right\} \tag{9}$$

Accordingly, the corresponding public cloud is called the task's "Ideal" Public Cloud (IPC). Obviously, the index of a task's IPC should meet

$$IPC_{ij} = \operatorname{argmin}\left\{h \mid r_{ij}p_{uh}\right\} \tag{10}$$

4.2. Insertion-Based Local Search Method. Given a task sequence ts with the length T, ILSM first regards it as a temp task sequence $temp$. Then, ILSM removes the $u(u = 1, 2, \ldots, T)$-th task from $temp$ and reinserted this task to the left $temp$ at each position except the task's original one. As a result, $T - 1$ new task sequences are generated and evaluated by TA/FTA (corresponding to CILS/FILS) to calculate their makespans. If one generated task sequence gts has a lower makespan than ts, both ts and $temp$ are set as gts. Afterwards, ILSM processes the $(u + 1)$-th task in $temp$ in the same way. After the T-th task has been processed, ILSM terminates. Obviously, the complexity of ILSM is $O(T^2 \cdot O(TA/FTA))$, in which $O(TA/FTA)$ represents the complexity of TA/FTA.

We use an example to clarify the procedure of ILSM. In this example, we have one application with three tasks. Given a task sequence $ts = (t_{11}, t_{12}, t_{13})$ with makespan 10, ILSM first sets $temp \longleftarrow ts$. Then, ILSM removes the first task t_{11} and reinserted it to $temp$ at each position except the first one. As a result, two new task sequences $ts_1 = (t_{12}, t_{11}, t_{13})$ and $ts_2 = (t_{12}, t_{13}, t_{11})$ are generated. Assume their makespans are 12 and 9, respectively. In other words, ts_2 gets a lower makespan than ts and we set $temp \longleftarrow ts \longleftarrow ts_2$. Afterwards, the second task in $temp$ (i.e., t_{13}) is removed and reinserted. The two obtained task sequences are $ts_3 = (t_{13}, t_{12}, t_{11})$ and $ts_4 = (t_{12}, t_{11}, t_{13})$ with the makespans 8 and 12, respectively. Accordingly, we set $temp \longleftarrow ts \longleftarrow ts_3$. Finally, the third task in $temp$ (i.e., t_{11}) is removed and reinserted. The two generated task sequences are $ts_5 = (t_{11}, t_{13}, t_{12})$ and $ts_6 = (t_{13}, t_{11}, t_{12})$ with the makespans 6 and 14, respectively. In other words, ts_5 obtains a lower makespan than ts does. Consequently, we set $temp \longleftarrow ts \longleftarrow ts_5$ and ts is the final result of ILSM.

4.3. Swap-Based Perturbation Operator. SPO is used to help the two ILSs to avoid local optimum. It iterates the following procedure l rounds: randomly select a pair of tasks in a given task sequence ts and swap them. Obviously, this task swap operator is able to adjust the relative orders of tasks partially and tries to make the two ILSs jump out from local optimum if they have already been trapped in. l is a very important parameter and will be determined by an experiment in Section 5.2. It is obvious that the complexity of SPO is $O(l)$.

4.4. Fast Task Assignment Method. FTA is developed by integrating an acceleration mechanism with TA without loss of any effectiveness. In TA (details of TA can be seen in our previous work [6]), we can find that the makespan corresponding to the case that the task is assigned to the private cloud (i.e., c_{max}^{ij0}) is first calculated and then compared with the one corresponding to the case that the task is

Input: a task sequence ts
Output: makespan c_{max}
1: Set $S \longleftarrow MAX$, $c_{max} \longleftarrow 0$, $budget \longleftarrow B$;
2: Set $cpu_s \longleftarrow CPU^*$ and $mem_s \longleftarrow Mem^*$ for each time slot $s \in \{0, 1, \ldots, S\}$;
3: **for** (each task t_{ij} in ts) **do**
4: Set $st_{ij} \longleftarrow 0$ and $AssignedToPrivateCloud \longleftarrow FALSE$;
5: **while** ($TRUE$) **do**
6: Set $AssignedToPrivateCloud \longleftarrow TRUE$;
7: **for** (each time slot $s \in \{st_{ij}, st_{ij} + 1, \ldots, st_{ij} + r_{ij} - 1\}$) **do**
8: **if** ($\sum_{q=1}^{k} x_{iq} CPU_q > cpu_s$ OR $\sum_{q=1}^{k} x_{iq} Mem_q > mem_s$) **then**
9: Set $AssignedToPrivateCloud \longleftarrow FALSE$ and break;
10: **end if**
11: **end for**
12: **if** ($AssignedToPrivateCloud$) **then**
13: Calculate $c_{max}^{ij0} \longleftarrow \max\{c_{max}, st_{ij} + r_{ij}\}$;
14: break;
15: **else**
16: Set $st_{ij} \longleftarrow st_{ij} + 1$;
17: **if** ($st_{ij} > \max\{c_{max} - r_{ij}, 0\}$ AND $budget \geq LPC_{ij}$) **then** ▷ **The acceleration mechanism using Theorem 1 is utilized**
18: Set $c_{max}^{ij0} \longleftarrow MAX$ and break;
19: **end if**
20: **end if**
21: **end while**
22: Calculate $c_{max}^{ij(IPC_{ij})} \longleftarrow \max\{c_{max}, r_{ij}\}$;
23: **if** ($LPC_{ij} \leq budget$ AND $c_{max}^{ij(IPC_{ij})} < c_{max}^{ij0}$) **then**
24: Set $c_{max} \longleftarrow c_{max}^{ij(IPC_{ij})}$ and $budget \longleftarrow budget - LPC_{ij}$;
25: **else**
26: Set $c_{max} \longleftarrow c_{max}^{ij0}$;
27: **for** (each time slot $s \in \{st_{ij}, st_{ij} + 1, \ldots, st_{ij} + r_{ij} - 1\}$) **do**
28: Update $cpu_s \longleftarrow cpu_s - \sum_{q=1}^{k} x_{iq} CPU_q$;
29: Update $mem_s \longleftarrow mem_s - \sum_{q=1}^{k} x_{iq} Mem_q$;
30: **end for**
31: **end if**
32: **end for**
33: **return** c_{max};

Algorithm 2: FTA.

dispatched to its IPC ($c_{max}^{ij(IPC_{ij})}$). The calculation of c_{max}^{ij0} is time-consuming because we need to determine the task's start time first. However, actually, we do not need to calculate c_{max}^{ij0} for some special cases, since the following theorem is true.

Theorem 1. *In TA, given a task t_{ij} to be scheduled, if $st_{ij} > \max\{c_{max} - r_{ij}, 0\}$ and $budget \geq LPC_{ij}$, the task will be assigned to its IPC.*

Proof. For the case that $c_{max} > r_{ij}$, according to Theorem 1, we have $c_{max}^{ij(IPC_{ij})} = c_{max}$. Additionally, due to $st_{ij} > \max\{c_{max} - r_{ij}, 0\} = c_{max} - r_{ij} > 0$, $st_{ij} + r_{ij} > c_{max}$ is true. And, according to Theorem 1 given in our previous work [6], we have $c_{max}^{ij0} = \max\{c_{max}, st_{ij} + r_{ij}\}$. So, $c_{max}^{ij0} = st_{ij} + r_{ij} > c_{max} = c_{max}^{ij(IPC_{ij})}$. As a result, the task will be assigned to its IPC.

For the case that $c_{max} \leq r_{ij}$, according to Theorem 1, we have $c_{max}^{ij(IPC_{ij})} = r_{ij}$. Additionally, due to $st_{ij} > \max\{c_{max} - r_{ij}, 0\}$, $st_{ij} > 0$ is true. Because of $c_{max} \leq r_{ij}$ and $st_{ij} > 0$, we have $c_{max}^{ij0} = \max\{c_{max}, st_{ij} + r_{ij}\} = st_{ij} + r_{ij} > r_{ij} = c_{max}^{ij(IPC_{ij})}$. Therefore, the task will be assigned to its IPC as well. □

Theorem 1 shows that if $st_{ij} > \max\{c_{max} - r_{ij}, 0\}$ and $budget \geq LPC_{ij}$, the task should be assigned to its IPC without respect to c_{max}^{ij0}. In other words, we do not need to calculate c_{max}^{ij0} for these special cases and the efficiency can thus be improved. Accordingly, we construct an acceleration mechanism that uses Theorem 1 to discover these special cases. As a result, FTA is established by integrating TA with this acceleration mechanism and is described in Algorithm 2.

FTA uses the aforementioned acceleration mechanism (Lines 17-19) to improve the efficiency without loss of effectiveness, which is ensured by Theorem 1. If the two conditions in Line 17 are true, c_{max}^{ij0} is set as a sufficiently large value MAX (so that $MAX > c_{max}^{ij(IPC_{ij})}$) and the "while" loop in Line 5 is

Input: all applications' tasks
Output: makespan c_{max}
1: Determine all tasks' IPCs and calculate their LPCs;
2: Use LTF to generate a task sequence ts;
3: $best \longleftarrow ts$ and $c_{max} \longleftarrow$ FTA(ts);
4: Set $temp \longleftarrow ts$, $c_{max}^{temp} \longleftarrow c_{max}^{ts} \longleftarrow c_{max}$;
5: **while** (termination criterion is not met) **do**
6: $Improved \longleftarrow TRUE$;
7: **while** ($Improved$) **do**
8: $Improved \longleftarrow FALSE$;
9: **for** ($u \longleftarrow 1$ to T) **do**
10: Remove the u-th task in $temp$;
11: Reinsert the removed task in $temp$ at each position except the original one;
12: **for** (each generated task sequence gts) **do**
13: $c_{max}^{gts} \longleftarrow$ FTA(gts);
14: **if** ($c_{max}^{gts} < c_{max}^{ts}$) **then**
15: $Improved \longleftarrow TRUE$;
16: Set $temp \longleftarrow ts \longleftarrow gts$ and $c_{max}^{temp} \longleftarrow c_{max}^{ts} \longleftarrow c_{max}^{gts}$;
17: **end if**
18: **end for**
19: **end for**
20: **end while**
21: **if** ($c_{max}^{ts} < c_{max}$) **then**
22: Set $best \longleftarrow ts$ and $c_{max} \longleftarrow c_{max}^{ts}$;
23: **end if**
24: Set $temp \longleftarrow best$;
25: **for** ($u \longleftarrow 1$ to l) **do**
26: Select a pair of tasks in $temp$ randomly and swap them;
27: **end for**
28: $c_{max}^{temp} \longleftarrow$ FTA($temp$);
29: Set $ts \longleftarrow temp$ and $c_{max}^{ts} \longleftarrow c_{max}^{temp}$;
30: **end while**
31: **return** c_{max};

Algorithm 3: FILS.

terminated. As a result, the two conditions in Line 23 are met; i.e., the task is assigned to its IPC. Obviously, the complexity of FTA is identical to that of TA. Nevertheless, compared with TA, FTA obtains much better efficiency (details can be seen in Section 5.3).

4.5. Description of FILS. Let *best* and c_{max} be the current best found task sequence and its makespan, respectively. Based on the aforementioned LTF, ILSM, SPO, and FTA, we can describe the proposed FILS in Algorithm 3. FILS starts from a task sequence generated by LTF (Line 2). ILSM (Lines 9-19) is iterated until no improvement is obtained (see the condition in Line 7). If a new best task sequence is found, *best* and c_{max} are accordingly updated (Lines 21-23). Afterwards, SPO is invoked to perturb *best* so that FILS can jump out from local optimum (Lines 24-27). Task sequences' makespans are calculated by FTA (Lines 3, 13, and 28).

5. Experimental Results

Following most of existing contributions, we use simulation experiments to evaluate algorithms' performance.

5.1. Testing Instances. We use the testing instances given in our previous work [6]. For the completeness of this paper, we describe these testing instances as follows. Three different Regions (us-east, us-west, and eu-east) of Amazon EC2 and GoGrid are regarded as four public clouds, and 7 different VM types are considered. The configurations and prices are described in Table 2. Note that the prices in Table 2 are shown per hour and we will convert them to values per second when we implement algorithms. The private cloud also provides the same seven VM types.

In order to explore the compared algorithms' performance on problems of different sizes, we consider the total number of tasks (i.e., T) as the problem size factor and construct a Testing Instance Set (TIS), which contains 3 groups with $T \in \{20, 50, 100\}$. Each group contains 3 same-sized subgroups corresponding to 3 different problem types: Small Application Type (SAT), Medium Application Type (MAT), and Large Application Type (LAT). The SAT problem has many small applications that have only a few tasks, while the LAT problem has a few large applications that include lots of tasks. The MAT problem is in between them. So, multiple types of problems are taken into account in this experiment.

Table 2: VM types provided by GoGrid and Amazon EC2.

VM Type	CPU	Memory	Prices (GoGrid)	Prices (EC2 us-east)	Prices (EC2 us-west)	Prices (EC2 eu-east)
t2.micro	1	1	0.02	0.013	0.017	0.014
t2.small	1	2	0.03	0.026	0.034	0.028
t2.medium	2	4	0.06	0.052	0.068	0.056
m3.medium	1	3.75	0.09	0.070	0.077	0.077
m3.large	2	7.5	0.17	0.140	0.154	0.154
m3.xlarge	4	15	0.34	0.280	0.308	0.308
m3.2xlarge	8	30	0.68	0.560	0.616	0.616

Table 3: Instances' sizes in TIS.

Instance Type	T	n	T	n	T	n
SAT	20	$[1, 0.8T]$	50	$[1, 0.8T]$	100	$[1, 0.8T]$
MAT	20	$[1, 0.5T]$	50	$[1, 0.5T]$	100	$[1, 0.5T]$
LAT	20	$[1, 0.2T]$	50	$[1, 0.2T]$	100	$[1, 0.2T]$

Each subgroup has 10 different instances and there are 90 instances in total. In order to generate instances of SAT, MAT, and LAT, the application number n is set as a random integer uniformly distributed within intervals $[1, 0.8T]$, $[1, 0.5T]$, and $[1, 0.2T]$, respectively. Each task is attributed to the n applications with the same probability $1/n$, separately. TIS is summarized in Table 3. The runtime of each task is an integer uniformly distributed in $[1 \times 3600, 24 \times 3600]$ (i.e., from one hour to one day). The VM type required by each application is randomly selected from the 7 considered VM types.

Let VM_b be the best one among all VM types. The budget is set by (11), where λ is a "Budget Factor" used to adjust the budget so that algorithms' performance with different budgets can be explored. According to (11), higher budgets are for larger testing instances.

$$B = T \times p_b \times \lambda \tag{11}$$

The private cloud's available CPU and memory capacities (i.e., CPU^* and Mem^*) are set by (12) and (13), respectively, in which ρ is a "Capacity Factor" used to adjust the two types of resources' capacities so that algorithms' performance with different capacities can be investigated. According to (12) and (13), the private cloud has more available resources for larger testing instances.

$$CPU^* = T \times CPU_b \times \rho \tag{12}$$

$$Mem^* = T \times Mem_b \times \rho \tag{13}$$

5.2. Parameter Determination. The proposed FILS has a parameter, i.e., the number of pairs of swapped tasks (i.e., l) in the SPO. This parameter is determined by experiments in this section. In order to use some statistical methods (e.g., the well-known multifactor analysis of variance (ANOVA)) to evaluate algorithms' performance, we set $\lambda \in \{1, 5, 10\}$ and $\rho \in \{0.05, 0.1, 0.15\}$, respectively. So, there are 9 factors' combinations. Each algorithm is tested on TIS with all possible factors' combinations. All algorithms are implemented in Java and run on the same PC with Dual Core Pentium (R) 3.10 GHz CPU and 4GB Memory. The termination criterion of FILS is set as the maximal number of iterations 100. Relative Error (RE) defined by (14) is adopted to evaluate the performance.

$$RE = \frac{\left(\sum_{r=1}^{R} \left(\left(c_r - c_{LB}\right)/c_{LB}\right)\right)}{R} \times 100\% \tag{14}$$

For each instance, c_r denotes the obtained makespan returned by an algorithm in the r-th replication. R is the total number of replications. c_{LB} represents the makespan's lower-bound, which can be obtained while both the private cloud's resource capacity and budget constraints are assumed to be relaxed. In other words, the private cloud's resource capacity and budget are assumed to be infinite. In this situation, all tasks can be executed in parallel. Accordingly, c_{LB} can be calculated by

$$c_{LB} = \max\left\{r_{ij} \ \left(i = 1, 2, \ldots, n, \ j = 1, 2, \ldots, T_i\right)\right\} \tag{15}$$

Smaller REs indicate better effectiveness since the same c_{LB} is used. Based on the RE for each instance, we further use ANOVA to check whether the differences in the observed average REs are statistically significant. Nonoverlapping confidence intervals between any two pairs of plotted averages mean that the observed differences in such averages are statistically significant at the indicated confidence level. FILS is executed 5 replications (i.e., $R = 5$) since it is metaheuristics including randomness. In order to determine the value of l, we set $l \in \{1, 2, 3, 4, 5\}$. In other words, l has 5 candidates. FILS is tested on TIS with the 5 candidates and the aforementioned two factors' (λ and ρ) 9 combinations. The plot of mean REs and 95% confidence LSD intervals for compared algorithms is given in Figure 2, where FILS1-FILS5 represent FILS with $l = 1, 2, 3, 4, 5$, respectively. Figure 2 shows that the mean REs of FILS with $l = 1, 2, 3, 4, 5$ are 1.083, 0.995, 1.120, 1.157, and 1.189, respectively. The parameter l is thus set as 2 in FILS.

5.3. Performance Evaluation. In order to evaluate FILS's performance, we generate another New TIS (NTIS) by using the

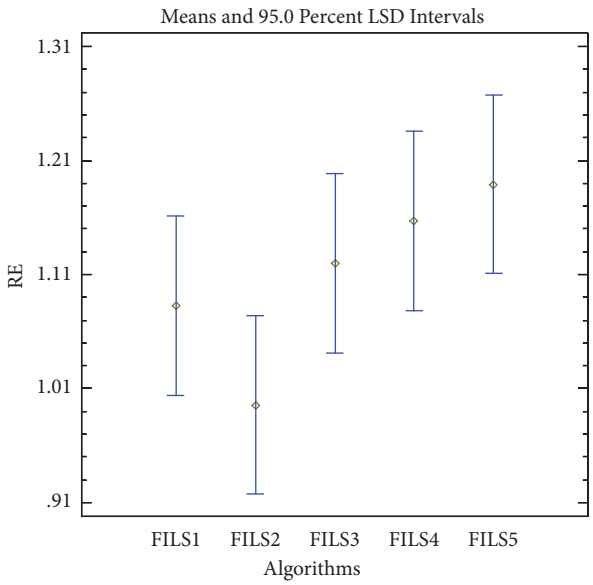


Figure 2: Plot of mean REs and LSD intervals for compared algorithms to determine the l of FILS.

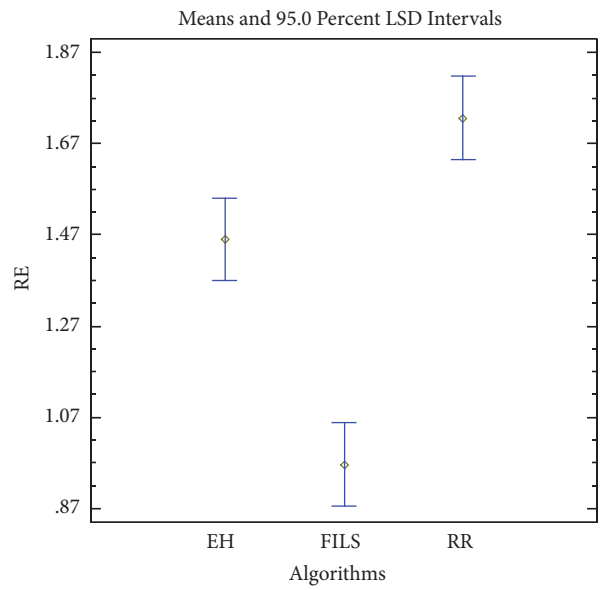


Figure 3: Plot of mean REs and LSD intervals for the compared algorithms.

same rules described in Section 5.1. Though the same rules are used, NTIS is different from TIS used in Section 5.2. EH [6] is the existing best algorithm for the considered problem and is thus regarded to be the baseline, accordingly. Meanwhile, the well-known RoundRobin (RR) is also adopted. In RR, the initial task sequence is generated randomly and each task in the obtained task sequence is assigned to all clouds randomly without violating the private cloud's resource capacity and the budget constraints. Same as those in Section 5.2, the two factors are set as $\lambda \in \{1, 5, 10\}$ and $\rho \in \{0.05, 0.1, 0.15\}$, respectively, whereas the termination criterion of FILS is set as the maximal number of iterations 100. Additionally, RR and FILS are executed 5 replications (i.e., $R = 5$). The plot of mean REs and LSD intervals (95% confidence level) for the compared algorithms is given in Figure 3.

In Figure 3, we can see that the mean REs of EH, FILS, and RR are 1.460, 0.966, and 1.727, respectively. This conclusion shows that FILS outperforms EH that is better than RR, remarkably and significantly. On the side of efficiency, EH and RR consume similar computation times for each testing instance. Their average computation times across over all the testing instances are in the level of tens of milliseconds, whereas that of FILS is in the level of tens of seconds. In other words, compared with the computation times of FILS, those of EH and RR can be negligible. Accordingly, we do not evaluate the efficiencies of all the three compared algorithms together. Instead, with the objective of evaluating the acceleration mechanism employed by FTA, we compare FILS with a Common ILS (CILS) that is the same as FILS except for using TA to calculate task sequences' makespans. For this purpose, we define the Normalized Efficiency (NE) as follows:

$$NE = \frac{t}{t_{baseline}} \tag{16}$$

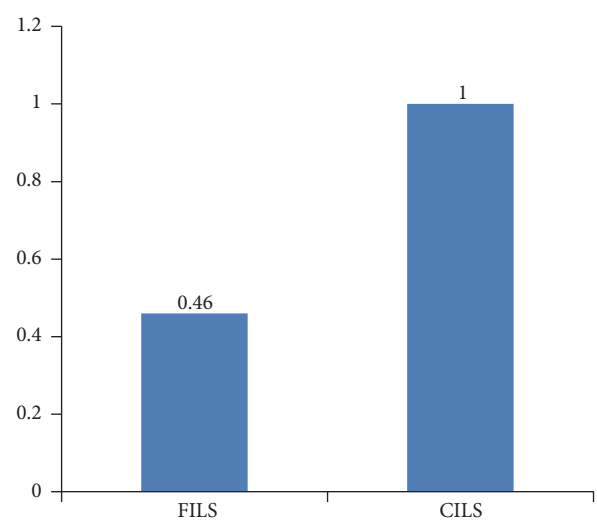


Figure 4: Plot of mean NEs for the two ILSs.

For each instance, t denotes an algorithm's computation time, and $t_{baseline}$ represents the computation time of the baseline which is selected from compared algorithms. Obviously, a lower NE indicates a better efficiency. Without loss of generality, we select CILS as the baseline in this experiment. As both algorithms are executed R replications on each instance, t and $t_{baseline}$ are the means of the R obtained computation times, while we calculate NE for each instance. The mean NEs of the two ILSs are presented in Figure 4, which shows that their mean NEs are 0.46 and 1.0 (CILS is regarded as the baseline), respectively. This conclusion denotes that FILS is much more efficient than CILS, indicating the acceleration mechanism employed by FTA improves the efficiency, considerably. Moreover, we can calculate the speedup achieved by FILS through calculating

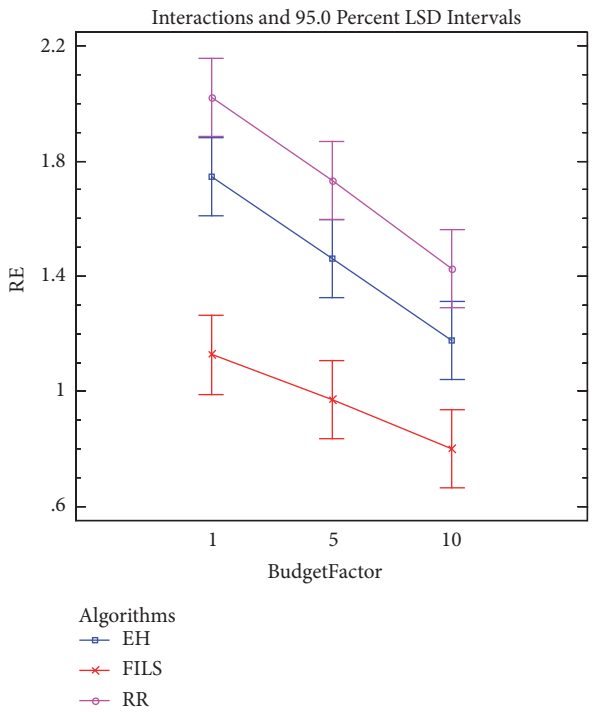


Figure 5: Plot of mean REs and LSD intervals for the interactions between the types of algorithms and the Budget Factor λ.

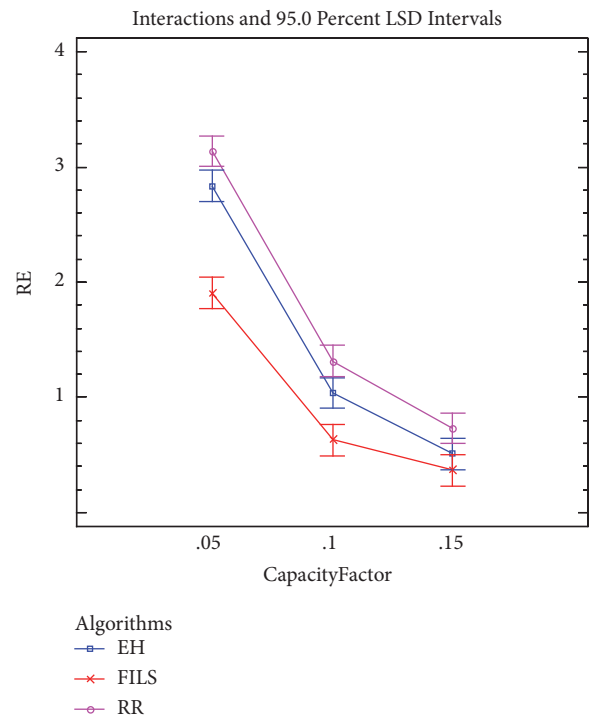


Figure 6: Plot of mean REs and LSD intervals for the interactions between the types of algorithms and the Capacity Factor ρ.

the mean of all instances' speedups, each of which is defined as the reciprocal of NE (i.e., $1/NE$). Accordingly, the speedup obtained by FILS is 2.42. In other words, FILS is 2.42x faster than CILS.

In order to investigate impacts of the two key factors (i.e., the Budget Factor λ and the Capacity Factor ρ), we present the plots of mean REs and LSD intervals (95% confidence level) for the interactions between the types of algorithms and the two factors in Figures 5 and 6, respectively. Figures 5/6 illustrates that mean REs of the three compared algorithms decrease while λ/ρ increases. Actually, this conclusion is reasonable. A larger λ indicates a higher budget, and more tasks can be executed on public clouds in parallel. A larger ρ denotes bigger resource capacity of the private cloud. Accordingly, more tasks can be executed on the private cloud in parallel when the private cloud has more resources. Therefore, we can conclude that the parallelism of task execution can be improved when the two factors are set as large values.

6. Conclusions

This paper schedules Parallel Intrusion Detection Applications (PIDAs) on hybrid clouds to minimize the makespan with the constraints of resource demands and budget. As this problem is NP-Hard, we construct a Fast Iterated Local Search (FILS) algorithm, which employs an effective heuristic to obtain the initial task sequence and utilizes an insertion-neighbourhood-based local search method to explore better task sequences with lower makespans. A swap-based perturbation operator is adopted to avoid local optimum. A Fast Task Assignment (FTA) method is developed by integrating an existing Task Assignment (TA) method with an acceleration mechanism designed through theoretical analysis and is used to calculate task sequences' objectives. Experimental results show that FILS outperforms the existing best algorithm for the considered problem, considerably and significantly. More importantly, compared with TA, FTA achieves a 2.42x speedup, which verifies that the acceleration mechanism employed by FTA is able to remarkably improve the efficiency. Impacts of the two key factors (the Budget Factor and the Capacity Factor) are also investigated with the results showing that the parallelism of task execution can be improved when the two factors are set as large values.

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 is supported by the National Natural Science Foundation of China [Grant nos. 71501096 and 61502234], by Natural Science Foundation of Jiangsu Province [Grant no. BK20150785], by China Postdoctoral Science Foundation

[Grant no. 2015M581801], and by the Fundamental Research Funds for the Central Universities [Grant no. 30916011325].

References

[1] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, 2013.

[2] P. Mishra, E. S. Pilli, V. Varadharajan, and U. Tupakula, "Intrusion detection techniques in cloud environment: A survey," *Journal of Network and Computer Applications*, vol. 77, pp. 18–47, 2017.

[3] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[4] L. Thai, B. Varghese, and A. Barker, "A survey and taxonomy of resource optimisation for executing bag-of-task applications on public clouds," *Future Generation Computer Systems*, vol. 82, pp. 1–11, 2018.

[5] R. Costa, F. Brasileiro, G. Lemos, and D. Sousa, "Analyzing the impact of elasticity on the profit of cloud computing providers," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1777–1785, 2013.

[6] Y. Zhang, J. Sun, and Z. Wu, "An heuristic for bag-of-tasks scheduling problems with resource demands and budget constraints to minimize makespan on hybrid clouds," in *Proceedings of the 5th International Conference on Advanced Cloud and Big Data, CBD 2017*, pp. 39–44, China, August 2017.

[7] P. Brucker, *Scheduling Algorithms*, Springer-Verlag, 2004.

[8] K. H. Kim, R. Buyya, and J. Kim, "Power aware scheduling of bag-of-tasks applications with deadline constraints on DVS-enabled clusters," in *Proceedings of the 7th IEEE International Symposium on Cluster Computing and the Grid (CCGrid '07)*, pp. 541–548, Rio De Janeiro, Brazil, May 2007.

[9] R. N. Calheiros and R. Buyya, "Energy-efficient scheduling of urgent bag-of-tasks applications in clouds through DVFS," in *Proceedings of the 2014 6th IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2014*, pp. 342–349, Singapore, Singapore, December 2014.

[10] G. Terzopoulos and H. D. Karatza, "Bag-of-task scheduling on power-aware clusters using a DVFS-based mechanism," in *Proceedings of the 28th IEEE International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2014*, pp. 833–840, Phoenix, Ariz, USA, May 2014.

[11] Y. Zhang, Y. Wang, and C. Hu, "CloudFreq: Elastic energy-efficient bag-of-tasks scheduling in DVFS-enabled clouds," in *Proceedings of the 21st IEEE International Conference on Parallel and Distributed Systems, ICPADS 2015*, pp. 585–592, Melbourne, Australia, December 2015.

[12] A.-M. Oprescu and T. Kielmann, "Bag-of-tasks scheduling under budget constraints," in *Proceedings of the 2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom '10)*, pp. 351–359, IEEE, Indianapolis, Ind, USA, December 2010.

[13] A.-M. Oprescu, T. Kielmann, and H. Leahu, "Stochastic tail-phase optimization for bag-of-tasks execution in clouds," in *Proceedings of the 2012 IEEE/ACM 5th International Conference on Utility and Cloud Computing, UCC 2012*, pp. 204–208, Chicago, Ill, USA, November 2012.

[14] M. Vasile, F. Pop, R. Tutueanu, and V. Cristea, "HySARC2: Hybrid Scheduling Algorithm Based on Resource Clustering in Cloud Environments," in *Algorithms and Architectures for Parallel Processing*, vol. 8285 of *Lecture Notes in Computer Science*, pp. 416–425, Springer International Publishing, Cham, Switzerland, 2013.

[15] J. O. Gutierrez-Garcia and K. M. Sim, "A family of heuristics for agent-based elastic Cloud bag-of-tasks concurrent scheduling," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1682–1699, 2013.

[16] L. Thai, B. Varghese, and A. Barker, "Executing bag of distributed tasks on the cloud: Investigating the trade-offs between performance and cost," in *Proceedings of the 6th IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2014*, pp. 400–407, Singapore, Singapore, December 2014.

[17] L. Thai, B. Varghese, and A. Barker, "Budget constrained execution of multiple bag-of-tasks applications on the cloud," in *Proceedings of the 8th IEEE International Conference on Cloud Computing, CLOUD 2015*, pp. 975–980, New York, NY, USA, July 2015.

[18] M. Mao, J. Li, and M. Humphrey, "Cloud auto-scaling with deadline and budget constraints," in *Proceedings of the 11th IEEE/ACM International Conference on Grid Computing, Grid 2010*, pp. 41–48, Brussels, Belgium, October 2010.

[19] M. H. Farahabady, Y. C. Lee, and A. Y. Zomaya, "Non-clairvoyant assignment of bag-of-tasks applications across multiple clouds," in *Proceedings of the 13th International Conference on Parallel and Distributed Computing, Applications, and Technologies, PDCAT 2012*, pp. 423–428, Beijing, China, December 2012.

[20] I. A. Moschakis and H. D. Karatza, "Multi-criteria scheduling of Bag-of-Tasks applications on heterogeneous interlinked clouds with simulated annealing," *The Journal of Systems and Software*, vol. 101, pp. 1–14, 2015.

[21] I. A. Moschakis and H. D. Karatza, "A meta-heuristic optimization approach to the scheduling of bag-of-tasks applications on heterogeneous clouds with multi-level arrivals and critical jobs," *Simulation Modelling Practice and Theory*, vol. 57, pp. 1–25, 2015.

[22] R. Van Den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads," in *Proceedings of the 3rd IEEE International Conference on Cloud Computing (CLOUD '10)*, pp. 228–235, Miami, Fla, USA, July 2010.

[23] R. Van Den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-efficient scheduling heuristics for deadline constrained workloads on hybrid clouds," in *Proceedings of the 2011 3rd IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2011*, pp. 320–327, Greece, December 2011.

[24] R. van den Bossche, K. Vanmechelen, and J. Broeckhove, "Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 973–985, 2013.

[25] M. Malawski, K. Figiela, and J. Nabrzyski, "Cost minimization for computational applications on hybrid cloud infrastructures," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1786–1794, 2013.

[26] B. Wang, Y. Song, Y. Sun, and J. Liu, "Managing Deadline-constrained Bag-of-Tasks Jobs on Hybrid Clouds," in *Proceedings of the 24th High Performance Computing Symposium*, Pasadena, Calif, USA, 2016.

[27] V. Pelaez, A. Campos, D. F. Garcia, and J. Entrialgo, "Autonomic scheduling of deadline-constrained bag of tasks in hybrid clouds," in *Proceedings of the 2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*, pp. 1–8, Montreal, QC, Canada, July 2016.

[28] S. Abdi, L. PourKarimi, M. Ahmadi, and F. Zargari, "Cost minimization for deadline-constrained bag-of-tasks applications in federated hybrid clouds," *Future Generation Computer Systems*, vol. 71, pp. 113–128, 2017.

[29] X. Zuo, G. Zhang, and W. Tan, "Self-adaptive learning pso-based deadline constrained task scheduling for hybrid iaas cloud," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 564–573, 2014.

[30] Y. Zhang and J. Sun, "Novel efficient particle swarm optimization algorithms for solving QoS-demanded bag-of-tasks scheduling problems with profit maximization on hybrid clouds," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 21, Article ID e4249, 2017.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 3486529, 13 pages
https://doi.org/10.1155/2018/3486529

Research Article

A Privacy Protection Model of Data Publication Based on Game Theory

Li Kuang,[1] Yujia Zhu,[1] Shuqi Li,[1] Xuejin Yan,[1] Han Yan,[1] and Shuiguang Deng[2]

[1] *School of Software, Central South University, Changsha 410075, China*
[2] *College of Computer Science, Zhejiang University, Hangzhou 310027, China*

Correspondence should be addressed to Shuiguang Deng; dengsg@zju.edu.cn

Received 17 August 2018; Accepted 23 September 2018; Published 14 October 2018

Guest Editor: Xuyun Zhang

With the rapid development of sensor acquisition technology, more and more data are collected, analyzed, and encapsulated into application services. However, most of applications are developed by untrusted third parties. Therefore, it has become an urgent problem to protect users' privacy in data publication. Since the attacker may identify the user based on the combination of user's quasi-identifiers and the fewer quasi-identifier fields result in a lower probability of privacy leaks, therefore, in this paper, we aim to investigate an optimal number of quasi-identifier fields under the constraint of trade-offs between service quality and privacy protection. We first propose modelling the service development process as a cooperative game between the data owner and consumers and employing the Stackelberg game model to determine the number of quasi-identifiers that are published to the data development organization. We then propose a way to identify when the new data should be learned, as well, a way to update the parameters involved in the model, so that the new strategy on quasi-identifier fields can be delivered. The experiment first analyses the validity of our proposed model and then compares it with the traditional privacy protection approach, and the experiment shows that the data loss of our model is less than that of the traditional k-anonymity especially when strong privacy protection is applied.

1. Introduction

The rapid development of sensor networks and cloud computing has pushed the emergence of a great deal of data innovation applications for the IoT and other intelligent network systems in the fields of urban transportation, education, medical treatment, and living [1–3]. Most service development processes involve users, data collection organizations, and data development organizations. Data collection organizations are usually trusted by users, but data development organizations are likely to be untrusted by users. With the frequent leakage of sensitive data, individuals and society are paying more attention to the protection of privacy information. It has become a meaningful and challenging problem to resolve the contradiction between the privacy protection required by individuals and the data availability required by data development organizations [4–6].

In a dataset that contains privacy information, the attributes can be summarized into three categories: identification attributes, quasi-identification attributes, and sensitive attributes [7]. Identification attributes are those that can directly distinguish an individual's identity. Quasi-identification attributes (also known as nonsensitive attributes) are multiple attributes that together may infer the identity of an individual. Sensitive attributes are those that contain privacy data. Identification attributes are removed before the dataset is published, but simply hiding the identification attributes does not guarantee privacy because attackers may infer identification attributes based on quasi-identification attributes when attackers adopt link attacks or have users' background knowledge. If the quasi-identification attribute and the identification attribute are hidden together, although it is not possible to deduce a one-to-one association between privacy information and

personal identification, the dataset with sensitive attributes becomes useless.

The approaches related to the issue can be roughly divided into three kinds: data distortion, secure multiparty computing, and data anonymity. The algorithm of data distortion randomly modifies the sensitive data, and it has a high degree of data distortion and strong data dependence. The algorithm of secure multiparty computing forces untrusted third parties to complete data mining work without the direct communication of detailed data through data nodes, and it is very expensive in terms of calculation and communication overhead. At present, most scholars are investigating variations of k-anonymity algorithm, which guarantee that only groups of a minimum size k can be identified, rather than individuals [8]. However, it is proved to be a NP-hard problem to find the optimal information loss and time complexity based on k -anonymity.

For the development of big data-driven services in smart environments, the existing privacy protection algorithms about data publishing have the following deficiencies: (1) the time complexity of most algorithms are relatively high, so they are hard to apply in practice and (2) service development involves many factors, such as the profit of services, quality of services, and privacy and security; however, existing algorithms have not taken the trade-offs between the factors into consideration comprehensively.

Since the attacker identifies the user based on the combination of user's quasi-identifiers and the fewer quasi-identifier fields result in a lower probability of privacy leaks, therefore, in this paper, we aim to investigate an optimal number of quasi-identifier fields under the constraint of trade-offs between service quality and privacy protection. In the initial phase, we first construct a loss function for privacy leakage and a rating function for service quality and then model the service development process as a cooperative game between the data owner and consumers. The Stackelberg game model is employed to get the global optimal solution based on historical data, which comprehensively considers the service quality, privacy leakage loss, and service revenue, so that the number of quasi-identifier fields can be determined. Since the functions of the game model may be biased from reality, we then design a way to determine whether the new data need to be learned and a way to adjust weights of historical data and new data in the update phase, and game model is then used repeatedly to get the new strategy on quasi-identifier fields delivery.

The rest of the paper is organized as follows: Section 2 discusses the work related to data publication with privacy protection. Section 3 introduces the preliminary knowledge about game theory. Section 4 defines the problem and explains our proposed privacy protection model. Section 5 presents the experiments and analyses the results. Section 6 gives the conclusions and future work.

2. Related Work

Many scholars have conducted a series of research work on privacy protection in data publication. Their approaches can be mainly divided into three kinds: data distortion, data encryption, and data anonymization.

Approaches based on data distortion protect the privacy information by disturbing the original data. The attacker cannot reconstruct the original data through the published distorted data, while some information obtained from the distorted data is approximately equivalent to the information obtained from the original data. This kind of approaches [9–13] mainly studies how to perform data perturbation and how to mine the perturbed data, so that people cannot get the value of original data but can get high-quality mining results from the perturbed data. Since the distribution of the perturbed data is almost the same as the original data, the perturbed data can be used to train the learning models well [9]. The method focuses on the goal of preserving privacy by suppressing and perturbing the quasi-identifiers in the data without causing any loss to the information in the process [14]. However, such kind of approaches has a high degree of data loss and strong data dependence.

Approaches based on data encryption define privacy-based data mining applications as a secure multiparty computing problem involving untrusted third parties in many distributed environments. Each part only knows its own input data and the final results of all calculations among two or more sites are communicated through some kind of protocol. In secure multiparty computing [15–18], data are distributed and stored on multiple nodes. Each data node wants to perform data mining on the global data but does not want to disclose its own data. Therefore, information exchange protocols based on secure multiparty computing should be designed. Each data node does not exchange the detailed data samples directly, but it uses the protocols to exchange the information needed by the data mining algorithms in the absence of details of other nodes. However, such kind of approaches has a high computational and communication overhead.

At present, data anonymization is investigated widely and becomes the mainstream way to privacy protection. The k-anonymity model originally proposed by Sweeney [19] is used to defend against background knowledge attacks and link attacks, and generalization and compression techniques are widely used to achieve k-anonymity [20, 21]. Since Aggrawal [22] proved that the clustering method can achieve anonymity in a more efficient way, researchers began to investigate on clustering anonymity algorithm in privacy data publication. Li et al. [23] proposed anonymity scheme that applied the clustering idea, and the anonymity process merges the equivalence classes repeatedly and selects a certain equivalence class according to the principle of the minimum amount of loss of the consolidated generalization until all equivalence classes contain more than k tuples. Zhihui Wang and et al. [24] proposed an L-clustering method that could classify quasi-identifier attributes, they measure the degree of uncertainty of attribute values before and after generalization, and give us a measure of information loss that transformed the data anonymity problem into a clustering problem with specific constraints.

To enhance the performance on time efficiency and information loss in k-anonymity, Zhang et al. [25] proposed

a k-anonymity clustering algorithm based on information entropy, and the first step is to divide the table data into a number of record subsets according to the principle of the minimum average distance on quasi-identification attribute values, while the second step is to merge and split the subsets as appropriate so that the number of records for each subset is between k and $2k$. Huowen Jiang et al. [7] proposed a greedy clustering anonymization method based on the idea of the greedy method and clustering and they separately measured the information loss of the quasi-identifier, and the distances between tuples and the distances between tuples and equivalence classes. The methods mentioned above try to optimize the performance of k-anonymity algorithm, but the optimal information loss and time complexity for k-anonymity algorithm have been proved to be a NP-hard problem, and there is still space for improving the performance of existing algorithms.

3. Introduction to Game Theory

Game theory is originated from "Game Theory and Economic Behaviours," which was coauthored by von Neumann and Morgen stern in 1944 [26]. For the first time, this book presents a complete and clear description of the research framework of game theory and expounds the basic axioms. For a long time, the study of game theory focused only on the double zero-sum game. In the early 1950s, Nash [27] proposed the most important theory in game theory called the Nash Equilibrium, which determined the form and theoretical foundation of the noncooperative game and extended the research field of game theory to noncooperative games and nonzero-sum games. Game theory is suitable for solving conflicts and seeking a Nash equilibrium solution for the problem. In the problem of data publication, we need to publish the user's data to develop smart services, and untrusted data developers may leverage user's private information. Therefore, it is a conflict issue to publish user data for service development, and game theory can be used for modeling.

Game theory is about how smart and self-interested people act in the strategic layout and interact with their opponents. It has three parts: (1) a group of participants; (2) the actions that participants can take; (3) the benefits that participants may get. Each participant chooses the best action for their maximum benefit, and each participant will always think that other participants are also trying to get the best result. If game theory can provide a unique solution to the game problem, the solution must be a Nash equilibrium. The strategy chosen by each participant must be the optimal response to the strategy chosen by the other participants, and no participant is willing to abandon his selected strategy alone.

According to whether there is a binding agreement between the two parties, game theory can be divided into cooperative game and non-cooperative game; according to whether the sum of the revenue of both players is zero, game theory can be divided into zero sum game and nonzero sum game; according to the decision order of the players in the game, game theory can be divided into static games and dynamic games; according to whether the two parties understand the each other's strategy and revenue function, game theory can be divided into complete information game and incomplete information game. We model the problem of privacy data publishing as a cooperative game problem between data collector and data developer and establish the strategy space and revenue function of both parties. The game sequence is that the data collector first publishes privacy data, and then the data developer performs service development based on the data.

4. Privacy Protection Model Based on Game Theory

4.1. Problem Definition. There are a set of datasets with sensitive information for publication, and each dataset can be expressed as $T = \{t_1, t_2 \ldots t_n\}$, where t_i is the ith record, and each record consists of q quasi-identifying attributes and one sensitive attribute, i.e., $t_i = (A^q, A^s)$, where $A^q = \{A^{q1}, A^{q2} \ldots A^{ql}\}$ denotes all the quasi-identifying attributes in the data table and A^s is the sensitive attribute in the table. Table 1 shows an illustrating example of the dataset containing privacy information, where *age*, *sex*, and *zip code* are quasi-identifier attributes and *disease* is the sensitive attribute. The set of datasets can be classified by the sensitive attribute, such as disease, property, and religious beliefs.

Assume we have historical records of service development $R = \{r_1, r_2, r_3, \ldots\ldots, r_m\}$, where r_i is the loss, investment, revenue, and rating score of the service when delivering '*quasi_number*' pieces of quasi-identifiers on datasets with sensitive attribute '*privacy_type,*' and r_i can be expressed as a 7-tuple r_i=<*privacy_type, quasi_number, privacy_loss, technique_investment, A_revenue, B_revenue, score*>. '*privacy_loss*' is the loss of privacy, which consists of direct losses and indirect losses incurred by data collectors. The direct losses include users' privacy disclosure by competitors, and indirect losses include complaints and claims from users. '*Technique_investment*' is the technique costs contributed by data developer. '*A_revenue*' and '*B_revenue*' are the revenues of the data application services achieved by data collectors and data developers, respectively, and '*score*' is the quality score of the service. The samples are shown in Table 2.

The identifying attributes have been removed from Table 1, so that a specific field in the table cannot be mapped to a specific individual. However, simply hiding the identifying attribute does not guarantee privacy security when the attacker knows some background knowledge of the user, for example, the combination of age, sex, and zip code. Under this circumstance, the attacker may also infer and identify a person, which leads to personal privacy disclosure. In the development model of existing services, data collection organizations are trusted by users; however, data development organizations are likely to be untrusted by users.

Given R, we need to perform privacy protection on T assuming that the data developer is not credible; that is, we need to determine an optimal number of quasi-identifier

Table 1: Example of a dataset with sensitive information.

age	sex	zip code	disease
20	female	100018	bronchitis
28	male	300017	flu
32	male	100018	pneumonia
33	male	400015	indigestion
36	female	200017	rhinitis

Table 2: History usage records for the privacy field.

privacy_type	quasi_number	privacy_loss	technique_investment	A_revenue	B_revenue	score
disease	20	1000	600	1700	1000	3.9
disease	16	800	500	1500	7000	3.3
.....	.....	.....	.....	.....	.....	.....
disease	22	300	100	600	300	2.7

fields under the constraint of trade-offs between service quality and privacy protection.

4.2. The Framework of Privacy Protection Model. In order to ensure the data availability for high service quality and to protect users' privacy at the same time, we design a privacy protection model for data publication as shown in Figure 1. The model consists of two phases, namely, service development and service update. In the initial phase of service development, we define the relevant function of the service development to simulate the problem firstly, which includes the revenue functions of the game participants and the variables on which the revenue functions depend. Then we employ the game theory to model the problem and obtain a balanced solution that both parties are willing to accept, and publish the data according to the strategy. In the following phase of service update, we adopt statistical method to detect the constantly updated data, determining whether the service needs to learn the new data. Then, we design an adjustment way of the parameters in the functions based on the actual results in the previous development phase and analyze the data publishing strategy for service update.

In the following sections, we will illustrate the game process of the data collectors and data developer by four parts: (1) the establishment of complete information; (2) the strategy generation in service development based on Stackelberg game model; (3) the detection of the need for service update; (4) the renewal strategy generation in service update.

4.3. The Establishment of Complete Information on Both Sides of the Game. Strategic space is a collection of actions that are available to the parties of the game, and each strategy corresponds to a result. Since the data collector and the data developer cooperate to complete the service development, in this section we need to determine the strategic space and the revenue function of both parties.

We use the new strategic cooperation model which widely used in game theory economics to establish the cooperative relationship between data collector and data developer. Because the dominant party in the cooperation model will subsidize the other party to achieve better cooperation result, we set (Q, k) to represent the strategy combination of the data collector, where 'Q' is the loss of privacy information leakage, and 'k' is the percentage of the economic subsidy to the technology investment. The strategy of the data developer is the investment cost of data mining technology 'a'.

Furthermore, we also need to define the expression of privacy leakage loss 'Q'. The basic principle is that the loss is proportional to the probability of privacy leakage P. In order to obtain the relationship between the probability of privacy leakage and the number of quasi-identifiers, we use Taylor's third-order function to model it. The expression of Q is given in formula (1):

$$\begin{aligned} Q &= u_q P \\ P &= b_3 x^3 + b_2 x^2 + b_1 x + b_0, \end{aligned} \tag{1}$$

where: x represents the number of quasi-identifiers; b_0, b_1, b_2, b_3 represent the pending parameters in the third-order Taylor formula; P is the probability of privacy leakage; $u_q$represents a positive coefficient between P and Q.

We assume that service quality score for data developer is affected by the amount of data information and the investment cost of mining technology. Because the privacy leakage loss Q is defined by the number of quasi-identifiers, and the more the quasi-identifiers, and the amount of information in the data set can be expressed by the number of quasi-identifiers, we need to define the quality of service score as an increasing function of privacy leakage loss 'Q' and mining technology investment cost 'a', and the maximum value of the function is the highest score of service quality. Then, we need to set the parameters to indicate the impact of 'Q' and 'a' on the function and consider the interaction between the two variables which show a complementary relationship to quality of service score function. We consider quality of service score function as shown in

$$\mathrm{S}(\mathrm{Q}, a) = \omega - \beta a^{-\gamma} Q^{-\varsigma}, \tag{2}$$

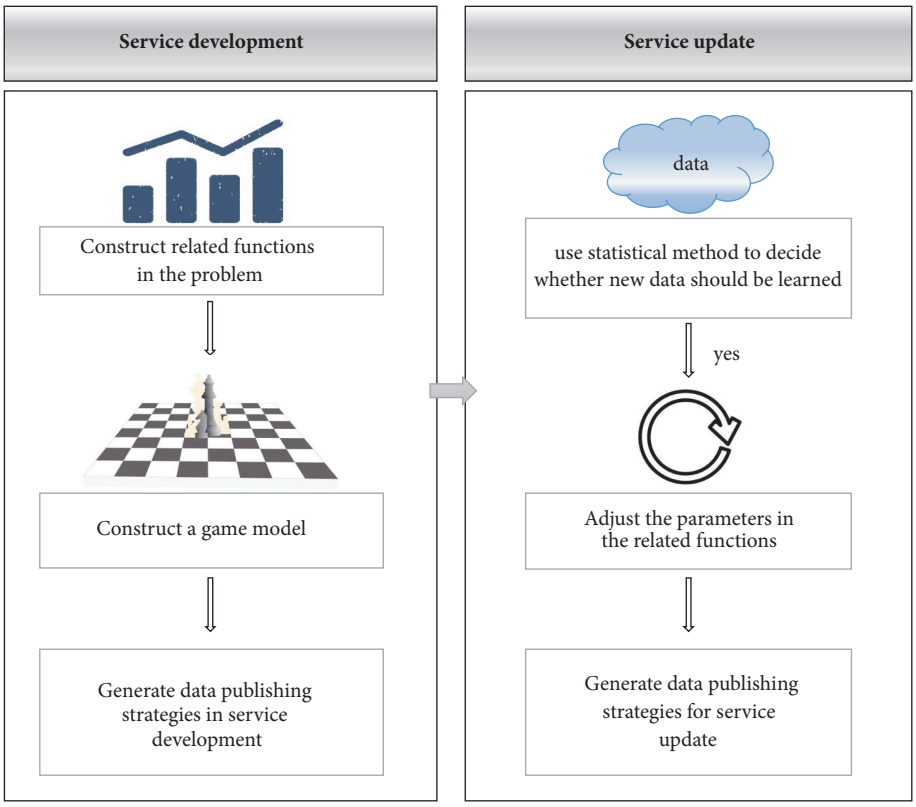


Figure 1: The framework of privacy protection model.

where ω represents the saturation value of the service level, β represents a normal number, γ represents the impact factor of a, and ς represents the impact factor of Q.

The return of both sides is proportional to the service quality score, and the form of revenue function is income minus cost. We define the revenue function of data collector r_m as (3), where ka and Q are the costs of the data collector. The revenue function of the data developer r_n is as (4), where $(1\text{-}k)a$ is the cost of service development.

$$r_m = \lambda S(Q, a) - ka - Q \tag{3}$$

$$r_n = \eta S(Q, a) - (1 - k)a, \tag{4}$$

where $S(Q,a)$ represents the score of service quality, Q represents the loss of privacy leakage, a represents the mining technology investment, k represents the proportion of subsidies to the data development organization, and λ and η represent direct proportionality coefficient.

4.4. Strategy Generation in Service Development Based on Stackelberg Game Model. We then aim to describe the process how data collector and data developer play the Stackelberg master-slave game. The data collector first proposes a cooperation program, and then data developer makes decisions based on the behavior of data collector. In the first phase of the game, data collector determines the proportion of economic subsidy for technique inputs to encourage service development, as well as the loss of the privacy leakage by quasi-identifier fields. In the second phase of the game, the data developer decides to invest the technique based on the data collector's program. Due to the sequence of actions, data collector makes decisions first, and data developer makes corresponding decisions based on the data collector's decision. This is a two-stage game problem that can be solved by inverse induction.

In order to obtain the Nash equilibrium solution, we consider the revenue function of data developer r_n first and get the response function of the data developer about Q(the loss of privacy leakage) and k(the proportion of subsidies to the data developer) by deriving r_n and the process is as follows:

$$\begin{aligned} r_n &= \eta\left(\omega - \beta a^{-\gamma} Q^{-\varsigma}\right) - (1 - k)a \\ \frac{\partial_{r_n}}{\partial_a} &= 0 \end{aligned} \tag{5}$$

From (5), we can obtain the response function of the data developer:

$$a = \left[\frac{\gamma\eta\beta}{(1 - k)Q^{\varsigma}}\right]^{1/(\gamma+1)} \tag{6}$$

After obtaining the reaction function (6) of the data developer, we eliminate the parameter a in the revenue

function of the data collector r_m and then find the partial derivative of the variables in function r_m, the process is as follows:

$$r_m = \lambda\left(\omega - \beta a^{-\gamma} Q^{-\varsigma}\right) - ka - Q$$
$$\frac{\partial_{r_m}}{\partial_Q} = 0, \tag{7}$$
$$\frac{\partial_{rm}}{\partial_k} = 0$$

From (6) and (7), we can obtain the value of Q and k of the data collectors $Q_1$and k_1:

$$k_1 = \frac{\lambda - (1+\gamma)\eta}{\lambda - \gamma\eta} \tag{8}$$

$$Q_1 = \left[\varsigma^{\gamma+1}\beta\gamma^{-\gamma}(\lambda - \eta\gamma)\right]^{1/(\varsigma+\gamma+1)} \tag{9}$$

After getting the value of Q, we can get the number of quasi-identifiers through (1) in data publication.

4.5. Detection of the Need for Service Update. We then aim to build a dynamic mathematical model for the constantly update data and analyze, diagnose, predict, and optimize the system based on the model. We construct a statistical model to identify the extent of the data changes.

To achieve this goal, we construct a unilateral hypothesis test about mean μ when the variance σ^2 is unknown. First, we get the model error evaluation index E_i of the ith test set. Then, we calculate the average error of the model according to Definition 1 and finally construct the distribution function according to Definition 2.

Definition 1. The XinQin Law of Large Numbers state that if $\{x_n\}$ is an independent and identically distributed random variable sequence and $EX_n = \mu$ exists, then

$$\frac{1}{n}\sum_{i=1}^{n} x_i \xrightarrow{p} \mu \tag{10}$$

Definition 2. The Central Limit Theorem states that regardless of $x_i \sim F(\mu, \sigma^2)$, in the case of large samples

$$\frac{1}{n}\sum_{i=1}^{n} x_i \sim N\left(\mu, \frac{\sigma^2}{n}\right) \tag{11}$$

Definition 1 indicates that when the random variables are independent and identically distributed, the average of the random variables tends to the true average with a certain probability p, and the larger the value of n, the closer the value of p is to 1. We derive the mean of the error evaluation criteria on the original test data set from Definition 1 as shown in

$$\mu_0 = \frac{1}{i}\sum_{i=1}^{i} E_i \tag{12}$$

Definition 2 means that it is not necessary to consider the original distribution of the random variables. In the case of large samples, the average of n random variables obeys the normal distribution. Collecting the data sample of a certain time interval, we can get the error evaluation criterion E_{i+1} of the model. Then, we establish two assumptions $H_0 : \mu \le \mu_0$ and $H_1 : \mu > \mu_0$ and construct the statistics as shown in (13), where S represents the sample variance and n represents the sample size.

$$T = \frac{E_{i+1} - \mu_0}{S/\sqrt{n}} \tag{13}$$

When the level of significance $\alpha(0 < \alpha < 1)$ is given, we can get the rejection domain of $H_0 : \mu \le \mu_0$ as $w = \{T > t_\alpha(n-1)\}$. If T falls into the denial domain, it explains that because of the big data dynamic characteristic, the performance of the model with the original knowledge is not within our expectation and the new knowledge needs to be learned. The specific process of update algorithm is shown in Algorithm 1.

4.6. The Renewal Strategy Generation in Service Update. Since the initial model is trained by historical development records of similar services, it may not be consistent with the actual service development; therefore, we need to adjust the weights of new samples according to the accuracy of the learned model. The real values of service score S and the income r_m, r_n of both parties will be generated in the previous service lifecycle, we can compare the predicted values $S^*, r_m{}^*, r_n{}^*$ with the real ones and adjust the weight of new sample data for training involved in the game model.

When the difference between the real value and the predicted value is within a specified range, we believe that the historical data of similar services can train the modeling functions well, so we can add the new data sample using the same weight with the historical data. When the difference exceeds the specified range, the historical data cannot reasonably train the modeling functions, so we adjust the weight of the new sample, making it appear more times so that the new sample will have more important impact on the training process. In formulas (14)-(16), w_1, w_2, and w_3 represent the weights of the new sample in the training process of the correlative functions (2)-(4), respectively, l_1, l_2, and l_3 are the corresponding prediction error threshold values, and N is the number of historical training samples.

$$w_1 = \begin{cases} 1, & |S * - S| < l_1 \\ \frac{1}{2}N, & |S * - S| \ge l_1 \end{cases} \tag{14}$$

$$w_2 = \begin{cases} 1, & |r_m * - r_m| < l_2 \\ \frac{1}{2}N, & |r_m * - r_m| \ge l_2 \end{cases} \tag{15}$$

$$w_3 = \begin{cases} 1, & |r_n * - r_n| < l_3 \\ \frac{1}{2}N, & |r_n * - r_n| \ge l_3 \end{cases} \tag{16}$$

The Nash equilibrium of Stackelberg model is unique. If there is a unique Nash equilibrium in the staged game,

```
input : n, s, E_{i+1}, μ_0, oldservice
output : service
T = (E_{i+1} − μ_0)/(S/√n)
service = 0
IF(T > t_α(n − 1))
    service = re.studyservice
Else
    service = oldservice
return service
```

Algorithm 1: Update Algorithm.

the Nash equilibrium solution of each stage in a game with repeated times is the same solution as the one-time game; therefore, in the next life cycle of the data application service, the decision expression is the same as that in the previous stage.

5. Experiment

In the experiment, we use Adult dataset as the dataset T with sensitive information, which is the protection target. The original dataset has a total of 15 fields, 32561 records. We remove the nonquasi-identifier attributes and add a field of user's ID. A sample data set is shown in Table 3. We then simulate and generate the historical records of service development R, in which parameters are set according to the relevant literature [28, 29]. Table 4 shows the six sets of simulation parameters in the functions.

In the following experiment, we first aim to analyze the rationality of modeling functions and Nash equilibrium solution under the constraints of service quality and privacy protection in Section 5.1, and we then visualize the process that when we have a biased estimate of the functions, the estimation curve will approximate the actual curve by parameters adjustment in Section 5.2. At last, we compare the proposed model with the traditional k-anonymity method in Section 5.3.

We adopt GCP (Global Certainty Penalty) [30] to compare the proposed method with the traditional k-anonymity method and to measure the data availability under the same level of privacy protection. The range of GCP is $[0, 1]$, where 0 means no information loss and 1 means total information loss.

5.1. Analysis of Modelling Functions. We randomly extract 80 sets of samples from Table 3, and each set has 100 records. The extracted sets of samples are used to calculate the probability of identifying a specific user based on the combination of quasi-identifier fields. The detailed process is to (1) calculate the probability that identifies each user in the samples according to the quasi-identifier fields and (2) calculate the average probability of all users. For example, when calculating the average probability of identifying a user in the samples under one quasi-identifier, we randomly select a quasi-identifier in the dataset and calculate the probability that value of the selected quasi-identifier for each user can identify the user, and the average probability of identifying a specific user under different numbers of quasi-identifier fields is shown in Table 5.

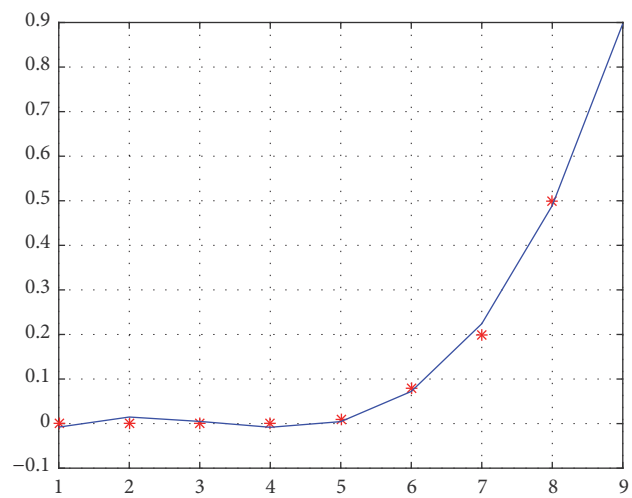


Figure 2: Simulation of formula (1).

We aim to prove that the Taylor third-order function mentioned in formula (1) can well simulate the relationship between the probability of privacy leakage and the number of quasi-identifiers first. In Figure 2, the red dot identifies the real sampling points, which are calculated from Adult dataset, and the blue curve is third-order Taylor function (the value of parameters can be determined by red dots: b_3 = 0.0049, b_2 = -0.0454, b_1 = 0.1248, and b_0 = -0.0921). We can see that the function curve fits the data very well, and it is reasonable to use the Taylor function to model the target variables.

We then aim to prove that the quality of service function and the Nash equilibrium solution are consistent with the reality, which is mentioned in formula (2), Section 4.3. As shown in Figure 3, we perform visual analysis of the quality of service function, and we can see that the curves under six sets of parameters in Table 4 basically follow the similar shape. The quality of service increases with the increase of a (mining technology investment) and Q (the loss of privacy leakage), and the function curve increases rapidly first, then the growth tends to be gentle. During the growth of the curve, the value tends to reach the highest service quality score, i.e., the full score ω in the function. Considering the user's privacy and the

Table 3: The sample of adult data set.

Id	workclass	education	marital_status	occupation	relationship	race	sex	native_country	income	agerank	numrank
A1	State-gov	Bachelors	Never-married	Adm-clerical	Not-in-family	White	Male	United-States	<=50K	30~40	13~16
A2	Self-emp-not-inc	Bachelors	Married-civ-spouse	Exec-managerial	Husband	White	Male	United-States	<=50K	50~60	13~16
A3	Private	HS-grad	Divorced	Handlers-cleaners	Not-in-family	White	Male	United-States	<=50K	30~40	9~12
A4	Private	11th	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	United-States	<=50K	50~60	5~8
A5	Private	Bachelors	Married-civ-spouse	Prof-specialty	Wife	Black	Female	Cuba	<=50K	20~30	13~16
A6	Private	Masters	Married-civ-spouse	Exec-managerial	Wife	White	Female	United-States	<=50K	30~40	13~16
A7	Private	9th	Married-spouse-absent	Other-service	Not-in-family	Black	Female	Jamaica	<=50K	40~50	5~8
A8	Self-emp-not-inc	HS-grad	Married-civ-spouse	Exec-managerial	Husband	White	Male	United-States	>50K	50~60	9~12
A9	Private	Masters	Never-married	Prof-specialty	Not-in-family	White	Female	United-States	>50K	30~40	13~16
A10	Private	Bachelors	Married-civ-spouse	Exec-managerial	Husband	White	Male	United-States	>50K	40~50	13~16

Table 4: The simulation of parameters in the functions.

simulation	λ	η	ω	β	γ	ς
1	246	82	50	10	0.2	0.4
2	459	153	67	13.4	0.3	0.6
3	30	10	78	15.6	0.1	0.2
4	87	29	73	14.6	0.1	0.2
5	318	106	58	11.6	0.3	0.6
6	291	97	45	9	0.2	0.4

Table 5: The average probability of identifying specific users.

Test	The number of quasi-identifier fields							
	1	2	3	4	5	6	7	8
test 1	0.00005	0.00016	0.00100	0.00261	0.01639	0.08333	0.16667	1.00000
test 2	0.00021	0.00060	0.00179	0.00307	0.01003	0.08001	0.14285	0.20000
test 3	0.00009	0.00013	0.00201	0.00211	0.01173	0.07990	0.14285	0.25000
test 4	0.00028	0.00078	0.00101	0.00324	0.01032	0.07720	0.25000	0.25000
test 5	0.00010	0.00014	0.00181	0.00201	0.01709	0.08070	0.25000	1.00000
test 6	0.00005	0.00092	0.00209	0.00200	0.02009	0.08003	0.16667	1.00000
test 7	0.00006	0.00040	0.00130	0.00209	0.02013	0.07693	0.16667	0.16667
test 8	0.00004	0.00072	0.00271	0.00350	0.01631	0.08000	0.16667	0.16667
test 9	0.00005	0.00055	0.00160	0.00311	0.02079	0.07932	0.14285	1.00000
test 10	0.00017	0.00031	0.00101	0.00301	0.01089	0.08702	0.14285	0.25000
average	0.00011	0.00048	0.00167	0.00269	0.01501	0.08077	0.17099	0.57121

response strategy of the data developer, the Nash equilibrium strategy is not close to the strategy combination with the highest service quality score, and the strategy takes a high point rather than the optimal point. In the untrustworthy cooperation process of reality, our intuition is that the target variable increases with the increase of both sides, the target variable will gradually approach the maximum value of the target variable, and because the two sides do not trust each other, they cannot achieve the best solution and the quality of service function can reasonably simulate the cooperation between data developer and data collector.

Next, we aim to prove that the profit function is reasonable and the maximum point is the Nash equilibrium solution. Since the revenue function of the data collector is related to three variables as shown in formula (3), Section 4.3, it is necessary to eliminate one variable a according to formula (6), Section 4.4. We then plot equivalent profit curves under (Q, k) space in Figure 4. We can find the six curves are all convex, and the Nash equilibrium of all curves corresponds to the maximum values. The profit of data collector increases first with respect to Q and k (the proportion of subsidies to the data developer), and the rate of increase becomes smaller as Q and k increase, and when it increases to the highest point, the function curve begins to drop. The revenue function of the data collector takes into account the quality of service, the loss of privacy leakage, and the strategic variables of the data developer, the maximum value of the function corresponds to the Nash equilibrium solution. The function curve is consistent with our expectation as follows: (1) the function value increases with the service quality score and (2) it is a convex function about the input cost.

5.2. Analyze the Iterative Process. In this section, we aim to prove that the training after sample weight adjustment can approximate the real function when the initial modeling is biased from reality, and the actual parameters of the functions in the experiment are $\lambda = 246$, $\eta = 82$, $\omega = 50$, $\beta = 10$, $\gamma = 0.2$, and $\varsigma = 0.4$.

When the absolute difference between the real value and the estimated value is greater than the specified threshold, we will adjust the weights of the new samples. As shown in Figure 5(a), we mark the real function curve and set the threshold of error to 2. We use MATLAB randomly generate 20 reasonable sample scatter points larger than the threshold and draw the initial estimation curve by gradient descent algorithm. At the end of the service, a real sample scatter is generated, so we take the point on the real curve and use the gradient descent algorithm to estimate the curve by the sample weight adjustment method in Section 4.6. When the function simulation curve is corrected several times according to the real data, the error between the predicted value and the true value is less than the threshold and the sample weight is no longer changed. Figure 5(b) represents the revenue function of the data collector, and the threshold is set to 100 and the iterative simulation process is similar with Figure 5(a).

5.3. Effectiveness Analysis of Privacy Protection. We aim to compare the effect of privacy protection of our proposed model with the traditional method k-anonymity, and the result is shown as Table 6, where P is the probability of privacy leakage and K presents the size of anonymity group when k-anonymity achieves the same privacy protection. We compare

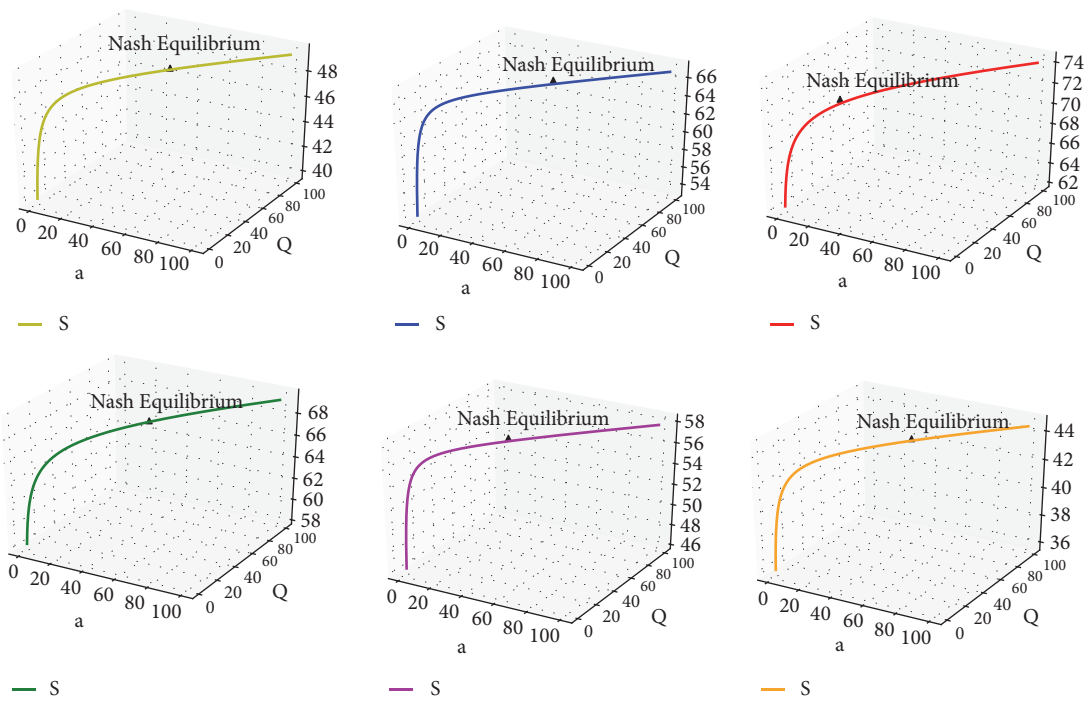


Figure 3: Simulation of formula (2).

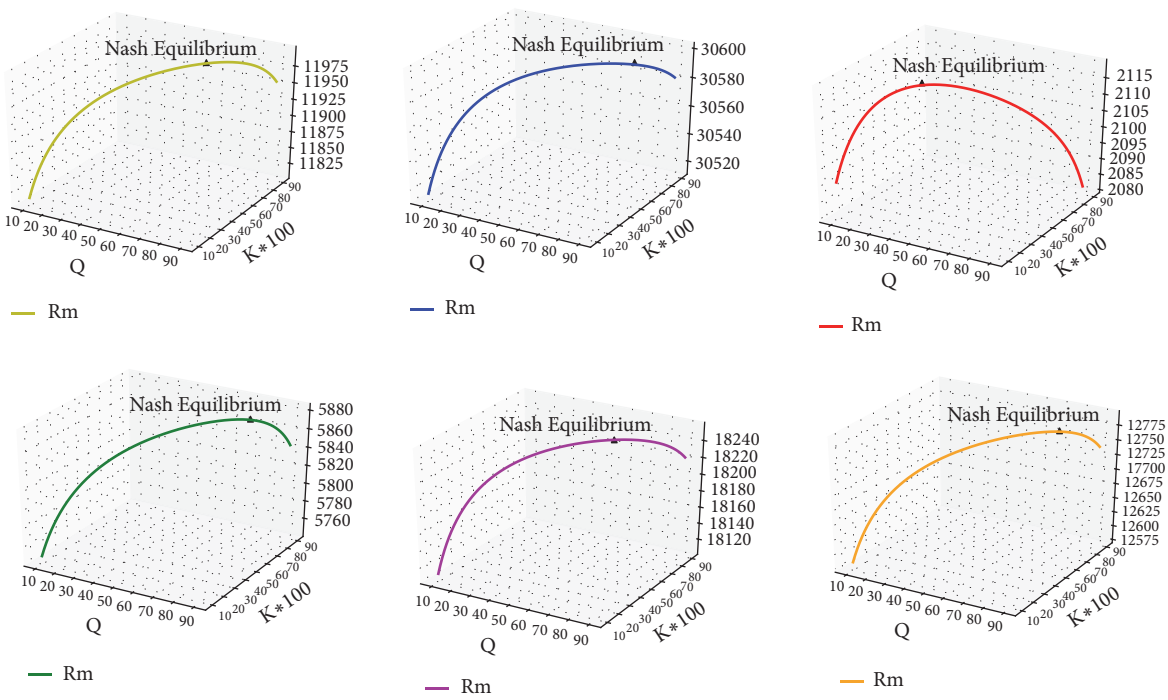


Figure 4: Simulation of formula (3).

the two methods from the GCP and time complexity. In our approach, *reduced number of fields* represents the number of quasi-identifier fields that are reduced.

In data applications, deleting the appropriate quasi-identifier fields will not affect the data mining results, but deleting too many quasi-identifier fields will reduce the upper bound of the accuracy. When k-anonymity algorithm is used for privacy protection, no matter how much K is equal to, it will cause loss of data information which limits the application of mining algorithm and the upper bound of accuracy. Generally, P is thought to be weak when P is between $[0.1, 0.5]$, middle when P is $[0.01, 0.1)$, and strong

Table 6: Comparison of the two methods.

P	k-anonymity		Game theory			
	GCP(%)	time complexity	K	GCP(%)	time complexity	Reduced number of fields
0.5	8.60		2	27.27		3
0.2	21.71		5	36.36		4
0.08	34.59		13	45.45		5
0.01	54.93	$O(n^2)$	100	54.54	$O(n)$	6
0.002	71.27		500	63.63		7
0.001	73.81		1000	72.72		8
0.0005	79.95		2000	81.81		9

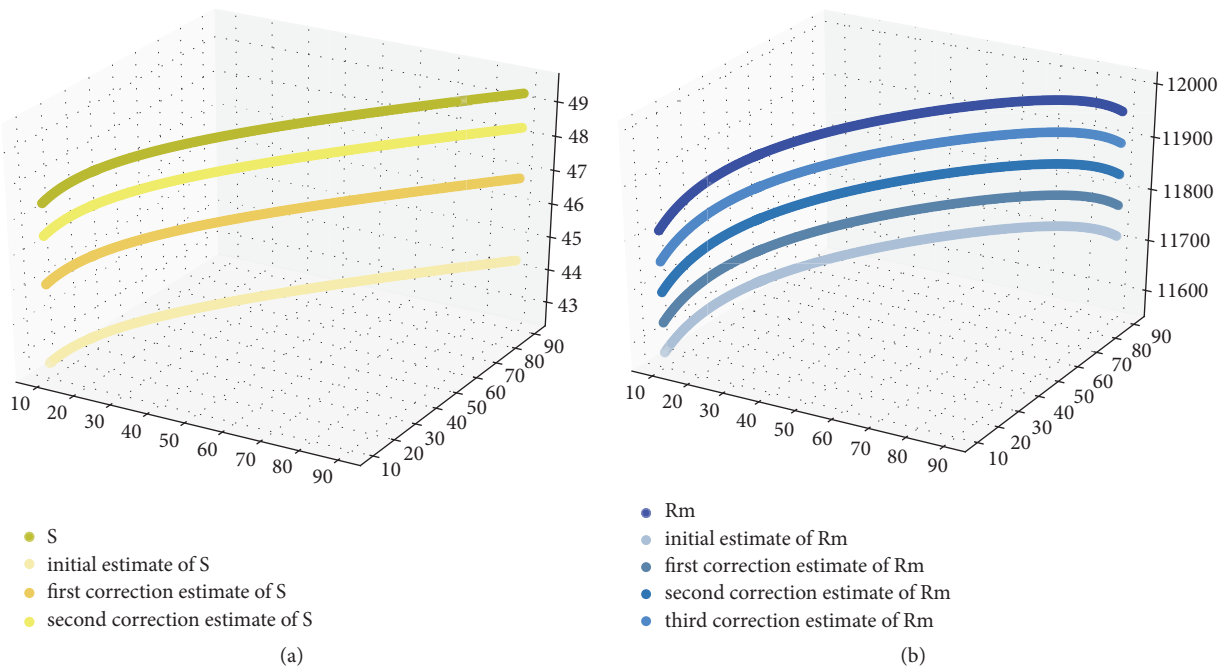


Figure 5: Iterative process of modeling functions.

when P is $[0, 0.01)$. GCP represents the information loss, 0 means no information loss and 1 means total information loss. The less the GCP, the better the data availability.

From Table 6 we can see that the GCP of k-anonymity increases stably when P changes from weak to strong, as well, and the size of k increases from 2 to 1000. The GCP of our proposed model also increases with P, and the reduced number of quasi-identifiers increases from 3 to 8. When users need week privacy protection, the GCP of k-anonymity is smaller than game theory model, and it is appropriate to select k-anonymity to process the data. But when users need middle privacy protection, our model performs better than k-anonymity. On the other hand, we calculate the number of published quasi-identifiers through the privacy protection model, and the data developer selects the corresponding number of quasi-identifier field from the source data fields, and when data needs to be applied with week privacy protection, fewer quasi-identifier fields need to be reduced and the truncated fields may have little impact on the target variable in practice.

Furthermore, we compare the time complexity of the two methods. Because our proposed privacy protection model relies on modeling functions, where the unknown parameters of the functions can be determined by gradient descent algorithm, we can use the stochastic gradient descent (SGD) [31] to solve the problem in practice and the time complexity of the SGD is $O(n)$. We use an improved k-anonymous algorithm [7] for comparison, and its time complexity is $O(n^2)$. Therefore, the game theory method is simple to implement in practical application and can adapt to the large amount of data.

6. Summary

This paper introduces the developmental characteristics of the data application service in the intelligent network system and analyzes the shortcomings of the privacy protection algorithm in solving such problems. On this basis, this paper proposes a privacy protection model based on game theory, which protects users' sensitive information by reducing the

number of quasi-identifier fields in the released data table, and the strategy calculated by the game model can simultaneously protect user privacy and service quality. This paper introduces the proposed architecture of the privacy protection model that is based on game theory and contains the realization of the process. Finally, we verify by experiments that the proposed privacy protection model can effectively protect the user privacy and service quality. However, in the development of the big data services of the intelligent network system, there is still a lack of algorithms and models that are effective and have the greatest degree of protection for user privacy. To achieve the maximum privacy and security of users, the relevant laws need to be further improved and a deeper study of the relevant issues in the industry is needed.

Data Availability

Previously reported Adult data were used to support this study and are available at http://archive.ics.uci.edu/ml/datasets/Adult. These prior studies (and datasets) are cited at relevant places within the text as [7].

Conflicts of Interest

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

Acknowledgments

The research is supported by National Natural Science Foundation of China (no. 61772560), National Key R&D program of China (nos. 2018YFB1003800, 2017YFB1400601), and National Science Foundation of China (no. 61772461).

References

[1] X. Zhang, W. Dou, Q. He et al., "LSHiForest: A generic framework for fast tree isolation based ensemble anomaly analysis," in *Proceedings of the 33rd IEEE International Conference on Data Engineering, ICDE 2017*, pp. 983–994, USA, April 2017.

[2] J. Zhang, Z. Zhou, S. Li et al., "Hybrid computation offloading for smart home automation in mobile cloud computing," *Personal and Ubiquitous Computing*, vol. 22, no. 1, pp. 121–134, 2018.

[3] L. Kuang, L. Yu, L. Huang et al., "A Personalized QoS Prediction Approach for CPS Service Recommendation Based on Reputation and Location-Aware Collaborative Filtering," *Sensors*, vol. 18, no. 5, p. 1556, 2018.

[4] L. Kuang, Y. Wang, P. Ma et al., "An Improved Privacy-Preserving Framework for Location-Based Services Based on Double Cloaking Regions with Supplementary Information Constraints," *Security and Communication Networks*, vol. 2017, Article ID 7495974, 15 pages, 2017.

[5] L. Qi, X. Zhang, W. Dou, C. Hu, C. Yang, and J. Chen, "A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment," *Future Generation Computer Systems*, vol. 88, pp. 636–643, 2018.

[6] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[7] H. W. Jiang, G. S. Zeng, and H. Y. Ma, "Greedy clustering anonymous method for privacy preservation of table-data publishing," *Journal of Software. Ruanjian Xuebao*, vol. 28, no. 2, pp. 341–351, 2017.

[8] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of the ACM SIGMOD Conference on Management of Data (SIGMOD)*, pp. 439–450, Dallas,Texas, 2000.

[9] J. Thanveer, "A Multiplicative Data Perturbation Method to Prevent Attacks in Privacy Preserving Data Mining," *International Journal of Computer Science and Innovation*, vol. 2016, no. 1, pp. 45–51, 2016.

[10] Z. Ming, W. Zheng-Jiang, and H. Liu, "Random projection data perturbation based privacy protection in WSNs," in *Proceedings of the 2017 IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2017*, pp. 493–498, USA, May 2017.

[11] T. Jahan, G. Narsimha, and C. V. Rao, "Multiplicative Data Perturbation Using Fuzzy Logic in Preserving Privacy," in *Proceedings of the International Conference on Information and Communication Technology for Competitive Strategies. ACM*, pp. 1–5, 2016.

[12] V. S. Reddy and B. T. Rao, "A combined clustering and geometric data perturbation approach for enriching privacy preservation of healthcare data in hybrid clouds," *International Journal of Intelligent Engineering and Systems*, vol. 11, no. 1, pp. 201–210, 2018.

[13] Y. Shen, R. Chen, and H. Jin, "Differentially Private User Data Perturbation with Multi-level Privacy Controls," in *Machine Learning and Knowledge Discovery in Databases*, vol. 9852 of *Lecture Notes in Computer Science*, pp. 112–128, Springer International Publishing, 2016.

[14] A. Kaur, "A hybrid approach of privacy preserving data mining using suppression and perturbation techniques," in *Proceedings of the 2017 IEEE International Conference on Innovative Mechanisms for Industry Applications, ICIMIA 2017*, pp. 306–311, India, February 2017.

[15] K. Gai, M. Qiu, H. Zhao, and J. Xiong, "Privacy-Aware Adaptive Data Encryption Strategy of Big Data in Cloud Computing," in *Proceedings of the 3rd IEEE International Conference on Cyber Security*, pp. 273–278, China, June 2016.

[16] K. Gai, M. Qiu, and H. Zhao, "Privacy-Preserving Data Encryption Strategy for Big Data in Mobile Cloud Computing," *IEEE Transactions on Big Data*, 2017.

[17] A. H. Aljammal, A. Alsarhan, A. Qawasmeh, H. Bani Salameh, and A. F. Otoom, "A new technique for data encryption based on third party encryption server to maintain the privacy preserving in the cloud environment," *International Journal of Business Information Systems*, vol. 28, no. 4, p. 393, 2018.

[18] H. Zhou, "Classification of Large Data Privacy Encryption Simulation Research," *Computer Simulation*, 2016.

[19] L. Sweeney, "k-Anonymity privacy protection using generalization and suppression," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 571–588, 2002.

[20] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Incognito: efficient full-domain *k*-anonymity," in *Proceedings of the Proceeding of the ACM SIGMOD International Conference on Management of Data (SIGMOD '05)*, pp. 49–60, June 2005.

[21] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan, "Mondrian multidimensional K-anonymity," in *Proceedings of the 22nd International Conference on Data Engineering (ICDE '06)*, 25 pages, April 2006.

[22] G. Aggarwal, T. Feder, K. Kenthapadi et al., "Achieving anonymity via clustering," in *Proceedings of the Proceeding of the 25th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '06)*, pp. 153–162, New York-NY-USA, June 2006.

[23] J. Li, R. C. Wong, A. W. Fu, and J. Pei, "Achieving k-Anonymity by Clustering in Attribute Hierarchical Structures," in *Data Warehousing and Knowledge Discovery*, vol. 4081 of *Lecture Notes in Computer Science*, pp. 405–416, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[24] Z. Wang, J. Xu, W. Wang, and B. Shi, "A Clustering-Based Approach for Data Anonymization," *Journal of Software*, vol. 21, no. 4, pp. 680–693, 2010.

[25] J. Yang, B. Zhang, J. P. Zhang, and J. Xie, "A k-anonymity clustering algorithm based on the information entropy," in *Proceedings of the 2014 IEEE the 18th Int' l Conf.on Computer Supported cooperative work in Design*, pp. 319–324, 2014.

[26] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*, Princeton University Press, Princeton, NJ, USA, 1944.

[27] J. Nash, "Equilibrium points in N-person games," *Proceedings of the National Acadamy of Sciences of the United States of America*, vol. 36, pp. 48-49, 1950.

[28] N. Amrouche, G. Martín-Herrán, and G. Zaccour, "Pricing and advertising of private and national brands in a dynamic marketing channel," *Journal of Optimization Theory and Applications*, vol. 137, no. 3, pp. 465–483, 2008.

[29] R. Frank H and B. S. Bernanke, "Principles of microeconomics," McGraw-Hill Irwin, New York, 2007.

[30] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis, "Fast data anonymization with low information loss," in *Proceedings of the 33rd International Conference on Very Large Data Bases, VLDB 2007*, pp. 758–769, Austria, September 2007.

[31] S. Ruder, *An overview of gradient descent optimization algorithms*, 2016.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 6241921, 11 pages
https://doi.org/10.1155/2018/6241921

Research Article

A Constraint-Aware Optimization Method for Concurrency Bug Diagnosis Service in a Distributed Cloud Environment

Lili Bo [1,2] **and Shujuan Jiang** [1,2]

[1] *School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China*
[2] *Engineering Research Center of Mine Digitalization of Ministry of Education, Xuzhou 221116, China*

Correspondence should be addressed to Shujuan Jiang; shjjiang@cumt.edu.cn

Received 23 August 2018; Accepted 23 September 2018; Published 9 October 2018

Guest Editor: Xuyun Zhang

The advent of cloud computation and big data applications has enabled data access concurrency to be prevalent in the distributed cloud environment. In the meantime, security issue becomes a critical problem for researchers to consider. Concurrency bug diagnosis service is to analyze concurrent software and then reason about concurrency bugs in them. However, frequent context switches in concurrent program execution traces will inevitably impact the service performance. To optimize the service performance, this paper presents a static constraint-aware method to simplify concurrent program buggy traces. First, taking the original buggy trace as the operation object, we calculate the maximal sound dependence relations based on the constraint models. Then, we iteratively check the dependent constraints and move forward current event to extend thread execution intervals. Finally, we obtain the simplified trace that is equivalent to the original buggy trace. To evaluate our approach, we conduct a set of experiments on 12 widely used Java projects. Experimental results show that our approach outperforms other state-of-the-art approaches in terms of execution time.

1. Introduction

Cloud computing organizes and integrates different computing resources (including software and hardware), providing end-users with different services in remote location over the Internet. Testing-as-a-Service (TaaS) based on cloud platform provides automated software testing services, saving capacity and reducing expense [1, 2]. With the increasing popularity of service computing, a vast amount of services-related business applications has emerged, such as service composition [3, 4], service recommendation [5–8], service evaluation [9–11], and service optimization [12–15]. As an important guarantee to the QoS (Quality of Service), such as test effectiveness and efficiency, service optimization has attracted much attention of researchers in software engineering.

Prevalent multicore architecture and big data applications today accelerate the development of concurrent systems [16, 17]. To fully utilize multicore CPUs, multiple execution flows can run simultaneously, i.e., data access concurrency. However, that is more likely to suffer from concurrency bugs, which can pose a great threat to the security and privacy in cloud [18–24]. Furthermore, more scalable and efficient anomaly detection and intrusion detection techniques are needed in big data applications [25–28].

Previous studies have proposed a lot of approaches to expose and detect all kinds of concurrency bugs [29], such as deadlocks [30, 31], data races [32, 33], atomicity violations [34, 35], and order violations [36]. Also, the studies have obtained many excellent results. In addition, a variety of record-replay systems are implemented to replay concurrency bugs effectively [37–39]. However, few researches focus on concurrency bug diagnosis. Concurrency bug is difficult to diagnose as frequent context switches hinder developers to understand concurrent program execution traces. In a concurrent program execution trace, most context switches are the fine-grained thread interleaves which are conflicted on accessing the shared memory. The order of accessing the shared memory for two threads forms a dependence relation. The more dependence relations there are, the more difficult it is to reason about concurrency bugs. Additionally, a series of

operations will happen when CPU executes a context switch, including preserving the current site and loading the next site. These operations obviously bring tremendous performance consumption.

Therefore, it is necessary to introduce an optimization technique that can reduce the shared memory dependences and increase the granularity of thread interleaving with the promise of replaying the same bugs. In this paper, we present a static constraint-aware approach to optimize the process of concurrency bug diagnosis. We analyze the original buggy trace offline and simplify it automatically to get a new equivalent trace with less context switches. Our experiments are conducted on 12 widely used Java concurrent benchmarks. Experimental results show that our approach performs better than or is comparable to the compared method (i.e., SimTrace [40]) in reduction as well as performance.

In summary, the main contributions of this paper are listed as follows:

(1) We present a static constraint-aware optimization method-CAOM for obtaining simplified traces which are equivalent to the original buggy traces.

(2) We demonstrate the effectiveness and efficiency of our approach with extensive evaluation on a suit of popular multithreaded projects.

The remainder of this paper is organized as follows. Section 2 summarizes the related work. Section 3 describes the problem formulation and the research motivation. Section 4 presents our constraint-aware optimization method for concurrency bug diagnosis service. In Section 5, we conduct an empirical study to show its validity and finally, in Section 6, we conclude the paper.

2. Related Work

How to optimize cloud users' service invocation cost is always a hot research topic in cloud computing. With considering the service's past invocation costs, FL-FL method was proposed in [41] to evaluate and predict a cloud user's service invocation cost. Unfortunately, it cannot generate an accurate service invocation cost. Work in [42] presented a cost-benefit-aware cloud service scheduling method Cost-plus, but it failed to minimize the service invocation cost. Some researchers focus their work on minimizing the service invocation cost. For example, Li et al. [43] proposed FCFS, which utilized the role of "Fist Come Fist Serve" to reduce the waiting time of user job to optimize the service invocation cost. However, these methods neglected many important factors, such as user job size. Recently, CS-COM was put forward in [12] with considering multiple factors, which significantly optimized the service invocation cost.

In addition to the service invocation cost optimization, researchers also put forward many methods to optimize the performance of other services, such as concurrency bug diagnosis service. Concurrency bug diagnosis attempts to reason about concurrency bugs in buggy traces. An effect optimization approach for improving the performance of concurrency bug diagnosis is to simplify the buggy trace. Trace simplification techniques can be divided into online analysis and offline analysis.

Online trace simplification technique uses vector clock or lock assignment and then groups variables with transitive reduction and thread/spatial locality. The author in [44] proposed to record only the conflicting shared memory dependences with transitive reduction, reducing the time and space overhead of recording. To further reduce the record overhead, Xu et al. proposed FDR (Flight Data Recorder) [45] and RTR (Regulated Transitive Reduction) [46], which record the strict vector dependences based on hardware. In [47], an execution reduction system was developed combined with checkpoint, which removes the events irrelevant to errors. Recently, a software-only algorithm (bisectional coordination protocol) was presented in [48] to reduce the shared memory dependences. Experimental results indicated that the software-only approach was effective and efficient in trace simplification.

Offline trace simplification technique first obtains a complete buggy trace and then simplifies it offline. SimTrace [40] is a classical offline trace simplification technique, but it consumed too much time on constructing the dependence graph and random selection. Jalbert and Sen were the first to present a heuristic dynamic trace simplification approach, Tinertia [49]. To reduce the context switches in the buggy trace, they performed three operations (i.e., Remove Last and Two-Stage Consolidate Up and Consolidate Down) iteratively and constantly replayed the middle trace to validate the equivalence, which increased the runtime overhead seriously. To speed up replay, the authors in [50] simplified the process of replaying concurrency bugs using replay-supported execution reduction. However, multiple replay verification reduced the simplification performance.

In view of the limitations of the existing approaches, we propose a new constraint-aware static trace simplification approach to optimize concurrency bug diagnosis service, as elaborated in the next section.

3. Problem Formulation and Motivation

In this section, we first formulate the problem of trace simplification for concurrent programs. Then, we present an example to motivate our research.

3.1. Problem Formulation. Trace simplification technique attempts to obtain a simplified trace with less context switches yet still equivalent to the original buggy trace. Next, we give the relevant definitions in detail.

(1) Event. A minimum execution unit in that cannot be interrupted in a concurrent program execution. If this event is an access to the shared variables, it is a global event. Otherwise, it is a local event.

(2) Trace. A trace, denoted by $tr = <e_1, e_2, \ldots e_i, \ldots>$, is an event sequence of a program execution.

(3) Context Switch (CS). In a trace, a context switch occurs when two consecutive events are performed by two different threads.

```
//Thread 1                  //Thread 2
If(count==0){
   lis. add(name);          print(count);
   count++;                 if (count == 1){
   print(count);               lis.clear();
   lis.get(count);          }
}
```

Figure 1: An example program.

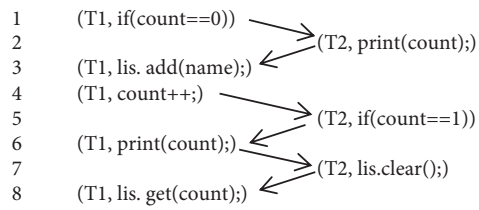


Figure 2: A buggy trace of the example program (CS=6).

(4) Dependence Relation. A dependence relation is the minimum transitive closure over the events in the trace, denoted as $e_i \longrightarrow e_j$.

The dependence relation can be divided into local dependence and remote dependence according to the fact that whether two events occur in different threads. If e_i immediately precedes e_j in the same thread, e_i and e_j are in local dependence relation. Otherwise, they are in remote dependence relation.

Note that two events belonging to remote dependence relation must access the same shared variable. Therefore, the remote dependence relation can be further classified into conflict order and synchronization order according to the types of events. Conflict order contains *read⟶write*, *write⟶read*, and *write⟶write*. Synchronization order consists of *unlock⟶lock*, *fork⟶start*, *exit⟶join*, and *notify⟶wait*.

(5) Equivalent Trace. The original trace is equivalent to the simplified trace if and only if they arrive at the same final state from the same initial program state. The simplified trace is called the equivalent trace of the original trace.

(6) Thread Execution Interval (TEI). The largest set of consecutive events in a thread is a thread execution interval. As we can see, the relationship between the number of context switches and the number of threads execution interval is

$$|TEI| = |CS| + 1 \tag{1}$$

$|TEI|$ and $|CS|$ represent the number of *TEI* and the number of *CS* in the trace, respectively. The goal of trace simplification problem is to make $|CS|$ as small as possible, that is, to make *TEI* as large as possible. Therefore, in the process of trace simplification, under the premise of ensuring trace equivalent, we can put together as many adjacent events in a thread as possible.

3.2. Research Motivation. We use the example in Figure 1 to illustrate trace simplification problem. Assume that the variables *count* and *lis* are initialized to zero and null, respectively. There are two threads accessing the shared variables *count* and *lis* concurrently under the sequence consistency memory model (*SC*). All statements are executed atomically. A null pointer exception will happen in the case that Thread2 executing "list.clear()" occurs between Thread1 executing "count++" and "list.get(count)". In fact, this is a concurrency bug. Figure 2 shows an execution sequence obtained after running the example program. Like [40], we call this sequence a buggy trace. In Figure 2, there are six context switches.

When developers debug concurrent programs, they may run them many times. Each time the program gets error, such as crash, hang, or inconsistent results, developers have to reason about the concurrency bugs along with frequent context switches. That undoubtedly consumes too much time and energy. Trace simplification technique can alleviate this problem effectively. However, two challenges arise in trace simplification: (1) the program semantics are easy to be changed by mistakes and (2) the efficiency of simplification is reduced tremendously because of too many instruments and dynamic verification.

In view of these challenges, we propose a new static constraint-aware approach to simplify concurrent program execution traces and optimize the process of concurrency bug diagnosis. The detailed description of our approach will be given in the next section.

4. A Constraint-Aware Optimization Method for Concurrency Bug Diagnosis Service

In this section, a constraint-aware trace simplification method is proposed to optimize the performance of concurrency bug diagnosis. We first briefly describe the overview of our method. Then, we present the algorithm and corresponding explanation.

4.1. Overview. The overview of our method is described in Figure 3. It mainly consists of three steps. The first step is

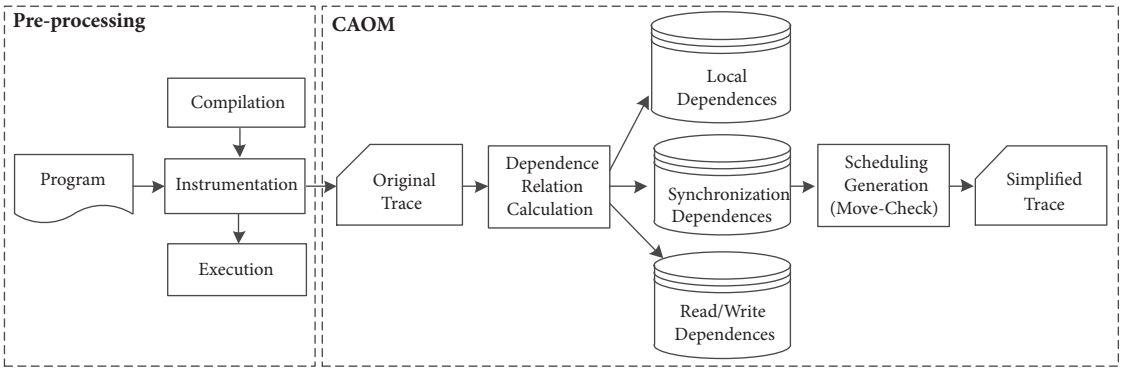


Figure 3: Overview of CAOM.

preprocessing. Test program is compiled to be the corresponding bytecode. After this bytecode is instrumented by Soot [51], we run the instrumented test program and collect the buggy execution trace. In the second step, we calculate local dependences, synchronization dependences, and read/write dependences. The final step is scheduling generation. We take the original trace as input and iteratively check the move forward condition for each event with the constraint dependence relations. If the condition is satisfied, the event is moved forward to extend the thread execution externally. Finally, we obtain the simplified trace.

Note that, in the process of preprocessing, we use the existing instrumentation and record tools to collect original traces. Therefore, we mainly focus on the last two steps: dependence relation calculation and scheduling generation.

4.2. Dependence Relation Calculation. The root cause of nondetermination for thread scheduling is the shared memory access. This leads to the fact that multiple threads can access the same shared variable simultaneously. Therefore, it is a precondition for trace simplification to accurately identify the dependence relations between events in the original buggy trace.

Calculating local dependences only needs to divide the events into several sequences in order according to the threads they belong to. The number of sequences is equal to that of threads. Synchronization dependences can be obtained during collecting the original trace. Then, in this step, we focus on calculating remote read/write dependences. In order to get accurate remote read/write dependence relations, we successively deal with two adjacent accesses on the same shared variable to ensure that the value by read access is always written by the latest write access.

First, we traverse the original trace and divide events into different lists according to the shared variables they access. Then, for the access sequence of every shared variable, we check two successive events in sequence from the first event. If the current two events belong to different threads, they form a remote dependence relation. Furthermore, if two events are both write accesses, they form a remote write-write dependence. If a read event precedes a write event, they form a remote read-write dependence. If a write event precedes a read event, they form a remote write-read dependence.

4.3. Scheduling Generation. Scheduling generation attempts to reduce the number of context switches by two operations (i.e., check and move forward) without breaking all the dependence relations in original traces. A natural thought is to employ a constraint solver that solves three constraints (i.e., write-write dependences, read-write dependences, and write-read dependences). Although the obtained trace satisfies dependence relations in the original trace, the number of context switches may not be reduced. Therefore, we directly take the original trace as the operate object. We check and move forward the atomic events in sequence. Checking is to maintain constraints. Moving is to extend thread execution interval, reducing as many context switches as possible. The detailed process of scheduling generation is shown in Algorithm 1.

Algorithm 1 takes the original trace and dependence relations as input and takes the simplified scheduling sequence as output. Algorithm begins from the second event. If the current event has no dependent event (synchronization dependences or remote read/write dependences), it is moved forward to the location behind the latest event which belongs to the same thread (lines (9)-(10)). If the dependent event is before the latest event which belongs to the same thread with current event, the current event is moved forward to the location behind the latest event (lines (14)-(15)). Otherwise, the events are not moved.

According to Theorem 1 in [40], we know that any rescheduling of events in a trace respecting the dependence relation generates an equivalent trace. In our method, all the feasible events were moved without breaking the dependence constraints. First, we check the dependence relations of the current event. Then, we move it forward under the constraint conditions. That is, all the dependence relations of every event in the new scheduling are the same as that in the original trace. Therefore, the generated trace simplified by our method is equivalent to the original trace.

5. Experiments

5.1. Experimental Configurations. In this section, we conducted experiments on 12 widely used Java multithreaded programs. The details are listed in Table 1. For every program, its lines of code (LOC), number of threads (#Thread),

```
(1)  input: δ - the original trace
(2)         deps - map of sv to its dependence relations
(3)  output: scheduleSeq - sequence of thread schedule
(4)  begin
(5)    for i ⟵ 1 to |δ| − 1 do
(6)      t_δ[i] ⟵the thread identifier of the event δ[i];
(7)      e_latest ⟵the latest event whose thread identifier is t_δ[i];
(8)      loc_e_latest ⟵the location of e_latest in the new trace;
(9)        if δ[i].getDep() = null then
(10)             insert δ[i] immediately after loc_e_latest;
(11)       else
(12)             e_dep ⟵ δ[i].getDep();
(13)             loc_e_dep ⟵the location of e_dep in the new trace;
(14)             if loc_e_dep < loc_e_latest then
(15)                 insert δ[i] immediately after loc_e_latest;
(16)             end if
(17)       end if
(18)   end for
(19) end
```

Algorithm 1: GenScheduling (δ, svs).

number of shared variables (#SV), and the origin (Origin) are summarized. It involves large-scale (LOC>10,000), middle-scale (10,000>LOC>1000), and small programs (LOC<1000). The program scales in terms of LOC vary from 73 for Critical to 17,596 for SpecJBB-2005. The number of shared variables is obtained using escape analysis [52]. Specifically, the number of threads or shared variables is not integer by accident. The reasons are that (1) the results were averaged over 50 runs for each program and (2) different paths may be chosen during program execution due to the natural character of dynamic analysis and the dynamic thread creation of Java. In addition, each subject has at least a concurrency bug. For example, Critical has 16 data races and 14 atomicity violations.

To evaluate the effectiveness and efficiency of our approach, we compared it with the state-of-the-art approach named SimTrace. Concretely, we designed three groups of experiments to validate the following three questions:

(1) Effectiveness: how many context switches can be reduced in trace simplification for CAOM?

(2) Efficiency: how much time does it consume in trace simplification for CAOM?

(3) Comparison: does CAOM perform better than SimTrace?

The experiments are conducted on a Samsung notebook running 64-bit Ubuntu-14.04 and jdk1.7 with 3.06 GHz Intel Core 4 processor and 4 GB memory. We utilize Soot to instrument bytecode programs. To collect original traces and replay concurrency bugs, we employ the existing record-replay tool LEAP [54]. We first use random testing to generate an original buggy trace for each subject. All the results are averaged over running 50 times.

5.2. Experimental Results and Analysis. Experimental evaluation is conducted in terms of effectiveness, efficiency, and comparison to answer the above three questions, respectively.

Profile 1 (Effectiveness). The effectiveness of trace simplification technique can be shown by the reduction of context switches. For better understanding, CAOM preserves all the program information; that is, we do not conduct any delete operations to subjects.

Table 2 lists the number of threads (#Thread), the length of original trace (Size), the number of context switches in the original buggy trace (#CSori), the number of context switches in the simplified trace (#CSsim), and the reduction (Reduction(%)), where the length of trace is the total number of synchronization operations and memory accesses. As CAOM does not conduct any delete operations for subjects, the length of trace stays the same before and after simplification. However, we can see that context switches are reduced obviously. The context switches in the simplified trace are reduced by 27.36%~99.97% (54.39% averaged) compared to that of the original buggy trace. Specifically, for the large-scaled subject SpecJBB-2005, the context switches are reduced from 124200.3 to 37.6, and the reduction is 99.97%. Besides, we can find that the more threads and more synchronization operations or memory accesses there are in the original trace, the higher the reduction we can get, such as Manager, Tsp, Cache4j, and SpecJBB-2005.

Profile 2 (Efficiency). The efficiency of trace simplification technique can be shown by the time consumption. This can affect whether it can be applied in practice. The time consumption of CAOM consists of three perspectives: data loading, dependence relations calculation, and scheduling generation. As CAOM is an offline approach and the original

Table 1: Experimental subjects.

Program	LOC	#Thread	#SV	Origin
Critical	73	4.3	1.4	IBM ConTest benchmark suit [53]
Account	148	3.0	4.0	IBM ConTest benchmark suit
Loader	148	4.0	2.0	IBM ConTest benchmark suit
Manager	212	4.4	3.0	IBM ConTest benchmark suit
BuggyProgram	385	4.0	5.0	IBM ConTest benchmark suit
ReadersWriters	103	4.0	4.1	SIR (http://sir.unl.edu/content/sir.php.)
Tsp	709	5.0	12.0	SIR
StringBuffer	1320	3.0	5.0	Suns JDK 1.4.2
LinkedList	5979	3.0	15.0	Suns JDK 1.4.2
ArrayList	5866	3.0	5.0	Suns JDK 1.4.2
Cache4j	3897	4.0	5.0	[40]
SpecJBB-2005	17,596	4.0	116.0	SPEC's benchmark (http://www.spec.org/web2005.)

Table 2: Experimental results I: effectiveness.

Program	#Thread	Size	#CSori	#CSsim	Reduction(%)
Critical	4.3	40.1	6.1	4.5	27.36
Account	3.0	73.0	8.1	4.9	39.31
Loader	4.0	64.0	4.2	3.0	27.88
Manager	4.4	1.4 K	110.1	11.9	**89.19**
BuggyProgram	4.0	228.5	6.5	4.2	36.00
ReadersWriters	4.0	327.3	7.4	3.4	54.59
Tsp	5.0	1001 K	24.0	5.5	**76.92**
StringBuffer	3.0	97.0	3.2	2.0	37.50
LinkedList	3.0	427.2	3.8	2.0	46.81
ArrayList	3.0	334.0	3.0	2.0	33.33
Cache4j	4.0	1190 K	140.0	22.7	**83.78**
SpecJBB-2005	4.0	1148 K	124 K	37.6	**99.97**

buggy trace is collected using instrument and record in the preprocessing step, the complete trace information needs to be loaded before starting simplification.

Table 3 lists experimental results in terms of the time consumption. Columns 4-7 represent the time consumed in data loading, dependence calculation, scheduling generation, and the total time, respectively. As we can see, for the 12 Java multithreaded programs, the maximum time consumed is no more than 30 min, which indicates good efficiency of our method. For example, for Tsp whose length of trace is 1001 K, the total simplification time is only 2.7 min.

Concretely, for most middle-scaled and small programs, the time is mainly consumed in data loading. For example, for ArrayList and Loader, the time consumed in data loading accounts for 84.49% and 84.16% of their total time, respectively. However, for Tsp and Cache4j, the time consumed in dependence relation calculation and scheduling generation is far more than that of data loading. The reason is that there are many synchronization operations, memory accesses, and context switches in the original trace, which leads to the fact that the dependence relations are much more complex; then, frequent check and move operations have to be conducted. Specifically, for Cache4j, the time consumed in dependence relation calculation is more than that of scheduling generation. The reason is that there are many lock dependence relations in which two locks are adjacent and belong to the same thread, saving many move operations.

For large-scaled programs, the time consumed in trace simplification increases because of the large trace size and a vast amount of context switches. However, our method still has good efficiency as it does not conduct multiple iterations and replay validation. For example, for SpecJBB-2005, the time consumption for the whole simplification is less than 30 min.

Profile 3 (Comparison Analysis). To evaluate that our approach performs better than the state-of-the-art approaches, we compared CAOM with SimTrace. Both SimTrace and CAOM reduce trace simplification problem to the combinatorial optimization problem. The difference is that SimTrace takes it as an optimization problem with graph merging.

Comparison results between SimTrace and CAOM on the reduction of context switches are presented in Figure 4. Figure 4 shows that, for most programs, CAOM can reduce more context switches compared with SimTrace. For example, for BuggyProgram, CAOM increases the reduction by 14.15%.

Table 3: Experimental results II: efficiency.

Program	LOC	Size	Data loading(ms)	Dependence calculation(ms)	Scheduling generation(ms)	Total time(ms)
Critical	73	40.1	37.6	2.3	2.7	45.7
Account	148	73.0	30.9	1.7	2.3	37.5
Loader	148	64.0	25.5	1.4	1.7	30.3
Manager	212	1.4 K	46.6	6.5	20.1	75.2
BuggyProgram	385	228.5	26.9	1.5	3.1	33.4
ReadersWriters	103	327.3	36.6	2.1	3.9	44.9
Tsp	709	1001 K	6.0E+03	7.1E+04	8.6E+04	1.63E+05
StringBuffer	1320	97.0	26.4	1.5	1.7	31.5
LinkedList	5979	427.2	32.6	1.9	3.0	39.1
ArrayList	5866	334.0	30.5	1.7	2.1	36.1
Cache4j	3897	1190 K	6.0E+03	2.39E+05	2.9E+04	2.74E+05
SpecJBB-2005	17,596	1148 K	8.0E+03	8.1E+04	9.31E+05	1.020E+06 (<30 min)

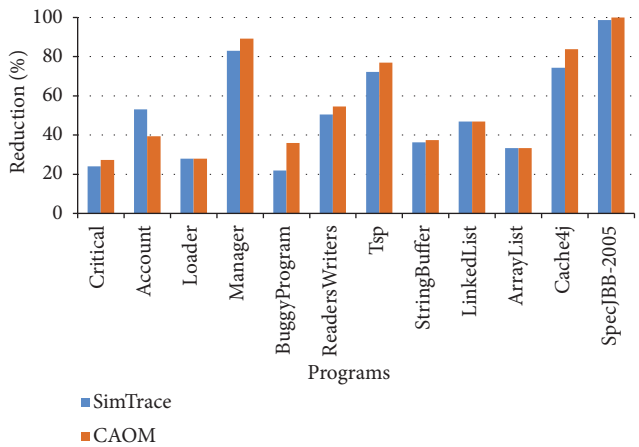
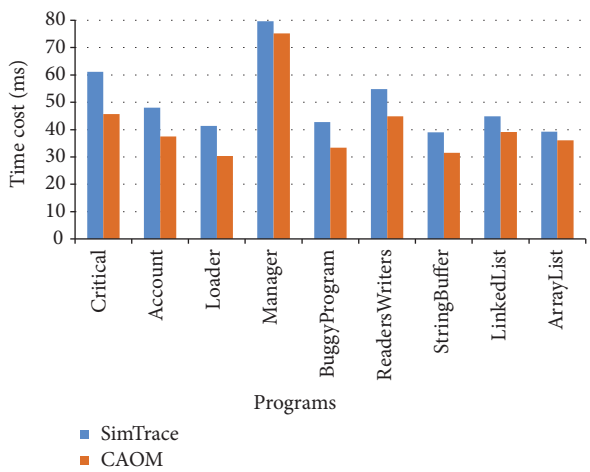
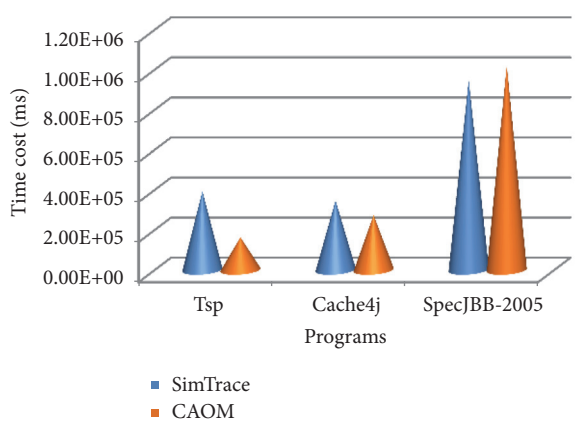


Figure 4: Comparison results between SimTrace and CAOM on reduction of CSs.



(a)



(b)

Figure 5: Comparison results between SimTrace and CAOM on time cost.

However, there is an exception to the overall result. For Account, our approach reduces the reduction by 13.83%. The reason is that SimTrace mainly pursuits performance improvement, which leads to a risk of replay failure. This situation cannot exist in our approach as we calculate the strict dependence relations between two successive events and simplify the original buggy trace in the promise of replay.

Comparison results between SimTrace and CAOM on time cost are shown in Figure 5. We separate three programs (shown in Figure 5(b)) from 12 subjects to show their time

cost as they cost too much time in the process of trace simplification. Figure 5 shows, for all the middle-scaled and small programs, CAOM is superior to SimTrace in terms of efficiency, because our method does not need to construct any abstract models. For example, for Tsp, CAOM improves the performance by 58.60%. But for SpecJBB-2005, CAOM is inferior to SimTrace. The reason is that the context switches are significantly reduced by our method, which implies that it suffers from frequent check and move operations, consuming too much time.

To sum up, we can conclude the following points based on the above experiments, which can also answer three questions in Section 5.1.

(1) CAOM is effective in trace simplification. It can reduce context switches by 27.36%~99.97% (54.39% on average).

(2) CAOM is efficient in trace simplification. Even for the large-scaled programs, it can finish the simplification within 30 min.

(3) CAOM performs better than SimTrace on both reduction of context switches and time cost.

5.3. Time Complexity Analysis. The time complexity of CAOM is $O(n^2)$, where n represents the number of events in the original trace. Given the original buggy trace, we first calculate the dependence relations for each event, whose time complexity is $O(n)$. Then, in scheduling generation, for each event, we need to search for the location of its dependence node and the latest node in the same thread, whose time complexity is $O(n^2)$. Dependence relation calculation and scheduling generation are executed in sequence. Therefore, with the above analysis, we can conclude that the time complexity of our method is $O(n^2)$.

5.4. Discussion. Based on our experiments, we can find that CAOM can simplify concurrent program execution traces with a length of million magnitudes effectively and efficiently, which is helpful to be applied in practice. Specifically, in theory, our method can provide enlightenment for the new optimization algorithms design about concurrency bug diagnosis service. In practice, experimental results show that CAOM can use less time but reduce more context switches, which implies that developers can save much time on debugging concurrent programs. Thus, our proposed method can speed up the concurrent software development.

However, there are still a few directions that may further improve our method. First, for efficiency, we only considered a one-way checking and moving. A bidirectional or a two-stage simplification [55] may further improve the effectiveness and reduce more context switches. Second, in the step of preprocessing, we used escape analysis to identify the shared variables. Both Static-TSA and Dynamic-TSA [56] are more precise and efficient than escape analysis. Moreover, they are scalable to real-world large multithreaded applications. Next, we will employ Static-TSA or Dynamic-TSA to improve the availability of our method. Third, for completeness, we calculated the dependence relations of all the events, which consumed much redundant time. We plan to utilize programming slicing [57] or Collaborative Filtering [58, 59] to extract the critical variables and the corresponding events.

6. Conclusions

The advance of cloud computation and big data applications accelerates current software development and produces various concurrency cloud services in the distributed cloud environment. However, it is a great challenge to guarantee both service quality and service performance. Concurrency bug diagnosis service is to reason about vulnerabilities in concurrent applications. The existing trace simplification techniques are either online analysis or based on the complex graph structures, which limits the performance of service optimization. In this paper, we present a novel static constraint-aware optimization method for concurrency bug diagnosis service in the distributed cloud environment. We obtain a simplified trace by iteratively checking dependence constraints and moving forward feasible events if the condition is satisfied. With the constraint-aware idea, we can guarantee that the simplified trace is equivalent to the original buggy trace. Furthermore, the effectiveness and efficiency of trace simplification have been significantly improved as we optimized concurrency bug diagnosis service offline without any complex structures. Finally, through a set of experiments conducted on 12 widely used java projects, we further demonstrate that our proposed CAOM outperforms other state-of-the-art approaches.

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 regarding the publication of this paper.

Acknowledgments

This paper is partially supported by Natural Science Foundation of China [No. 61673384 and No. 61502497].

References

[1] D. Savchenko, N. Ashikhmin, and G. Radchenko, "Testing-as-a-service approach for cloud applications," in *Proceedings of the IEEE/ACM International Conference on Utility and Cloud Computing (UCC '17)*, pp. 428-429, Shanghai, China, December 2016.

[2] S. Herbold and A. Hoffmann, "Model-based testing as a service," *International Journal on Software Tools for Technology Transfer*, vol. 19, no. 3, pp. 271–279, 2017.

[3] T. Fissaa, H. Guermah, M. E. Hamlaoui, H. Hafiddi, and M. Nassar, "A synergy of semantic and context awareness for service composition in ubiquitous environment," *Computer and Information Science*, vol. 11, no. 2, pp. 88–98, 2018.

[4] L. Barakat, S. Miles, and M. Luck, "Adaptive composition in dynamic service environments," *Future Generation Computer Systems*, vol. 80, pp. 215–228, 2018.

[5] Y. Xu, L. Qi, W. Dou, and J. Yu, "Privacy-preserving and scalable service recommendation based on simhash in a distributed cloud environment," *Complexity*, vol. 2017, Article ID 3437854, 9 pages, 2017.

[6] X. Xia, J. Yu, S. Zhang, and S. Wu, "Trusted service scheduling and optimization strategy design of service recommendation," *Security & Communication Networks*, vol. 2017, Article ID 9192084, 9 pages, 2017.

[7] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[8] W. Gong, L. Qi, and Y. Xu, "Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment," *Wireless Communications & Mobile Computing*, vol. 2018, Article ID 3075849, 8 pages, 2018.

[9] Z. Ma, R. Jiang, M. Yang, T. Li, and Q. Zhang, "Research on the measurement and evaluation of trusted cloud service," *Soft Computing*, vol. 22, no. 4, pp. 1247–1262, 2018.

[10] H. Alabool, A. Kamil, N. Arshad, and D. Alarabiat, "Cloud service evaluation method-based Multi-Criteria Decision-Making: A systematic literature review," *The Journal of Systems and Software*, vol. 139, pp. 161–188, 2018.

[11] M. Tang, X. Dai, J. Liu, and J. Chen, "Towards a trust evaluation middleware for cloud service selection," *Future Generation Computer Systems*, vol. 74, pp. 302–312, 2017.

[12] L. Qi, J. Yu, and Z. Zhou, "An invocation cost optimization method for web services in cloud environment," *Scientific Programming*, vol. 2017, Article ID 4358536, 9 pages, 2017.

[13] C. Vorhemus and E. Schikuta, "Blackboard meets Dijkstra for optimization of web service workflows," *Computing Research Repository*, 2017, https://arxiv.org/abs/1801.00322.

[14] H. Cui, X. Liu, and T. Yu, "Cloud service scheduling algorithm research and optimization," *Security & Communication Networks*, vol. 2017, Article ID 2503153, 7 pages, 2017.

[15] X. Guo, S. Chen, Y. Zhang, and W. Li, "Service composition optimization method based on parallel particle swarm algorithm on spark," *Security & Communication Networks*, vol. 2017, Article ID 9097616, 8 pages, 2017.

[16] Y. Liu and X. Sun, "CaL: Extending Data Locality to Consider Concurrency for Performance Optimization," *IEEE Transactions on Big Data*, vol. 4, no. 2, pp. 273–288, 2018.

[17] G. Pinto, W. Torres, B. Fernandes, F. Castor, and R. S. M. Barros, "A large-scale study on the usage of Java's concurrent programming constructs," *The Journal of Systems and Software*, vol. 106, pp. 59–81, 2015.

[18] 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 Distributed Systems*, vol. 26, no. 5, pp. 1206–1216, 2015.

[19] 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.

[20] 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, no. 15, pp. 117–123, 2018.

[21] L. Yang, Z. Han, Z. Huang, and J. Ma, "A remotely keyed file encryption scheme under mobile cloud computing," *Journal of Network and Computer Applications*, vol. 106, no. 15, pp. 90–99, 2018.

[22] 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.

[23] Z. Cai, H. Yan, P. Li, Z.-A. Huang, and C. Gao, "Towards secure and flexible EHR sharing in mobile health cloud under static assumptions," *Cluster Computing*, vol. 20, no. 3, pp. 2415–2422, 2017.

[24] L. Qi, S. Meng, X. Zhang et al., "An Exception Handling Approach for Privacy-Preserving Service Recommendation Failure in a Cloud Environment," *Sensors*, vol. 18, no. 7, p. 2037, 2018.

[25] X. Zhang, W. Dou, Q. He et al., "LSHiForest: A Generic Framework for Fast Tree Isolation Based Ensemble Anomaly Analysis," in *Proceedings of the the 33rd IEEE International Conference on Data Engineering (ICDE'17)*, pp. 983–994, San Diego, Calif, USA, April 2017.

[26] K. Peng, V. C. M. Leung, and L. Zheng, "Intrusion detection system based on decision tree over big data in fog environment," *Wireless Communications Mobile Computing*, vol. 2018, Article ID 4680867, 10 pages, 2018.

[27] K. Peng, V. C. M. Leung, and Q. Huang, "Clustering approach based on mini batch kmeans for intrusion detection system over big data," *IEEE Access*, vol. 6, pp. 11897–11906, 2018.

[28] 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.

[29] Z. Wu, K. Lu, and X. Wang, "Surveying concurrency bug detectors based on types of detected bugs," *Science China Information Sciences*, vol. 60, no. 3, 2017.

[30] Y. Cai and W. K. Chan, "MagicFuzzer: Scalable deadlock detection for large-scale applications," in *Proceedings of the 34th International Conference on Software Engineering (ICSE '12)*, pp. 606–616, Zurich, Switzerland, June 2012.

[31] Y. Cai and Q. Lu, "Dynamic testing for deadlocks via constraints," *IEEE Transactions on Software Engineering*, vol. 42, no. 9, pp. 825–842, 2016.

[32] S. Savage, M. Burrows, G. Nelson, P. Sobalvarro, and T. Anderson, "Eraser: a dynamic data race detector for multithreaded programs," *ACM Transactions on Computer Systems*, vol. 15, no. 4, pp. 391–411, 1997.

[33] P. Kang, "Software analysis techniques for detecting data race," *IEICE Transaction on Information and Systems*, vol. E100-D, no. 11, pp. 2674–2682, 2017.

[34] C. Flanagan and S. N. Freund, "Atomizer: a dynamic atomicity checker for multithreaded programs," *Science of Computer Programming*, vol. 71, no. 2, pp. 89–109, 2008.

[35] Q. Shi, J. Huang, Z. Chen, and B. Xu, "Verifying synchronization for atomicity violation fixing," *IEEE Transactions on Software Engineering*, vol. 42, no. 3, pp. 280–296, 2016.

[36] S. Lu, S. Park, E. Seo, and Y. Zhou, "Learning from mistakes: a comprehensive study on real world concurrency bug characteristics," *ACM SIGOPS Operating Systems Review*, vol. 42, no. 2, pp. 329–339, 2008.

[37] Y. Jiang, T. Gu, C. Xu, X. Ma, and J. Lu, "CARE: cache guided deterministic replay for concurrent Java programs," in *Proceedings of the International Conference on Software Engineering (ICSE '14)*, pp. 457–467, Hyderabad, India, May 2014.

[38] S.-B. Tang, F.-L. Song, S. Zhang, D.-R. Fan, and Z.-Y. Liu, "Reducing log of dependencies based on global synchronous logic clock," *Chinese Journal of Computers*, vol. 37, no. 7, pp. 1487–1499, 2014.

[39] Y. Chen, S. Zhang, Q. Guo, L. Li, R. Wu, and T. Chen, "Deterministic replay: a survey," *ACM Computing Surveys*, vol. 48, no. 2, pp. 1–47, 2015.

[40] J. Huang and C. Zhang, "An efficient static trace simplification technique for debugging concurrent programs," in *Proceedings of the 18th International Conference on Static Analysis (SAS '11)*, pp. 163–179, Venice, Italy, September 2011.

[41] Q. Liu, W. Cai, J. Shen, Z. Fu, X. Liu, and N. Linge, "A speculative approach to spatial-temporal efficiency with multi-objective optimization in a heterogeneous cloud environment," *Security and Communication Networks*, vol. 9, no. 17, pp. 4002–4012, 2016.

[42] W.-H. Choi and K.-S. Kang, "A Study on deciding optimal price of bioinformatics services," *Journal of the Korea Safety Management and Science*, vol. 18, no. 1, pp. 203–208, 2016.

[43] M. Li, D. Subhraveti, A. R. Butt, A. Khasymski, and P. Sarkar, "CAM: a topology aware minimum cost flow based resource manager for MapReduce applications in the cloud," in *Proceedings of the 21st ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC '12)*, pp. 211–222, Delft, The Netherlands, June 2012.

[44] R. H. B. Netzer, "Optimal tracing and replay for debugging shared-memory parallel programs," in *Proceedings of the ACM/ONR Workshop on Parallel and Distributed Debugging (PADD '93)*, pp. 1–11, San Diego, Calif, USA, May 1993.

[45] M. Xu, R. Bodik, and M. Hill, "A "flight data recorder" for enabling full-system multiprocessor deterministic replay," in *Proceedings of the Annual International Symposium on Computer Architecture (ISCA'03)*, pp. 122–133, San Diego, Calif, USA, June 2003.

[46] M. Xu, M. D. Hill, and R. Bodik, "A regulated transitive reduction (RTR) for longer memory race recording," *ACM SIGARCH Computer Architecture News*, vol. 34, no. 5, pp. 49–60, 2006.

[47] S. Tallam, C. Tian, R. Gupta, and X. Zhang, "Enabling tracing of long-running multithreaded programs via dynamic execution reduction," in *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA '07)*, pp. 207–218, London, UK, July 2007.

[48] Y. Jiang, C. Xu, D. Li, X. Ma, and J. Lu, "Online shared memory dependence reduction via bisectional coordination," in *Proceedings of the 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE'16)*, pp. 822–832, Seattle, WA, USA, November 2016.

[49] N. Jalbert and K. Sen, "A trace simplification technique for effective debugging of concurrent programs," in *Proceedings of the ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE'10)*, pp. 57–66, Santa Fe, NM, USA, November 2010.

[50] J. Huang and C. Zhang, "LEAN: Simplifying concurrency bug reproduction via replay-supported execution reduction," *ACM SIGPLAN Notices*, vol. 47, no. 10, pp. 451–465, 2012.

[51] "Soot-a Java bytecode optimization framework," https://github.com/Sable/soot.

[52] J.-D. Choi, M. Gupta, M. Serrano, V. C. Sreedhar, and S. Midkiff, "Escape analysis for Java," *SIGPLAN Notices (ACM Special Interest Group on Programming Languages)*, vol. 34, no. 10, pp. 1–19, 1999.

[53] E. Farchi, Y. Nir, and S. Ur, "Concurrent bug patterns and how to test them," in *Proceedings of the 17th International Symposium on Parallel and Distributed Processing (IPDPS '03)*, pp. 1–7, Nice, France, April 2003.

[54] J. Huang, P. Liu, and C. Zhang, "LEAP: lightweight deterministic multi-processor replay of concurrent java programs," in *Proceedings of the 8th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE '10)*, pp. 385-386, Santa Fe, NM, USA, November 2010.

[55] L. Qi, X. Zhang, W. Dou, C. Hu, C. Yang, and J. Chen, "A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment," *Future Generation Computer Systems*, vol. 88, pp. 636–643, 2018.

[56] J. Huang, "Scalable thread sharing analysis," in *Proceedings of the 38th International Conference on Software Engineering (ICSE '16)*, pp. 1097–1108, Austin, TX, USA, May 2016.

[57] Z. Wu, K. Lu, X. Wang, and X. Zhou, "Collaborative technique for concurrency bug detection," *International Journal of Parallel Programming*, vol. 43, no. 2, pp. 260–285, 2015.

[58] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI '98)*, pp. 43–52, Madison, WI, USA, July 1998.

[59] L. Qi, Z. Zhou, J. Yu, and Q. Liu, "Data-sparsity tolerant web service recommendation approach based on improved collaborative filtering," *IEICE Transactions on Information & Systems*, vol. E100-D, no. 9, pp. 2092–2099, 2017.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 8180451, 13 pages
https://doi.org/10.1155/2018/8180451

Research Article

Energy-Efficient Cloudlet Management for Privacy Preservation in Wireless Metropolitan Area Networks

Xiaolong Xu,[1,2,3] Rui Huang,[1,2] Ruihan Dou,[4] Yuancheng Li,[1,2] Jie Zhang,[3] Tao Huang,[5] and Wenbin Yu[1,2]

[1] *School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing, China*
[2] *Jiangsu Engineering Centre of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing, China*
[3] *State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China*
[4] *Jinling High School, Nanjing, China*
[5] *Silicon Lake College, Kunshan, China*

Correspondence should be addressed to Jie Zhang; njujiezhang@gmail.com

Received 30 April 2018; Accepted 1 August 2018; Published 24 September 2018

Academic Editor: David Megias

Currently, cloudlet based wireless metropolitan area network (WMAN) is emerging as an effective paradigm to extend the performance of mobile devices, which enables the execution of computational intensive mobile applications. But the normal operation of cloudlets consumes a large amount of energy, which brings about carbon dioxide emissions, aggravation of the greenhouse effect, etc. Meanwhile the data transmission of various mobile users among cloudlets would cause leakage of personal privacy. In view of this challenge, we propose an energy-efficient cloudlet management method, named ECM, for privacy preservation in WMAN. Technically, an optimization model is designed to formalize our problem. Then, based on live virtual machine (VM) migration technique, a corresponding privacy-aware VM scheduling method for energy saving is designed to determine which VMs should be migrated and where they should be migrated. Finally, experimental data demonstrate that the proposed method is both efficient and effective.

1. Introduction

In recent years, wireless metropolitan area network (WMAN) is emerging as a public network, which covers the metropolis and provides abundant services for human beings in urban cities [1]. Mobile devices in WMAN ubiquitously access rich resources from remote data centers via access points (APs). With the popularity and development of mobile devices, the demands of mobile users for wireless broadband are constantly increasing. WMAN is useful and powerful to satisfy such increasing resource requirements [2, 3].

In practice, as the service requirements of mobile users increase, mobile applications, such as interactive gaming, virtual reality, and natural language processing, are becoming more computation-intensive. But the computing capacity of mobile devices remains limited, due to the service experience of the mobile devices about weight, size, and battery life [4]. A powerful and effective way to extend the performance of mobile devices is to offload some loads on the mobile devices to the remote clouds for execution. Most existing researches in computation offloading tend to regard the cloud as the remote offloading destination, due to its abundant computing resources [5]. However, the cloud generally is located remotely and far away from the mobile users, which leads to the unneglectable and unavoidable communication delay between the mobile device and the remote cloud. To reduce such communication delay, cloudlets, which are some small mobile data centers deployed in WMAN, are introduced in the service framework of WMAN [6, 7].

There are a large amount of APs colocated with cloudlets in WMAN, where mobile devices are available to access the cloudlet resources easily, provided in the form of virtual machines (VMs) [8]. The obvious advantage of the cloudlet, compared to cloud, is the close physical distance between

the mobile devices and the cloudlet, which shortens the communication delay and improves the experience of mobile users. In such a cloudlet based WMAN, users can deploy computational intensive mobile applications to cloudlets nearby, such as resolving HD video, real-time large-scale online games, and real-time navigation. However, when the cloudlets deployed in WMAN are placed incorrectly or many mobile users concurrently access the cloudlet resources in a region, high communication delay which affects the experience of mobile users would be generated in WMAN [9, 10]. Therefore, it is necessary to pay attention to the communication delay for cloudlet management.

On the other hand, in order to support the efficient and effective operation of each cloudlet deployed in WMAN, a certain amount of energy is necessary to be consumed by the computing nodes and the cooling systems in the cloudlets [11, 12]. However, when the user traffic of cloudlets in some areas of urban cities is low, the usage of cloudlet resources would decrease. And if these cloudlets keep operating, excessive energy consumption continues to grow, which brings about some effects in urban cities, such as increased carbon dioxide emissions and aggravating the greenhouse effect [13, 14]. As a result, it is of great importance to implement cloudlet management for WMAN in an energy-efficient way.

Live VM migration technique provides potential and opportunities for cloudlet management to reduce energy consumption. Live VM migration is of great benefit to improve the usage of cloudlet resources, which does a great favor to reduce the number of running cloudlets and in turn achieve the energy efficiency. Consider the scenario that a cloudlet near a scenic spot is heavily loaded during the daytime, but this cloudlet has little load at night, so that these running VMs on this cloudlet could be migrated to another cloudlet and turn this cloudlet off to reduce energy consumption.

Moreover, due to the continuous evolution of WMAN and increasing diverse set of mobile users, WMAN has significant privacy problems [1]. When using the cloudlets to extend the performance of mobile devices, multiple mobile services with the personal private information could be accessed by multiple cloudlets due to the VM migrations. This information is very sensitive, as users' preferences, habits, and interests could be inferred from them. Especially in an open network such as WMAN, personal information is easily leaked with high probability. The more nodes data pass through in WMAN when the data about users' privacy is transmitted, the more possibly users' privacy could be leaked [15]. To realize privacy preservation of mobile users, it is necessary to reduce the data transmission hops due to VM migrations [16].

With these observations, it is still a challenge to realize energy-efficient resource management for cloudlets, while considering the privacy preservation due to VM migrations. In view of this challenge, a VM scheduling method for cloudlets in WMAN, to save energy and protect privacy, is proposed in this paper [17]. Our main contributions are threefold. Firstly, an optimization model for energy consumption of VM migrations in the cloudlets is designed. Secondly, taking into account the transmission delay and the possibility of privacy leakage, an energy-efficient cloudlet management method, named as ECM, is proposed to achieve the goal of energy saving and privacy preservation, through VM migrations. Specifically, ECM consists of five main steps, i.e., the destination cloudlet identification, shortest time identification, VM migration scheduling strategy designing, idle-state detection for cloudlets, and global VM0scheduling. Finally, comprehensive experiments and simulations are conducted to demonstrate the validity of our proposed method.

The structure of this paper is organized as follows. Section 2 shows the analysis of the energy consumption of cloudlets and privacy preservation in WMAN. An effective method of VM scheduling to save energy and privacy preservation is illustrated in Section 3. Section 4 presents the experiment results. Section 5 reviews the related work. Section 6 concludes the paper and gives an outlook of the future work [18].

2. Preliminary Knowledge

In this section, we introduce some formalized concepts and definitions of the cloudlet management in WMAN. Besides, we analyze the potential energy consumption and privacy preservation in the paper. Key terms and descriptions used in this paper are listed in Table 1.

2.1. Basic Concepts and Definitions. In WMAN, mobile users access the computation and storage resources through the access points (APs) and offload their computing tasks to the cloudlets for resource response [19].

Suppose that there are N cloudlets in the WMAN, denoted as $C = \{c_1, c_2, \ldots, c_N\}$ and there are M mobile tasks running on the cloudlets, denoted as $K = \{k_1, k_2, \ldots, k_M\}$. Let W be the amount of the APs deployed in the WMAN, and the AP set is denoted as $A = \{a_1, a_2, ..., a_W\}$. Figure 1 shows an example of WMAN framework with five APs (i.e., a_1, a_2, a_3, a_4, and a_5) and three cloudlets (i.e., c_1, c_2, and c_3). In this example, c_1 is connected to a_1, c_2 is connected to a_2, and c_3 is connected to a_3. Mobile users access resources from the cloudlets c_1 and c_2 via the APs a_1 and a_2. Without loss of generality, we assume that there is only one physical machine (PM) deployed in a cloudlet. Let g_n be the number of VM instances accommodated in the nth ($n = \{1, 2, .., N\}$) cloudlet c_n.

Some basic concepts for further energy-aware VM scheduling method design about cloudlet management in WMAN are presented as follows.

Definition 1 (VM instance requirement of t_m). The VM instance requirement of t_m is defined by the requested number of VM instances, denoted as r_m.

Definition 2 (composed VM for t_m). The VM instances in the same cloud for executing the same task t_m could be composed as a special VM, denoted as s_m.

As the tasks are deployed on the VMs among the cloudlets, there are M running composed VMs, denoted as $S = \{s_1, s_2, \ldots, s_M\}$.

Table 1: Key terms and descriptions.

Terms	Description
C	The set of running cloudlets, $C = \{c_1, c_2, \ldots, c_N\}$
K	The set of tasks running on the cloudlets, $K = \{k_1, k_2, \ldots, k_M\}$
A	The set of APs deployed in WMAN, $A = \{a_1, a_2, \ldots, a_W\}$
$E(t)$	The total energy consumption without VM migrations
g_n	The number of VM instances c_n ($n = \{1, 2, \ldots, N\}$) accommodates.
λ	The data transmission rate between APs
ε_n	The resource utilization of c_n ($n = \{1, 2, \ldots, N\}$)
$Y(t)$	The data transmission delay time
$\Theta_{n,i}(t)$	The running time of c_n ($n = \{1, 2, \ldots, N\}$)
r_m	The requested number of VM instances.
s_m	The VM instances in the same cloud for executing the same task t_m could be composed as a special VM

Definition 3 (the data transmission rate between Aps). When the tasks and VMs are migrated between p_w and $p_{w'}$, there is a delay limited by the bandwidth between p_w and $p_{w'}$. We call it data transmission rate, denoted as $\lambda_{w,w'}$.

2.2. Energy Consumption Analysis. The energy consumption of cloudlets contains several aspects, including the energy consumption of active VMs, the energy consumed by the idle VMs, and the baseline energy consumption of running cloudlets. These three parts of energy consumption are calculated as follows [11].

The energy consumption of active VMs at the time instant t for the nth cloudlet c_n in C is calculated by

$$VE_n^{active}(t) = \sum_{i=1}^{g_n} a_{n,i} \cdot \phi_{n,i}, \tag{1}$$

where $\alpha_{n,i}$ is the energy consumption rate of $v_{n,i}$ in active mode and $\phi_{n,i}$ is the total operating time of $v_{n,i}$.

In fact, the idle VMs with no task executions also have a baseline energy consumption. The idle time is determined by the running time of the longest running VM. The running time of the longest running VM is calculated by

$$\theta_{n,i}(t) = \max_{i=1}^{g_n} \{\phi_{n.i}\}, \tag{2}$$

Then the energy consumption of idle VMs in c_n is calculated by

$$VE_n^{idle}(t) = \sum_{i=1}^{g_n} \beta_{n,i} \cdot (\theta_{n,i}(t) - \phi_{n,i}), \tag{3}$$

where $\beta_{n,i}$ is the energy consumption rate of $v_{n,i}$ in idle mode.

When a cloudlet is active, there is at least one active VM running on it. It will consume the baseline energy. The energy for the all running cloudlets is calculated by

$$E_{cloudlet} = \sum_{n=1}^{N} \gamma_n \cdot \theta_{n,i}(t), \tag{4}$$

where γ_n is the energy consumption rate of c_n.

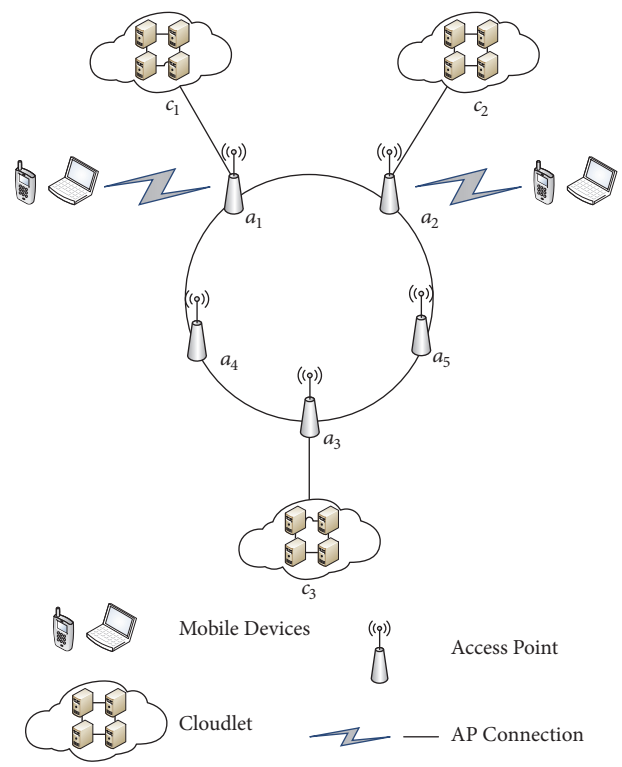


Figure 1: An example of WMAN with five APs and three cloudlets.

Through the above analysis, the total energy consumption is calculated by

$$E(t) = \sum_{n=1}^{N} VE_n^{active}(t) + \sum_{n=1}^{N} VE_n^{idle}(t) + E_{cloudlet}(t), \tag{5}$$

2.3. Resource Utilization Analysis. The resource utilization is a standard for measuring whether the resources are used sufficiently. When the resource utilization of c_n is low, the VMs in this cloudlet should be migrated to other cloudlets,

and then it could be shut down to save energy. The resource utilization of c_n which is denoted as ε_n is calculated by

$$\varepsilon_n = \frac{1}{g_n} \cdot \sum_{i=1}^{g_n} v_{n,i}(t), \tag{6}$$

where $v_{n,i}(t)$ is a binary variable employed to judge whether $v_{n,i}$ hosts a load, defined by

$$v_{n,i}(t) = \begin{cases} 0, & \text{if } \phi_{n,i} = 0, \\ 1, & \text{Otherwise.} \end{cases} \tag{7}$$

So, the average resource utilization of all the cloudlets is calculated by

$$\eta = \frac{1}{N'} \cdot \sum_{n=1}^{N} \varepsilon_n, \tag{8}$$

where N' is the number of all active cloudlets, calculated by

$$N' = \sum_{n=1}^{N} f_n(t), \tag{9}$$

where $f_n(t)$ is a binary variable employed to judge whether c_n accommodates a mobile task, defined by

$$f_n(t) = \begin{cases} 0, & \text{if } \sum_{i=1}^{g_n} \phi_{n,i} = 0, \\ 1, & \text{Otherwise.} \end{cases} \tag{10}$$

2.4. Delay and Privacy Preservation Analysis. Besides, when the data are transmitted between v_i and $v_{i'}$, there is always a delay due to the data transmission rate, which affects the experience of mobile users. The delay of all the tasks for data transmission among VMs is calculated by

$$Y(t) = \sum_{i=1}^{g_n} \sum_{i'=1}^{g_n} \frac{D_{i,i'}}{\lambda_{i,i'}}, \tag{11}$$

where $D_{i,i'}$ is the total size of data which are transferred from v_i to $v_{i'}$ and $\lambda_{i,i'}$ is the data transmission rate which is mentioned above.

When data are transmitted from one cloudlet to another, it is relayed through multiple APs. The APs in the WMAN have limited storage and computational capabilities, as well as low security, and the mobile users are vulnerable to violations while transmitting the applications and data with private personal information. An efficient and effective method to solve this problem is decreasing the number of APs that the transmitted data pass through. When there are fewer nodes which data passes through, the probability of privacy leakage is lowered; thus we could achieve the goal of privacy preservation.

3. A Cloudlet Management Method for Energy Conservation with Privacy Preservation

Based on the analysis of energy and the possibility of VM migrations for privacy preservation in WMAN in Section 2, an energy-efficient cloudlet management method is proposed in this section.

3.1. Method Overview. Technically, as specified in Box 1, our proposed cloudlet management method for energy saving and privacy preservation consists of five steps, i.e., the destination cloudlet identification, privacy-aware cloudlet confirmation, VM scheduling strategy obtaining, idle-state detection for cloudlets and global VM scheduling. The destination cloudlet identification is designed to confirm and record the active cloudlets with idle spaces, which could be employed to host the migrated VMs. These obtained cloudlets could be served as the input for the processing of Step 2. Step 2 is developed for the VMs in the source cloudlet to select the final cloudlet with the minimum data transmission time and, at the same time, achieve the purpose of privacy preservation. Step 3 is designed to produce the available migration strategies. Step 4 detects the idle state of active cloudlets in a dynamic way. All the achieved migration strategies and records are the input of the final global VM scheduling.

3.2. Destination Cloudlet Identification. When tasks are being hosted on a VM, each task may have a different requirement of idle VMs. The active cloudlets can host multiple VMs for task execution. For example, if a task requiring three virtual machines is migrated from another low-load virtual machine, the destination cloudlet must have more than or equal to three idle VMs. Otherwise, the task cannot be migrated to the currently selected cloudlet. Therefore, in Algorithm 1, we design the algorithm to find the migrated cloudlets which have the idle VMs and meet the requests. It plays an important role in cloudlet management.

Once the resource requirement of a task is responded by a VM instance, it generates a VM occupation record for the resource manager. Here, each cloudlet hosts only one PM. Let $OR = \{or_1, or_2, \ldots, or_Z\}$ be the VM occupation record collection, where Z is the number of the records in OR, defined as follows.

Definition 4 (VM occupation record or_z). or_z records the occupation status of the task deployed in the cloudlet, which is denoted as $or_z = (tid_z, cid_z, pid_z, vid_z, stim_z, dtim_z)$, where tid_z, cid_z, pid_z, vid_z, $stim_z$, and $dtim_z$ are the allocated task, the occupied cloudlet, the relevant employed PM in the cloudlet, the applied VM instance, the service start time, and the duration time, respectively.

Here we select the VMs that meet the task requirements. To better schedule the VMs in an energy-efficient way, they are classified according to VM occupation records. These records generate from the destination cloudlets once current round of migration is finished.

Firstly, we get the running cloudlet list from or_z and then sort the cloudlets in the decreasing order of idle VMs; that is, we need to sort the different cloudlets according to the load distribution of the cloudlets. Suppose we want to select d cloudlets that meet the requirements. The cloudlets with higher load, meeting the resource requirements, are selected. Once the cloudlet meets the requirement, we add it to the selected running cloudlet list, denoted as $RS = \{rs_1, rs_2, \ldots, rs_d\}$. When we traverse all the cloudlets in the

Step1: Destination cloudlet identification. Select a certain range of cloudlets with free VMs, which form a primitive cloudlet collection. This collection is identified for VM migration.
Step2: Privacy-aware cloudlet confirmation. Confirm the cloudlet with the shortest data transmission time for the source cloudlet from the cloudlet collection obtained in Step 1.
Step3: VM scheduling strategy obtaining. Migrating the tasks to the cloudlet obtained from Step 1 and Step 2. Update the migration policies and the task storage records.
Step4: Idle-state detection for cloudlets. Detect the idle state of each cloudlet, and calculate the resource utilization for the cloudlets, which is used as the migrated standard of dynamic scheduling.
Step 5: Global VM scheduling. Based on the VM migration strategies obtained in Step 3, the dynamic VM scheduling strategies for all the running VMs are achieved according to the cloudlet utilization obtained in Step 4.

Box 1: Specification of VM scheduling method for energy conservation.

Input: The acquired number q of idle VMs
VM occupation record or_z
Output: The selected running cloudlets $RS = \{rs_1, rs_2, \ldots, rs_d\}$
(1) Get the running cloudlets lists from or_z
(2) Sort the PM in cloudlets in the decreasing order of idle VMs, $PM_s = \{pm_1, pm_2, \ldots pm_N\}$
(3) $flag = 1$, $H = 0$, $n = 1$
(4) **while** $flag = 1$ **do**
(5) Get the number of idle VMs, i.e., rn
(6) **if** r_n can hosts the VMs && $H < d$ **then**
(7) Add r_n to rs_d
(8) $n = n + 1$
(9) $H = H + 1$
(10) **else if** $n == N \parallel H == d$ **then**
(11) $flag = 0$
(12) **end if**
(13) **end if**
(14) **end while**
(15) **Return** RS

Algorithm 1: Destination cloudlet identification DCI.

original list or find the number of qualified VMs that meet expectations, the searching processing would be stopped.

Algorithm 1 specifies the process of destination cloudlet identification. In this algorithm, we first sort the PMs in cloudlets in descending order of load (Lines (1) and (2)). Then we identify the cloudlets that satisfy the meets of idle VMs (Lines (3) to (14)).

3.3. Privacy-Aware Cloudlet Identification. Algorithm 1 identifies a range of cloudlets that meet the criteria. In this section, we use Dijkstra's algorithm to sort the selected cloudlets and then select the closest one to migration tasks from source cloudlet to the destination, with shortest path [20].

Besides, the processes of VM scheduling could lead to potential risks due to the privacy leakage in WMAN. In order to cater for the application scenario of WMAN, our algorithm is in demand to take privacy protection into consideration. In view of this challenge, we find that using the algorithm of shortest time identification based on Dijkstra is an appropriate method to achieve the goal of protecting users' privacy. When identifying the shortest data transmission time between source cloudlet and destination cloudlet, the data also go through the fewest APs; in other words, due to the principle of Dijkstra, we reduce the possibility of privacy leaks.

Definition 5 (data transmission time dis_{ij}). The transmission time between the cloudlets i and j is denoted as dis_{ij} ($1 \leq i < j \leq d$) according to the transmission delay among the APs of these two cloudlets.

Firstly, all the vertices should be divided into two parts: i.e., the known shortest time vertex set P_d and the unknown shortest time vertex set Q_d. Initially, only one collection of the source cloudlet is known in the collection P_d of the shortest time. Secondly, the source cloudlet C_s is set by its own shortest time as 0, that is, $dis_{Cs} = 0$, if there is a vertex i that the source point can directly reach, set dis_i to y_{si}. The cloudlet with the smallest data transmission time among the collection Q_d is selected, and it should be added to the collection P_d. And all the edges starting from this cloudlet are examined. For example, if there is an edge from C_i to C_j, then the transmission time from C_s to C_j can be extended by adding the edge $C_i \longrightarrow C_j$ to the tail. The length of this time is $dis_i + y_{ij}$. If this value is smaller than the currently known value of dis_i, we can replace the current value in dis_i with the new one.

Input: The selected running cloudlets $RS = \{rs_1, rs_2, .., rs_d\}$
The known shortest time vertex set $P = \{p_1, p_2, \ldots, p_d\}$
The unknown shortest time vertex set $Q = \{q_1, q_2, \ldots, q_d\}$
The collections y_{ij} and dis_i
The VM occupation record or_z
Output: The cloudlet with the shortest time C_{dij}
(1) Get the transmission time between each cloudlet in RS from or_z to y_{ij}
(2) Add the source cloudlet (C_s) to p and set its dis = 0
(3) **if** C_i is directly connected to the source cloudlet **then**
(4) $dis_i = y_{si}$
(5) **end if**
(6) sort Q in the increasing order of the transmission time
(7) **for** each cloudlet in Q **do**
(8) **if** $dis_i + y_{ij} < d_i$ **then**
(9) $dis_i = dis_i + y_{ij}$
(10) **end if**
(11) **end for**
(11) sort dis_i in the increasing order of the transmission time
(12) $C_{dij} = d_i$
(13) **return** C_{dij}

Algorithm 2: Privacy-aware cloudlet identification.

If the collection Q_d is empty, the algorithm ends. Then return the cloudlet with the shortest transmission time.

Algorithm 2 specifies the process of privacy-aware cloudlet identification. In this algorithm, we firstly divide all the vertices into two parts (Lines (1) and (2)); then we use Dijkstra algorithm to find the destination with the shortest time (Lines (3) to (11)). At last, we return the result (Lines (12) and (13)).

3.4. VM Scheduling Strategy Confirmation. In order to obtain the optimal energy savings, we try to migrate the tasks on the low-load cloudlets to the high-load cloudlets as much as possible, based on the results achieved by the above analysis. Once all the tasks on a cloudlet are migrated, the cloudlet could be changed to the sleep mode to save power. At the same time, a fully loaded cloudlet can also achieve higher resource utilization. And each migration procedure is recorded in the relevant VM occupation records with the migration time instant t.

Definition 6 (migration composition collection MC_d). For all the running VMs in rs_d, the synchronous VM migration strategies constitute the migration composition.

Suppose there are Z cloudlets in OR_Z and K migration compositions, i.e., $\{l_1, l_2, \ldots, l_K\}$, for each turn of migration composition generation. First, we sort the cloudlets in the decreasing order of idle VMs, which means sorting from low load to high load. Here, flag is a sign. Get cloudlets to migrate from low load to high load from the list of cloudlets.

To facilitate the energy-efficient VM scheduling, the running VM list and hosting tasks in the occupation record set ORz are obtained first. Then for each task in the list, we use Algorithms 1 and 2 to identify the most suitable cloudlet for migration and migrate the task from the high-load cloudlet to the low-load cloudlet. And once a task in a low-load VM cannot be all migrated, the temporal migration strategies should be removed. For example, if a cloudlet hosts two tasks, the first task takes up one VM instance and the other takes up two VM instances. The two VMs instances of the second task should be placed in the same target cloudlet. Therefore, we should migrate the second task first. If the second task can be migrated but the first one does not work, leave the two tasks on the original cloudlet. Otherwise, the migration is successful and the temporary migration records $c_1 \sim c_k$ should be updated to the migration record collection MC_d, respectively.

Here, after the VM migrations at the time instant t, the start time and duration of the relevant VM occupation records should be updated, according to the real occupation time. Specifically, if the migration occurs, the duration of the task on the original cloudlet is necessary to be modified. For the task that is migrated, the relevant new VM occupation records are generated, the start time is updated to t, and the duration also needs to be changed as the predicted occupation time.

Algorithm 3 specifies the process of VM scheduling strategy confirmation. We firstly sort the cloudlets in the decreasing order according to the load (Line (1)). Then we move the tasks on the low-load cloudlets to the high-load cloudlets and store the records in the list migration record MC_d (Lines (2) to (21)).

3.5. Idle-State Detection for Cloudlets. The idle-state detection of active cloudlets is also one of the important factors for global migration considerations. It is a significant standard for the global VM scheduling. When the next migration is going to be conducted, we should evaluate the idle-state of

Input: The running cloudlet set rs_d; VM occupation record or_z
VM occupation record or_z
Output: The migration composition collection MC_d
(1) Sort the cloudlets in the decreasing order of idle VMs
(2) $flag = 0, z = 1, K = 1$
(3) **while** $flag = 0$ **do**
(4) Get the running VM list in ORz
(5) Get the hosting task list in ORz
(6) **for** each hosted task **do**
(7) Get the number of occupied VMs
(8) Select the destination cloudlet by Algorithms 1 and 2
(9) **end for**
(10) **if** all the VMs can be migrated away **then**
(11) Generate a migration composition l_K
(12) Update the cloudlet status and VM distribution
(13) **else** $flag$=1
(14) Delete the temporal migration strategies
(15) **end if**
(16) $z = z + 1, K = K + 1$
(17) **end while**
(18) **for** the reserved tasks' records **do**
(19) $dtim_k = t$
(20) **end for**
(21) **for** $k = 1$ *to* K **do**
(22) Add l_1~l_k as a migration composition to MC_d
(23) Get the start time and time of each migration task
(24) $dtim_k = dtim_k - t$
(25) $stim_k = t$
(26) Update the start time and duration time in MC_d
(27) Update task storage record cs_x
(28) **end for**
(29) **Return** MC_d

Algorithm 3: VM scheduling strategy confirmation.

the active cloudlets, to confirm and record the idle space before the migration time t. After detecting the idle space, the current cloudlet resource utilization before the migration time could be achieved.

As the running time changes dynamically, some tasks would be competed over time. In this section, we detect the idle state of cloudlets as time progress. Therefore, to find the available destination cloudlet, the idle space and resource utilization of the all the running cloudlets should be detected first.

Once the migration strategy is designed in Algorithm 3, it generates one or more VM occupation records based on the migration records for the resource manager. So, we should also update the task storage records before the migration based on the latest migration strategy. Suppose there are x tasks that still need to be hosted. Let $CS = \{cs_1, cs_2, \ldots, cs_x\}$ be the task occupation record collection, where x is the number of the records in CS, defined as follows.

Definition 7 (task storage record cs_x). cs_x records the storage status of the tasks deployed in the cloudlet, whose number is x, start time is denoted as $stim_x$, and duration time is denoted as $dtim_x$, which is dynamically changed.

The running cloudlet list from the task storage record cs_x is gotten first as the main data. Secondly, all the active tasks are traversed and the cloudlet idle space of all the cloudlets needs to be updated accordingly. Then, use the formula (9) to calculate the number of idle VM instances and the amount of the active cloudlets. At last, RP is returned as the final results of the idle spaces of the definite cloudlet.

Algorithm 4 specifies the process of idle-state detection of cloudlets. Firstly, we update the cloudlet idle space (Lines (1) to (6)). Then we calculate the resource utilization and return it (Lines (7) and (8)).

3.6. Global VM Scheduling. To achieve the goal of energy-efficient cloudlet management in WMAN, the optimal VM migration composition should be confirmed from the migration strategies which is achieved by Algorithm 3. In this section, global VM scheduling is designed to determine which VMs and where should be migrated for the whole execution period of the VMs.

Due to the goal of scheduling all the running VMs from a global perspective in this section, migration operation time is necessary for monitoring the resource utilization of all the cloudlets through the calculating of the current VM

Input: The task storage record cs_x
Migration time t
Output: The current cloudlet resource utilization RP
(1) Get running cloudlet list from the cs_x of time t
(2) **for** each hosted task **do**
(3) Get the number of occupied VMs
(4) **end for**
(5) Update the cloudlet idle space
(6) end for
(7) Calculate the current cloudlet resource utilization RP by (9)
(8) Return RP

Algorithm 4: Cloudlet idle-state detection.

Input: The VMs $V = \{v1, v2, \cdots, vz\}$
Migration time t
Total operation time H
Output: The cloudlets scheduling policy
(1) Get the migration time t;
(2) Get the number of idle VMs on each cloudlets
(3) **while** $t < H$ **do**
(4) Get RP from Algorithm 4
(5) **if** $RP <$ utilization threshold level **then**
(6) Update the migration time t as the next finish time of the VMs.
(7) Get migration compositions MC_h by Algorithm 3
(8) Schedule the VMs by the policy
(9) Set the vacant cloudlets to the sleeping mode
(10) **end if**
(11) **end while**

Algorithm 5: Global VM scheduling.

occupation records. The number of idle VMs of each cloudlet can be detected dynamically. Algorithm 4 is designed to acquire the idle space as the migration standard which is one of the inputs in Algorithm 5.

As we aim to schedule the VMs with the goals of energy saving and privacy preservation, the VM migration composition with highest utilization will be chosen as the ultimate VM scheduling policy. And after VM scheduling for the identified VMs, the vacant PMs should be set to the sleeping mode to save the energy. Besides, through the Algorithm 2, we can also attain another purpose for reducing the possibility of privacy leaks.

The migration times are necessary to be obtained for global VM scheduling according to the utilization threshold. Besides, the number of the idle VMs on each cloudlet is a key measure to decide the destination cloudlet of the migrations. During the execution period of all the mobile applications in WMAN, the resource utilization RP of the cloudlets is obtained by Algorithm 4. Then, if the current RP is less than utilization threshold, that is, there are enough space in the active cloudlets after the previous round of scheduling, we need to schedule the VMs by the policy and set the vacant cloudlets to the sleeping mode according to MC_h by Algorithm 3. Thus, we achieve the goal of dynamic scheduling.

Algorithm 5 specifies the process of global VM scheduling (t). Firstly, we update the RP and $MC_h$of the migration time (Lines (1) to (6)). Then schedule the running VMs and set the vacant cloudlets to the sleeping mode (Lines (8) to (10)).

4. Experiment Evaluation

In this section, a series of comprehensive experiments are operated to evaluate the performance of our proposed energy-efficient cloudlet management method for privacy preservation in WMAN. For comparison analysis, the running state without migration is marked as a benchmark is employed to verify the effectiveness of our proposed ECM method.

4.1. Experimental Context. In our experiment, we select the specified type of cloudlet servers to create our simulated cloud infrastructure services net and the basic hardware parameter are as follows. Its configuration consists of Intel Xeon 3040, dual-Processor clocked at 1860 MHz, and 4 GB of RAM. Its baseline power consists of two parts, the PM part and the VM part. The power consumption of the PM, active VM, and idle VM are 86W, 6W, and 4W, respectively.

Six basic parameters are initialized as records in our experiment. The value domain of each parameter is specified

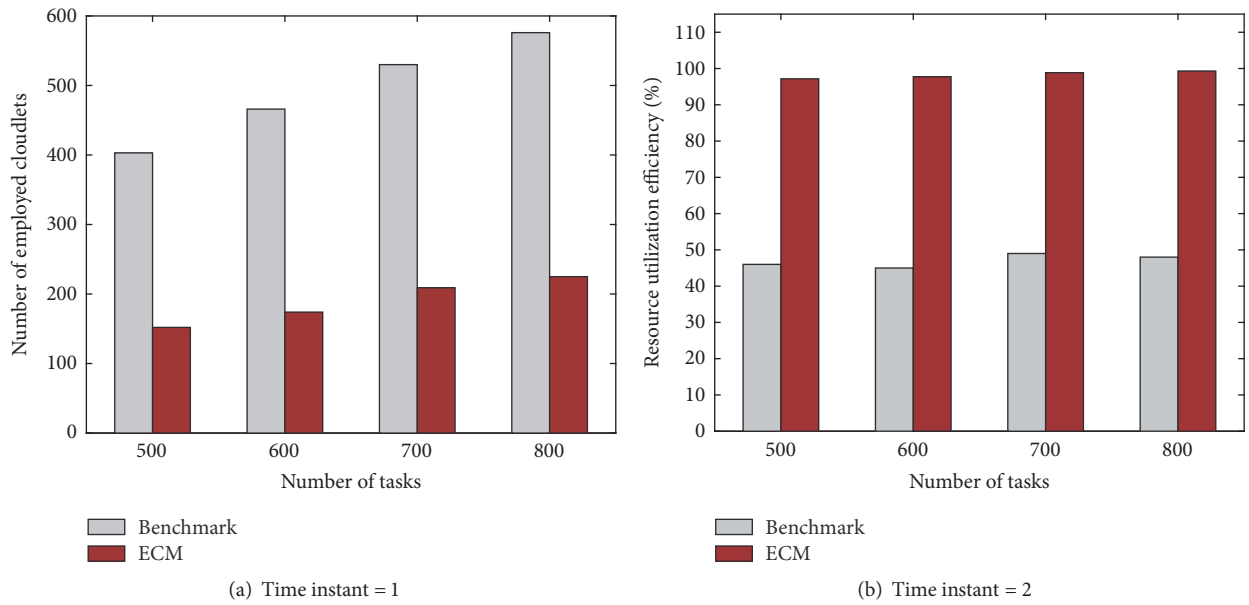


Figure 2: Comparison of the number of employed cloudlets for different time instants with Benchmark and ECM.

Table 2: Parameter settings.

Parameter Item	Domain
Number of running PMs	$\{500, 600, 700, 800\}$
Number of running VMs in each PM	$[1, 3]$
Number of VMs on each PM	6
Data transmission rate (Mb/s)	$[270, 540]$
VM duration time	$\{1, 3\}$
VM transmission data (G)	$[0.5, 0.8]$

in Table 2. In other words, 4 different-scale datasets are generated, respectively, for our experiment with the number of running tasks 500, 600, 700, and 800. Each data record in these datasets is a tuple with 6 attributes. For example, there is a data record (6, 18, 2, 0, 3, and 2), where "6" is the sequence number of the hosted task, "18" is the cloudlet sequence number, "2" is the number of VMs that this hosted task simultaneously occupies, "0" is the VM start time, "3" is the VM duration time, and "2" is the total amount of data for this hosted task.

In our experiment, there are two rounds of multiple VM migration, and we use time instants before migration to distinguish different round of VM scheduling. The two time instants are recorded as 1 and 2 in the experimental section, respectively.

4.2. Performance Evaluation. In this section, evaluations on energy consumption, resource utilization, and transmission delay are discussed to validate our proposed method.

4.2.1. Performance Evaluation on Energy Consumption. Our energy consumption evaluation is based on the energy consumption of the running cloudlets and VMs. The number of active employed cloudlets is also significant to the total energy consumption. We used four different sets of data to compare energy consumption in the case of benchmark and ECM. And we use two time instants to compare the employed number of different time instants in the process of dynamic scheduling. From Figure 2, it can be concluded that, with the ECM method, the number of running cloudlets is much less than that in the benchmark case in both two time instants. Since the idle cloudlet can be directly considered to be completely closed, its energy consumption can be ignored in our experiment when setting them to the sleeping mode. Thus, less active cloudlet can generate less energy. The energy consumption here is divided into three parts, the idle VMs, the active VMs, and the energy consumed by active cloudlets themselves.

After two rounds of VM migrations, the total energy consumption can be calculated respectively. Figure 3 indicates these energy values in the case of benchmark and ECM. Table 3 shows the results of improvements of energy consumption with ECM compared to benchmark. From the analysis of Figure 3 and Table 3, using our proposed method ECM saves more energy than using benchmark.

4.2.2. Performance Evaluation on Resource Utilization. Resource utilization efficiency for two different time instants is also one of the dimensions that measure the value of our proposed method. Resource utilization changes dynamically in different rounds of VM migrations because when the resource utilization of one cloudlet is low, the VMs in this cloudlet should be migrated to other cloudlets, and then it could be shut down to save energy. It refers to the relationship between VMs and active cloudlets. Similar to the energy consumption evaluated above, resource utilization is also tested by four different datasets. The ECM and benchmark are compared with these four datasets, and the simulated resource utilization result is illustrated in Figure 4.

Table 3: Improvements of energy consumption with ECM compared to benchmark.

	Number of servers			
ECM	500	600	700	800
	54.4%	54.1%	41.9%	52.1%

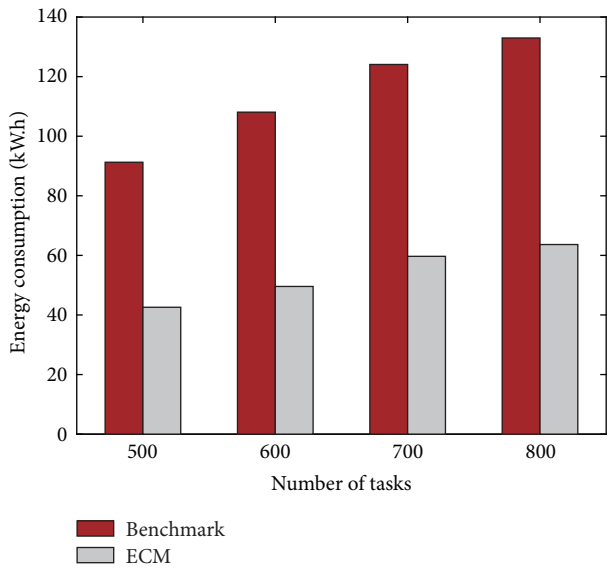


Figure 3: Comparison of total energy consumption with Benchmark, ECM.

As can be shown in the chart, after VM migrations, our proposed method ECM has better resource utilization than benchmark. And, with the increasing times of VM scheduling, the resource utilization for cloudlets is getting higher. Since ECM considers the energy consumption by cloudlets, some VMs would not be migrated to other cloudlets and the cloudlet hosting it still keeps in active mode, which would raise the resource utilization. It shows that, with our method for energy saving, the resource utilization becomes higher than the benchmark.

4.2.3. Performance Evaluation on Transmission Delay. Figure 5 illustrates the observation results of the amount of the migrations after VM scheduling. According to the system model, the transmission delay is determined based on the amount of data and the data transmission rate. At the same time, the number of VM migrations is also one of the important factors of total transmission delay. Compared to benchmark, our proposed method ECM uses a large amount times of VM migrations to achieve the goal of saving energy which would generate high transmission delay.

Furthermore, to make the evaluation on performance of our proposed ECM method more clearly, the effect of VM migrations on transmission delay is fully investigated in detail. The transmission delay of our proposed ECM method is based on four different data sets and two time instants for simulation evaluation, respectively.

The result which illustrates transmission delay time caused by the VM migrations is evaluated. From Figure 6, simulation results can be intuitively achieved that the transmission delay based on the ECM method fluctuates around 10 minutes in the simulation environment. The result shows that there is still a lot of room for improvement in handling challenges of high transmission delay.

Besides, from the results of evaluations on VM migrations, we can achieve that using the shortest time identification can indeed reduce the number of migrations; in other words, we can also achieve the goal of reduces the possibility of privacy leaks.

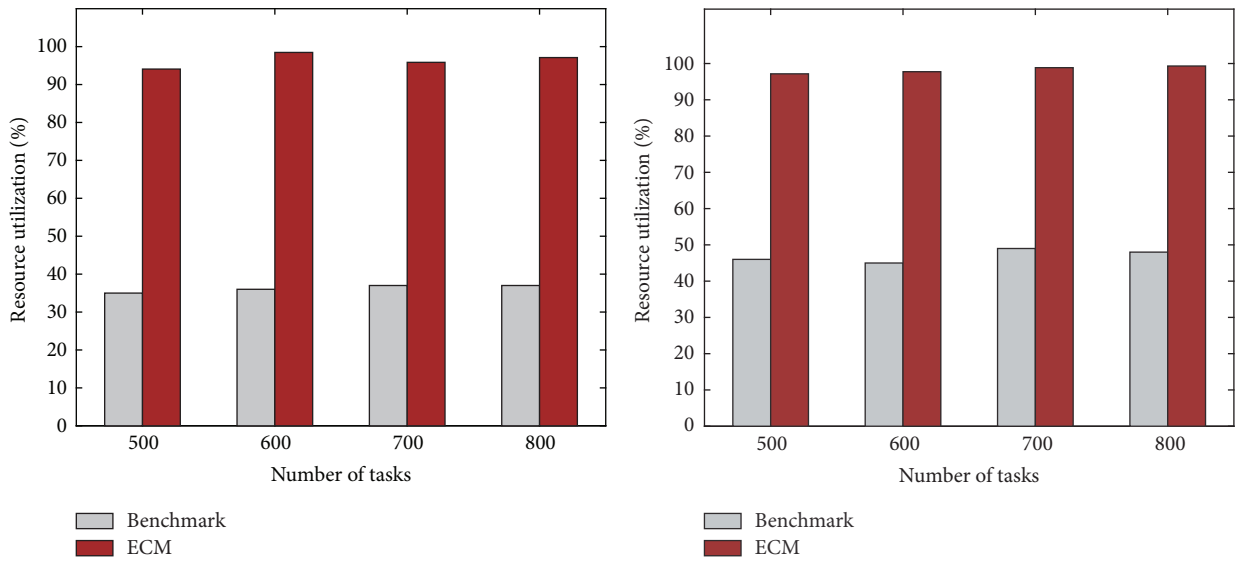
5. Related Work

Recently, the emergence and development of WMAN aim to meet the growing market requirements of wireless broadband. An increasing number of mobile users in urban cities have tendency to accomplish the execution of various tasks in clouds due to the abundant resources, but one significant limitation of offloading tasks to the remote clouds is the long physical distance between mobile users and remote cloud [21–23]. To reduce such delay of offloaded tasks to remote clouds, cloudlets are deployed in WMAN to support mobile devices, by executing offloaded tasks on local cloudlets.

Ma et al. [24] realized the placement of cloudlets in WMAN is important and proposed a New Heuristic Algorithm (NHA) to reduce the sorting process of APs running time to solve the delay problem. Xiang et al. [25] proposed an adaptive cloud placement method for GPS big data mobile applications that adjusts the placement of the cloud by identifying the aggregated area of mobile devices. Mike Jia et al. [26] proposed an algorithm of allocating the requirements of mobile users to suitable cloudlets deployed in WMAN to implement load balancing to solve the delay problem and discussed how their algorithm can be practically applied to WMANs with dynamic and constantly moving users.

Currently, live VM migration techniques significantly make contribution to manage resource in cloudlet, thus improving energy efficiency and effectively lower the delay in cloudlets as much as possible via providing high quality of resources for execution of tasks and migrating VMs to idle cloudlets to reduce task queuing time.

Decreasing the delay in cloudlets could be divided into two aspects: decreasing transmission delay between cloudlets and decreasing processing delay in cloudlets [27]. But Rodrigues et al. [28] reduced the two kinds of delay at the same time by utilizing a mathematical model with a Particle Swarm Optimization (PSO) model. Dhanoa et al. [29] quantified the energy consumption and time in the VM migration process and mathematically modeled them and then discovered that the main factors of energy consumption during VM migration are the size of the VMs and the network bandwidth. Xu et al. [30] proposed a VM scheduling



(a) Time instant = 1

(b) Time instant = 2

Figure 4: Comparison of resource utilization for different time instants with Benchmark, ECM.

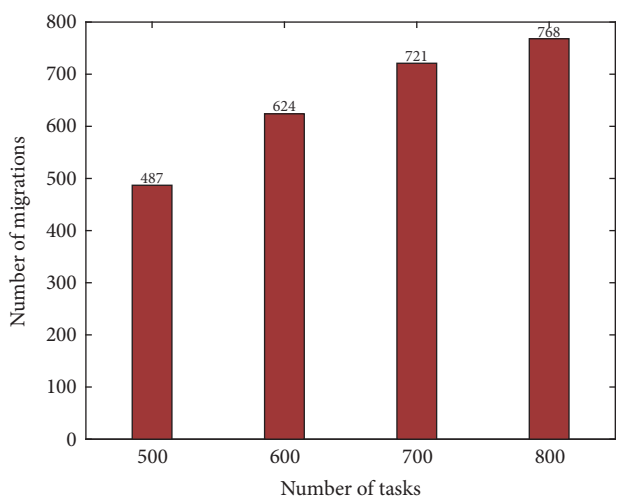


Figure 5: Number of migrations with ECM.

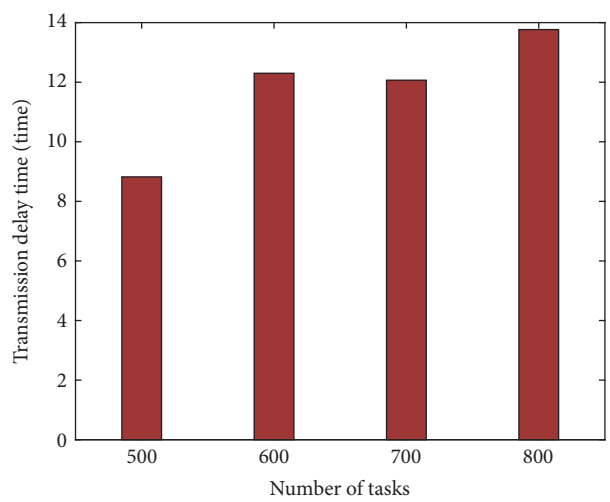


Figure 6: Transmission delay time of ECM.

method to achieve balance between energy and performance in cyberphysical cloud systems and the method determines which VMs and where should be migrated. Shaw et al. [31] proposed a method to save energy by using a VM migration dynamic merge to reduce the number of running physical machines and the authors designed an algorithm to determine if the VM really needs to continue running or if it needs to be migrated.

The optimization problem of energy consumption for cloudlet management could be solved by the heuristic methods and the convex optimization technique [32, 33]. Besides, along with the other scheduling objectives, the game theory with Nash Equilibria is employed to realize the competition usage of the cloudlet resources [34, 35]. However, some cloudlet services need to access mobile users' personal private information. When transmitting the mobile applications of the mobile users, personal privacy is easily leaked through some APs with poor security, which may cause some bad events and even hurt the personal safety of mobile users. Therefore, privacy preservation could be treated as a constraint for the optimization problem in cloudlet management [36–39].

Maria et al. [40] proposed a conceptual framework in Ambient Intelligent focused in cloud computing that includes the various privacy policies and privacy issues. Ji et al. [41] implemented a hybrid privacy protection solution with KP-ABE and CP-ABE, whose advantage is balancing the performance and security in cloud platforms. Wang et al. [42] developed a method to protect privacy by dividing user's private information into several parts and then this

system saves the data in different cloud servers and user's fog devices. Moreover, when the data itself is insecure, which may include trust spoof attacks and replicated-sink attacks, the data collection needs to identify whether such data could be received. Wang et al. [43] proposed a trust evaluation scheme to ensure the trustworthiness of sensors and mobile sinks, which protects the system from trust spoof attacks and replicated-sink attacks.

The above analysis concludes that live VMs migration technique has not been well used in the cloudlets researches and few studies have considered privacy preservation as much as possible while reducing energy consumption about cloudlets deployed in WMAN.

6. Conclusion and Future Work

The utilization of cloudlet in WMAN is an important technology which extends the performance of mobile devices. As WMAN continues to develop, cloudlets which are public and easy to access would be vital to mobile cloud computing.

In this paper, we proposed an energy-efficient cloudlet management for privacy preservation in WMAN abbreviated as ECM taking advantage of the live VM migration techniques. We implemented an effective algorithm that determines the place to which VMs should be migrated. Our goal is to reduce the energy consumption of cloudlets and privacy preservation in WMAN. Experimental evaluations have been conducted to validate the efficiency and effectiveness of our proposed method. As this paper mainly focused on the energy saving and privacy in WMAN, there are also many important issues which need to be investigated in future. For example, in the people-intensive areas, there are large amounts of mobile applications that need to be offloaded to the cloudlets for execution; how to reduce the energy consumption and achieve load balancing of the cloudlets in WMAN remains challenging. When we only pay attention to energy saving, the communication delay would be high, which affects the experience of mobile users.

For future work, we try to take cloudlet load balancing and energy consumption of cloudlets into consideration at the same time. Furthermore, we will design a corresponding method for trade-offs between energy consumption and load balancing.

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 research is supported by the National Science Foundation of China (under Grant nos. 61702277, 61672276, 61501247, 61772283, 61402167, and 61672290), the Key Research and Development Project of Jiangsu Province (under Grant nos. BE2015154 and BE2016120), and the Natural Science Foundation of Jiangsu Province (under Grant no. BK20171458). Besides, this work is also supported by The Startup Foundation for Introducing Talent of NUIST, the open project from State Key Laboratory for Novel Software Technology, Nanjing University under Grant no. KFKT2017B04, the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD) fund, Jiangsu Collaborative Innovation Center on Atmospheric Environment and Equipment Technology (CICAEET), and the project "Six Talent Peaks Project in Jiangsu Province" under Grant no. XYDXXJS-040.

References

[1] S. B. M. Baskaran and G. Raja, "Blind key distribution mechanism to secure wireless metropolitan area network," *CSI Transactions on ICT*, vol. 4, no. 2-4, pp. 157–163, 2016.

[2] H. M. T. Al-Hilfi, A. H. Najim, and A. M. Alsahlany, "Evaluation of WIMAX network performance of Baghdad city using different audio codecs," in *Proceedings of the 16th RoEduNet Conference: Networking in Education and Research, RoEduNet '17*, pp. 1–5, 2017.

[3] J. Cheng, R. Xu, X. Tang et al., "An abnormal network flow feature sequence prediction approach for DDoS attacks detection in big data environment," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 95–119, 2018.

[4] M. Satyanarayanan, P. Bahl, R. Cáceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[5] V. Meena, M. HariPrasath, V. Kalpana, K. ArunKumar, and J. SenthilKumar, "Optimal resource reservation for offloaded tasks in mobile cloud computing," in *Proceedings of the 2nd International Conference on Communication and Electronics Systems (ICCES)*, pp. 677–682, 2017.

[6] M. Jia, W. Liang, Z. Xu, and M. Huang, "Cloudlet load balancing in wireless metropolitan area networks," in *Proceedings of the 35th Annual IEEE International Conference on Computer Communications, IEEE INFOCOM '16*, pp. 1–9, 2016.

[7] L. Li, F. Meng, and P. Ju, "Some new integral inequalities and their applications in studying the stability of nonlinear integro-differential equations with time delay," *Journal of Mathematical Analysis and Applications*, vol. 377, no. 2, pp. 853–862, 2011.

[8] F. Teka, C.-H. Lung, and S. Ajila, "Seamless Live Virtual Machine Migration with Cloudlets and Multipath TCP," in *Proceedings of the 39th IEEE Annual Computer Software and Applications Conference, COMPSAC '15*, vol. 2, pp. 607–616, 2015.

[9] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Capacitated cloudlet placements in wireless metropolitan area networks," in *Proceedings of the 40th Conference on Local Computer Networks, LCN '15*, pp. 570–578, 2015.

[10] S. Guan, R. E. De Grande, and A. Boukerche, "A Cloudlet-based task-centric offloading to enable energy-efficient mobile applications," in *Proceedings of the IEEE Symposium on Computers and Communications, ISCC '17*, pp. 564–569, 2017.

[11] X. Xu, W. Dou, X. Zhang, and J. Chen, "An energy-aware resource allocation method for scientific workflow executions in cloud environment," *IEEE Transactions on Cloud Computing*, vol. 4, no. 2, pp. 166–179, 2016.

[12] M. S. Shinde and K. M. Ashtankar, "Effect of different shapes of conformal cooling channel on the parameters of injection

molding computers," *Computers, Materials & Continua*, vol. 54, no. 3, pp. 287–306, 2018.

[13] T. Wang, M. Z. A. Bhuiyan, G. Wang, M. A. Rahman, J. Wu, and J. Cao, "Big data reduction for a smart city's critical infrastructural health monitoring," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 128–133, 2018.

[14] C. Wu, E. Zapevalova, Y. Chen, and F. Li, "Time optimization of multiple knowledge transfers in the big data environment," *Computers, Materials & Continua*, vol. 54, no. 3, pp. 269–285, 2018.

[15] Z. Jia, X. Wei, H. Guo, W. Peng, and C. Song, "A privacy protection strategy for source location in WSN based on angle and dynamical adjustment of node emission radius," *Journal of Electronics*, vol. 26, no. 5, pp. 1064–1072, 2017.

[16] C. Jinhua, Z. Yuanyuan, C. Zhiping, L. Anfeng, and L. Yangyang, "Securing display path for security-sensitive applications on mobile devices," *Materials & Continua*, vol. 55, no. 1, pp. 17–35, 2018.

[17] L. Yuling, H. Peng, and J. Wang, "Verifiable Diversity Ranking Search Over Encrypted Outsourced Data," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 037–057, 2018.

[18] L. Donghui, Z. Guozheng, X. Zeng, L. Yong et al., "Modelling the Roles of Cewebrity Trust and Platform Trust in Consumers' Propensity of Live-Streaming: An Extended TAM Method," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 137–150, 2018.

[19] L. Wenyan, L. Xiangyang, L. Yimin, L. Jianqiang, L. Minghao, and Q. S. Yun, "Localization Algorithm of Indoor Wi-Fi Access Points Based on Signal Strength Relative Relationship and Region Division," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 71–93, 2018.

[20] J.-Q. Cai and H. Li, "An implicit degree condition for relative length of long paths and cycles in graphs," *Acta Mathematicae Applicatae Sinica*, vol. 32, no. 2, pp. 365–372, 2016.

[21] Y. Cao, Z. Zhou, X. Sun, and C. Gao, "Coverless information hiding based on the molecular structure images of material," *Computers, Materials and Continua*, vol. 54, no. 2, pp. 197–207, 2018.

[22] C. Yuan, X. Li, Q. M. J. Wu, J. Li, and X. Sun, "Fingerprint liveness detection from different fingerprint materials using convolutional neural network and principal component analysis," *Computers, Materials and Continua*, vol. 53, no. 4, pp. 357–371, 2017.

[23] J. Kaur and K. Kaur, "A fuzzy approach for an IoT-based automated employee performance appraisal," *Computers, Materials and Continua*, vol. 53, no. 1, pp. 24–38, 2017.

[24] L. Ma, J. Wu, and L. Chen, "Fast algorithms for capacitated cloudlet placements," in *Proceedings of the 21st IEEE International Conference on Computer Supported Cooperative Work in Design, CSCWD '17*, pp. 439–444, 2017.

[25] H. Xiang, X. Xu, H. Zheng et al., "An adaptive cloudlet placement method for mobile applications over GPS big data," in *Proceedings of the 59th IEEE Global Communications Conference, GLOBECOM '16*, pp. 1–6, 2016.

[26] M. Jia, J. Cao, and W. Liang, "Optimal Cloudlet Placement and User to Cloudlet Allocation in Wireless Metropolitan Area Networks," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 725–737, 2017.

[27] T. Wang, J. Zeng, Y. Lai et al., "Data collection from WSNs to the cloud based on mobile Fog elements," *Future Generation Computer Systems*, 2017.

[28] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "A PSO model with VM migration and transmission power control for low Service Delay in the multiple cloudlets ECC scenario," in *Proceedings of the 2017 IEEE International Conference on Communications, ICC '17*, pp. 1–6, 2017.

[29] I. S. Dhanoa and S. S. Khurmi, "Analyzing energy consumption during VM live migration," in *Proceedings of the International Conference on Computing, Communication and Automation, ICCCA '15*, pp. 584–588, 2015.

[30] X. Xu, X. Zhang, M. Khan, W. Dou, S. Xue, and S. Yu, "A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems," *Future Generation Computer Systems*, 2017.

[31] S. B. Shaw and A. K. Singh, "Use of proactive and reactive hotspot detection technique to reduce the number of virtual machine migration and energy consumption in cloud data center," *Computers and Electrical Engineering*, vol. 47, pp. 241–254, 2015.

[32] P. Wang and L. Zhao, "Some geometrical properties of convex level sets of minimal graph on 2-dimensional Riemannian manifolds," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 130, pp. 1–17, 2016.

[33] P. Li, S. Zhao, and R. Zhang, "A cluster analysis selection strategy for supersaturated designs," *Computational Statistics & Data Analysis*, vol. 54, no. 6, pp. 1605–1612, 2010.

[34] J. Zhang, B. Qu, and N. Xiu, "Some projection-like methods for the generalized Nash equilibria," *Computational Optimization and Applications*, vol. 45, no. 1, pp. 89–109, 2010.

[35] B. Liu, B. Qu, and N. Zheng, "A successive projection algorithm for solving the multiple-sets split feasibility problem," *Numerical Functional Analysis and Optimization*, vol. 35, no. 11, pp. 1459–1466, 2014.

[36] H. Wu, Y. Ren, and F. Hu, "Continuous dependence property of BSDE with constraints," *Applied Mathematics Letters*, vol. 45, pp. 41–46, 2015.

[37] S. Lian and Y. Duan, "Smoothing of the lower-order exact penalty function for inequality constrained optimization," *Journal of Inequalities and Applications*, 2016.

[38] Y. Wang, X. Sun, and F. Meng, "On the conditional and partial trade credit policy with capital constraints: A Stackelberg Model," *Applied Mathematical Modelling*, vol. 40, no. 1, pp. 1–18, 2016.

[39] H. Li and S. Wang, "Partial condition number for the equality constrained linear least squares problem," *Calcolo. A Quarterly on Numerical Analysis and Theory of Computation*, vol. 54, no. 4, pp. 1121–1146, 2017.

[40] M. Del Mar López Ruiz and J. Pedraza, "Privacy risks in cloud computing," in *Intelligent Agents in Data-Intensive Computing*, vol. 14, pp. 163–192, Springer International Publishing, 2015.

[41] J. Yi-mu, K. Jia-bang, L. Hai et al., "A privacy protection solution for the balance of performance and security in cloud computing," in *Proceedings of the International Conference on Web Information Systems Engineering – WISE 2013 Workshops*, pp. 335–345, 2014.

[42] T. Wang, J. Zhou, M. Huang et al., "Fog-based storage technology to fight with cyber threat," *Future Generation Computer Systems*, vol. 83, pp. 208–218, 2018.

[43] T. Wang, Y. Li, W. Fang et al., "A comprehensive trustworthy data collection approach in sensor-cloud system," *IEEE Transactions on Big Data*, p. 1, 2018.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 8087303, 10 pages
https://doi.org/10.1155/2018/8087303

Research Article

RoughDroid: Operative Scheme for Functional Android Malware Detection

Khaled Riad [1,2] and Lishan Ke [3]

[1] *School of Computer Science, Guangzhou University, Guangzhou 510006, China*
[2] *Mathematics Department, Faculty of Science, Zagazig University, Zagazig 44519, Egypt*
[3] *College of Mathematics and Information Science, Guangzhou University, Guangzhou 510006, China*

Correspondence should be addressed to Lishan Ke; kelishan@gzhu.edu.cn

Received 11 June 2018; Accepted 6 August 2018; Published 20 September 2018

Academic Editor: Lianyong Qi

There are thousands of malicious applications that invade Google Play Store every day and seem to be legal applications. These malicious applications have the ability to link the malware referred to as Dresscode created for network hacking as well as scrolling information. Since Android smartphones are indispensable, there should be an efficient and also unusual protection. Therefore, Android smartphones usually continue to be safeguarded from novel malware. In this paper, we propose *RoughDroid*, a floppy analysis technique that can discover Android malware applications directly on the smartphone. *RoughDroid* is based on seven feature sets ($FS_1, FS_2, \ldots, FS_7$) from the XML *manifest* file of an Android application, plus three feature sets (FS_8, FS_9, and FS_{10}) from the Dex file. Those feature sets pass through the Rough Set algorithm to elastically classify the Android application as either benign or malicious. The experimental results mainly consider 20 most common malware families, plus three new malware families (*Grabos, TrojanDropper.Agent.BKY,* and *AsiaHitGroup*) that invade Google Play Store at 2017. According to the experimental results, *RoughDroid* has 95.6% detection performance for the malware families at 1% false-positive rate. Finally, *RoughDroid* is a lightweight approach for straightly examining downloaded applications on the smartphone.

1. Introduction

The world's most preferred mobile operating system currently is Android OS. Android surpasses Windows as the globe's most preferred OS, yet some Android applications have been discovered to privately swipe individual details from various other applications. Recently, the GadGet Hacks website stated that, after evaluating 110,150 Android applications over a duration of 3 years, the scientists located countless sets of applications that can possibly leakage delicate phone or individual information as well as permitting unapproved applications to hack the blessed information. With numerous thousands of applications in various markets, Android OS offers riches of capability to its customers. Smart devices running Android are progressively targeted by assaulters as well as contaminated with destructive software programs [1].

Google took down over 700,000 bad Android applications in 2017, that is, 70% more than in 2016 [2]. In addition to the existing malware families, three new Android malware families (*Grabos*, *TrojanDropper.Agent.BKY*, and *AsiaHitGroup*) invade Google Play Store at 2017 [3]. It appears that there is an urgent requirement for quitting the expansion of malware on Android markets and also smartphones. The Android platforms constantly attempt as well to supply numerous security solutions that stop the installment of malware applications, most significantly the Android authorization system. To carry out particular tasks on the Android device, such as capturing a picture, the application needs to clearly ask for consent from the individual throughout the setup procedure. Some customers thoughtlessly approve the installment agreement to unidentified applications without thoroughly reviewing it.

As a result, malicious software is hardly constricted from the Android permission program in training. Opening your Android phone or tablet as much as for applications and video games outside Google's protective walled yard likewise makes your device considerably a lot more at risk to malware. It is the cost you spend for a totally free software programs [4]. There

should be a method that is able to restrict these malicious applications.

1.1. Motivation. To the very best of our understanding, a huge body of research study has actually examined approaches for evaluating and also discovering Android malware applications before their setup. These approaches could be approximately classified right into techniques making use of dynamic as well as static evaluation. There are some techniques that could keep track of the habits of applications at run-time, such as TaintDroid [5], DroidRanger [6], and DroidScope [7] which are techniques that could check the actions of applications at run-time. Although run-time monitoring is really reliable in determining harmful task, it experiences a substantial cost and could not be straight used for mobile devices. On the other hand, static evaluation techniques, such as Stowaway [8] and RiskRanker [9], typically generate just a tiny run-time overhead. While these methods are scalable and also reliable, they mostly improve the handcrafted discovery patterns, which are commonly not readily available for new malware circumstances. This is behind our motivation to propose a new Android malware detection scheme that makes it possible to recognize malware straight on the smartphone throughout the setup process based on Rough Set algorithm.

1.2. Main Contributions. In this paper, we introduced *RoughDroid* that is a new broad floppy analysis malware detector on smart Android phones during the installation time by introducing robust feature extraction framework. The main contributions could be summed up as follows:

(i) **Effective Detection:** We introduce a novel scheme (*RoughDroid*) for combining floppy analysis and machine learning that is capable of identifying Android malware with high accuracy and few false alarms. Also, it is independent of manually crafted detection patterns.

(ii) **Various Features:** *RoughDroid* groups numerous features from the *manifest* file as well as *application's Dex code*. Those features are categorized into ten feature sets ($FS_1, FS_2, \ldots, FS_{10}$).

(iii) **Rough-Based Detection:** The proposed scheme considers the *adware* Android applications during the detection of malware applications. This is due to executing the detection process elastically using *Rough Set* algorithm that introduces flexible (not straight line) classification into benign and malware applications.

(iv) **Lightweight Analysis:** For efficiency, we apply linear time analysis and learning techniques that enable detecting malware on the smartphone as well as analyzing large sets of applications in a reasonable time.

Finally, the experiments with 131,611 applications and 5,560 malware samples, in addition to 158 malware applications introducing three new malware families at 2017, demonstrate the efficacy of our method for directly checking downloaded applications on the smartphone.

1.3. Organization. The rest of this paper is organized as follows: Section 2 introduces our *RoughDroid* scheme with its ten feature sets and Rough Set detection algorithm. Section 3 presents the experimental evaluation of *RoughDroid* by comparing it with some popular detection schemes and ten of the most common antiviruses. Section 4 introduces the related work and smooth comparison between the currently proposed Android malware detection schemes and *RoughDroid*. This is followed by the conclusion in Section 5.

2. RoughDroid

In this paper, we present *RoughDroid*, a lightweight technique for discovering Android malware that presumes discovery patterns immediately. In addition, it allows recognizing malware straight on the smartphone. *RoughDroid* performs a broad floppy analysis, gathering as numerous features from an *application's code* as well as *manifest* as feasible. These features are organized in groups of strings (for instance, features API calls along with network speeches) and embedded within a combined vector space. As an example, an application sending out premium SMS messages is cast to a particular area in the vector room connected with the equivalent consent, intents, and also API calls. This geometric depiction allows *RoughDroid* to recognize mixes as well as patterns of features indicative for malware automatically, by utilizing machine learning techniques.

To this end, our technique utilizes a broad floppy analysis that extracts feature sets from various resources and examines these features in a meaningful vector space. This procedure is shown in Figure 1 and also described as follows:

(I) **Floppy Analysis:** *RoughDroid* floppily inspects a given Android application and various feature collections from the application's *manifest* and also disassembled *Dex code*. *RoughDroid* inspects the application's *manifest* and disassembled *Dex code* of a given Android Application in *Parallel Sweep* to reduce the time of analysis.

(II) **Constructing Vector Space:** The extracted feature sets are after that mapped to a joint vector space, where patterns and also mixes of the features could be evaluated geometrically.

(III) **Rough-Based Detection:** The embedding of the feature sets allows us to recognize malware utilizing effective strategies of machine learning (Rough Set algorithm).

2.1. Floppy Analysis. As the primary step, *RoughDroid* carries out a lightweight floppy analysis of an offered Android application. The floppy extraction of features should run in a constricted environment and in full prompt way. The customer might avoid the recurring procedure, if the evaluation takes so long time. Appropriately, it becomes vital to pick features that can be extracted effectively. We therefore focus on the *manifest* in addition to the disassembled *Dex code* of this application, which could be obtained with a parallel sweep within the application's content. To enable an extensible as well as common evaluation, we represent all extracted

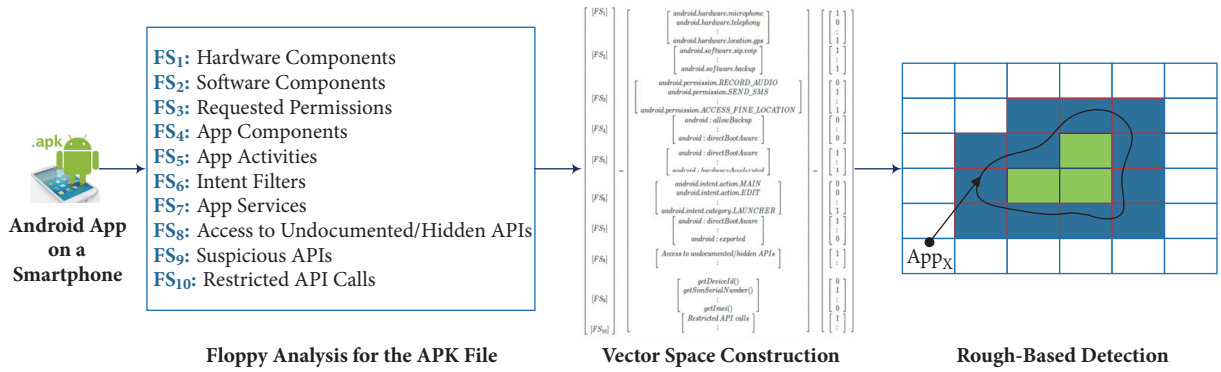


Figure 1: General overview for the proposed *RoughDroid* system model.

features as sets of strings, such as authorizations, intents, and also API calls. Specifically, we extract the adhering ten sets of features.

2.1.1. Manifest Feature Sets. It is an effective file in the Android system that defines the performance and also demands of an application to Android. AndroidManifest.xml could be located at the root of the project and has numerous various feature sets.

A simple XML *manifest* file generated for an Android application tested under *RoughDroid* is shown in Figure 2. The presented XML file declares seven different features sets (FS_1, FS_2, FS_3, FS_4, FS_5, FS_6, and FS_7) as follows:

FS_1 :***Hardware Components:*** It has the requested hardware features by an application. The figure indicates three requested hardware features (microphone, telephony, and location.gps). An application that has access to GPS and network modules is, for instance, able to collect fine location information and send it to an attacker over the network.

FS_2 :***Software Components:*** It indicates that the application utilizes or requires software features. The figure declares sip.voip that allows the application to use Session Initiation Protocol (SIP) services and do VOIP calls.

FS_3 :***Requested Permissions:*** It is very important for Android security mechanisms. The figure mentions three dangerous permissions (RECORD_AUDIO, SEND_SMS, and ACCESS_FINE_LOCATION) that are granted to the application during the application's setup time by the user.

FS_4 :***App Components:*** It is a set of Boolean expressions that grant some services to the application, such as allowBackup and directBootAware.

FS_5 :***App Activities:*** It allows the application to execute a specific activity, such as directBootAware and hardwareAccelerated.

FS_6 :***Intent Filters:*** It specifies the types of intents that an activity, service, or broadcast receiver can respond to, such as action.MAIN and action.EDIT.

FS_7 :***App Services:*** It represents a service as one of the application's components, such as directBootAware and exported.

The information saved in AndroidManifest.xml file could be effectively obtained on the device by making use of the *Android Asset Packaging Tool* that allows us to extract out the previously mentioned sets of features.

2.1.2. Disassembled Code Feature Sets. We implement a lightweight disassembler, which takes as input the Dalvik Executable (*Dex*) and provides *RoughDroid* with the complete information about API calls and the data utilized in the application. The *Dex* file contains a set of class definitions and their associated adjunct data. Table 1 introduces a simple example for the *Dex* file that is enhanced bytecode for the Dalvik virtual machine. Every Android application has a unique classes.dex file, which references any type of approaches or courses utilized within an application. Basically, any type of task, things, or piece utilized within the codebase will certainly be changed right into bytes within a *Dex* file that could be run as an Android application.

We are mainly interested in the API calls and method calls, because they can be easily extracted from the *Dex* file of an application, as follows:

FS_8 :***Access to Undocumented/Hidden APIs:*** Applications could be limited from accessing APIs that are undocumented in the Android Software Development Kit (SDK). *RoughDroid* looks for the incident of these demands in the *Dex* file, in order to get a further understanding of the behavior of an application.

FS_9 :***Suspicious APIs:*** Requesting some delicate information or sources of the Android phone might result in destructive behavior. We are laying more importance to a set of such suspicious APIs:

(i) **Sensitive data (*IMEI* and *USIMnumbeleakage*) APIs**, where the Android requests are such as *getDeviceId()*, *getSimSerialNumber()*, and *getImei()*;

(ii) **Network communication APIs**, such as *setWifiEnabled()* and *execHttpRequest()*;

Table 1: A simple Dex file has code which is eventually performed by the Android Runtime.

(a)

No.	Dex File Magic		Checksum	SHA1-Hash Signature					File Size	Header Size
1	6465780A	30333800	7A44CBBB	FB4AE841	0286C06A	8DF19000	3C5DE024	D07326A2	E0010000	70000000
2	00000000	02000000	9C000000	01000000	AC000000	14010000	CC000000	E4000000	EC000000	07010000
3	00000000	01000000	01000000	00000000	00000000	FFFFFFFF	00000000	57010000	00000000	01000100
4	706C6963	6174696F	6E3B0023	4C636F6D	2F627567	736E6167	2F646578	6578616D	706C652F	42756773

(b)

No.	Endian Constant	Link Size & Offset	Map Section	String ID	Type ID	Photo ID	Field ID	Method ID	Class Def	Data Size
1	78563412	00000000	00000000	64010000	05000000	70000000	03000000	84000000	01000000	90000000
2	2C010000	2F010000	01000000	02000000	03000000	03000000	02000000	00000000	00000000	00000000
3	01000000	00000000	04000000	70100000	00000E00	063C696E	69743E00	194C616E	64726F69	642F6170
4	6E616741	70703B00	01560026	7E7E4438	7B226D69	6E2D6170	69223A32	362C2276	65727369	6F6E223A

(iii) **Location leakage APIs**, such as *getLastKnownLocation(), getLatitude(), getLongitude(),* and *requestLocationUpdates();*

(iv) **Sending and receiving SMS/MMS messages APIs**, such as *sendTextMessage(), SendBroadcast(),* and *sendDataMessage()*.

FS_{10} :***Restricted API calls:*** The Android authorization system limits accessibility to a collection of crucial API calls. Our approach looks for the event of these calls that represent a apart of the *Dex code*, in order to get a much deeper understanding of an App's capability.

$$V(App) = \begin{bmatrix} [FS_1] \\ [FS_2] \\ [FS_3] \\ [FS_4] \\ [FS_5] \\ [FS_6] \\ [FS_7] \\ [FS_8] \\ [FS_9] \\ [FS_{10}] \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} android.hardware.microphone \\ android.hardware.telephony \\ \vdots \\ android.hardware.location.gps \end{bmatrix} \\ \begin{bmatrix} android.software.sip.voip \\ \vdots \\ android.software.backup \end{bmatrix} \\ \begin{bmatrix} android.permission.RECORD\_AUDIO \\ android.permission.SEND\_SMS \\ \vdots \\ android.permission.ACCESS\_FINE\_LOCATION \end{bmatrix} \\ \begin{bmatrix} android:allowBackup \\ \vdots \\ android:directBootAware \end{bmatrix} \\ \begin{bmatrix} android:directBootAware \\ \vdots \\ android:hardwareAccelerated \end{bmatrix} \\ \begin{bmatrix} android.intent.action.MAIN \\ android.intent.action.EDIT \\ \vdots \\ android.intent.category.LAUNCHER \end{bmatrix} \\ \begin{bmatrix} android:directBootAware \\ \vdots \\ android:exported \end{bmatrix} \\ \begin{bmatrix} Access\ to\ undocumented/hidden\ APIs \\ \vdots \end{bmatrix} \\ \begin{bmatrix} getDeviceId() \\ getSimSerialNumber() \\ \vdots \\ getImei() \end{bmatrix} \\ \begin{bmatrix} Restricted\ API\ calls \\ \vdots \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \\ \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \\ \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \\ \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \\ \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \\ \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \\ \begin{bmatrix} 1 \\ \vdots \\ 0 \end{bmatrix} \\ \begin{bmatrix} 1 \\ \vdots \end{bmatrix} \\ \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \\ \begin{bmatrix} 1 \\ \vdots \end{bmatrix} \end{bmatrix} \quad (1)$$

2.2. Vector Space Construction. A harmful task is normally shown in particular patterns as well as mixes of the extracted features. As an example, a malware application sending the fine location of a smartphone might have the permission *android.permission.ACCESS_FINE_LOCATION* in FS_3 and the hardware feature *android.hardware.location.gps* in FS_1.

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:versionCode="1"
    android:versionName="1.0"
    package="com.example.IntentApp">
    <uses-sdk android:minSdkVersion="15" android:targetSdkVersion="26" />

    <!-- FS1: Hardware Components -->
    <uses-feature android:name="android.hardware.microphone" />
    <uses-feature android:name="android.hardware.telephony" />
    <uses-feature android:name="android.hardware.location.gps" />

    <!-- FS2: Software Components -->
    <uses-feature android:name="android.software.sip.voip" />
    <uses-feature android:name="android.software.backup" />

    <!-- FS3: Requested Permissions -->
    <uses-permission android:name="android.permission.RECORD_AUDIO"/>
    <uses-permission android:name="android.permission.SEND_SMS"/>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>

    <!-- FS4: App Components -->
    <application
        android:allowBackup="true"
        android:directBootAware="true">

        <!-- FS5: App Activities -->
        <activity
            android:directBootAware="true"
            android:hardwareAccelerated="true"/>

        <!-- FS6: Intent Filters -->
        <intent-filter>
             <action android:name="android.intent.action.MAIN" />
             <action android:name="android.intent.action.EDIT" />
             <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>

        <!-- FS7: App Services -->
        <service android:name=".IntentService"
         android:directBootAware="true"
         android:exported="true">
        </service>
    </application>
</manifest>
```

Figure 2: A simple XML *manifest* example (AndroidManifest.xml) that declares seven feature sets for an Android application.

Preferably, we would like to create Boolean expressions that catch this reliance in between features as well as returning true if a malware is found.

We will need to place the extracted feature collections from an Android application $(FS_1, FS_2, \ldots, FS_{10})$ in a vector. In our experiments the vector space $(V(App))$ contains approximately 550,000 different extracted features. If the application (App) contains the feature (f), the vector space element for that feature is mapped to 1 $(V(App, f) = 1)$; otherwise, it is mapped to 0 $(V(App, f) = 0)$. A simple structure as an example of the vector space is shown in (1). Regardless of the measurement of the vector space, it is hence enough to just save the extracted features from an application for sparsely standing for the vector $V(App)$ by using either hash tables [10] or Bloom filters [11].

2.3. Rough-Based Detection. Rough Set based data analysis [12–14] starts after constructing the vector space (feature table), as depicted in Section 2.2. Each row represents a specific feature obtained from a certain feature set according to a specific Android application in our scheme. The Rough system has multiple entities and stages.

(i) **Feature Table:** It is a pair $FS = (Apps, F)$ where $Apps$ is a nonempty finite set of Android applications called the universe and F is a nonempty finite set of features such that $f : Apps \longrightarrow Vf$ for every $f \in F$. The set Vf is called the value set of f, and elements of $Apps$ are called Android applications.

(ii) **Decisions:** It is the feature table in the form $FS = (Apps, F \cup \{app_f\})$, where App_f (not a feature in F) is the decision feature. The features of F are called conditional features or simply conditions.

(iii) **Approximations:** Let $App_X \subseteq Apps$:

(a) **Lower Approximation:** It consists of all Android applications, which definitely belong to $\underline{R}(App_X) = \{App \in Apps \mid [App]R \subseteq App_X\}$.

(b) **Upper Approximation:** It contains all Android applications, which possibly belong to $\overline{R}(App_X) = \{App \in Apps \mid [App]R \cap App_X \neq \phi\}$.

(c) **Boundary Region:** The difference between the upper and lower approximations constitutes the boundary region of the Rough Set algorithm. Boundary positive and negative regions [15] are described as below.

$$BND_R(App_X) = \left|\overline{R}(App_X X) - \underline{R}(App_X)\right|$$
$$POS_R(App_X) = \underline{R}(App_X) \tag{2}$$
$$NEG_R(App_X) = U - \overline{R}(App_X)$$

An Android application of the negative region $NEG_R(App_X)$ does not belong to App_X, an application of the positive region $POS_R(App_X)$ belongs to App_X, and only one application of the boundary region $BND_R(App_X)$ belongs to App_X. Those approximation sets and regions are shown in Figure 3.

(d) **Approximation Accuracy:** The roughness precision of any subset $App_X \subseteq Apps$ with regard to $R \subseteq F$, represented as $\alpha_R(App_X)$, is quantified by $\alpha_R(App_X) = |\underline{R}(App_X)/\overline{R}(App_X)|$, where $|App_X|$ represents cardinality of App_X. For an empty set ϕ, we define $\alpha_R(\phi) = 1$. It is worth noting that $0 \leq \alpha_R(App_X) \leq 1$. If $\alpha_R(App_X) = 1$, the set App_X is crisp with respect to R. If $\alpha_R(App_X) < 1$, App_X is tough with reference to R.

3. Evaluation

After providing *RoughDroid* thoroughly, we currently continue to an empirical assessment of its efficiency. In order to do so, we first describe the used dataset and then run some experiments to evaluate the detection performance.

3.1. Considered Data Sets. Our experiments are executed based on a dataset of genuine Android applications and also actual malware. We are utilizing Drebin dataset [1], it comprises 131,611 software samples collected from other tools, including Google Play Store, both Chinese and Russian Markets, and also Android sites. Additionally, it Includes 5,560 malware applications, from 179 distinct malware families, for example, *FakeInstaller, DroidKungFu, Plankton, Opfake, GingerMaster, BaseBridge, Iconosys, Kmin, FakeDoc, Geinimi, Adrd, DroidDream, Linux/Lotoor, GoldDream, MobileTx, FakeRun, SendPay, Gappusin, Imlog,* and *SMSreg*.

Table 2: Statistical measures for *RoughDroid* and four other schemes.

Scheme	Area	Std. Error	Asymptotic Prob.	95% LCL	95% UCL
RCP [16]	0.52123	0.08965	5.63972E-5	0.43983	0.62718
Peng et al. [17]	0.08182	0.85211	4.941212E-5	0.51643	0.73156
SigPID	0.91364	0.07131	4.72643E-4	0.57846	0.87653
Drebin [1]	0.93521	0.06344	3.66257E-4	0.66972	0.91842
RoughDroid	0.95633	0.04577	2.48984E-6	0.79892	0.97833

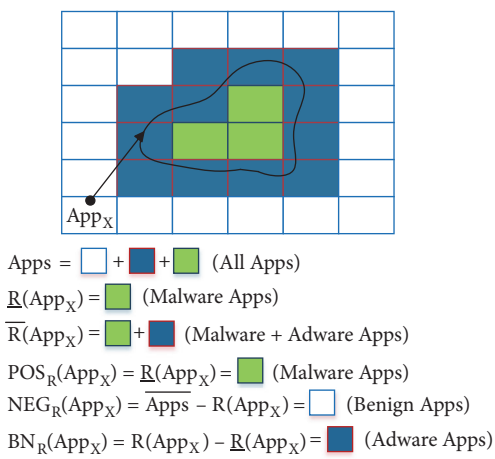


Figure 3: The approximations and regions of an Android applications' set App_X using Rough Set algorithm.

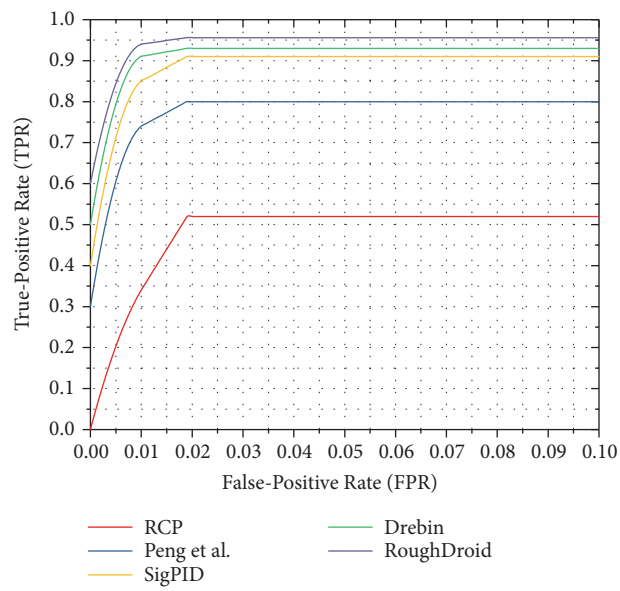


Figure 4: *RoughDroid* performance analysis ROC curve comparison with three approaches.

In addition, we have also considered 158 Android applications introducing three new malware families (*Grabos*, *TrojanDropper.Agent.BKY*, and *AsiaHitGroup*) that invade Google Play Store at 2017. It should be mentioned that the adware applications are considered in our dataset.

3.2. Performance Analysis. Our *RoughDroid* does not need initial training in advance, which is one of its basic advantages. *RoughDroid* analyzes each application in a broad floppy way and grab a great collection of features categorized in ten feature sets ($FS_1, FS_2, \ldots, FS_{10}$). It should be mentioned that the results are obtained from the average of 25 trials using the same environmental conditions. We introduce our analysis based on comparing *RoughDroid's* results with the results obtained from related approaches and ten popular antivirus scanners, finally employing *RoughDroid* to find the detection rate for the most popular malware families.

3.2.1. RoughDroid and Related Approaches. We initially contrast the efficiency of *RoughDroid* versus associated static methods for the discovery of Android malware. Specifically, we think about Drebin [1], RCP [16], Peng et al. [17], and SigPID [18]. The outcomes of these experiments are displayed in Figure 4 as ROC curve. *RoughDroid* outperforms the four previously mentioned approaches by detecting 95.6% of the malware applications at a false-positive rate (FPR) equal to 1%.

Also, according to the statistical measures introduced in Table 2, the *Asymptotic Probability* of the five schemes is much smaller than 0.05; thus we can conclude that all schemes are effective. In addition, the area under the curve of *RoughDroid* is 0.95633, which is closer to 1.0; hence, *RoughDroid* is the best scheme in successfully detecting the malware of an Android application. The excellent efficiency of *RoughDroid* arises from the various feature sets that are used to design the malicious activity of an application.

3.2.2. RoughDroid and Popular AV Scanners. *RoughDroid* reveals a much better efficiency compared to related approaches ([1, 16–18]). We likewise contrast it with ten picked antivirus scanners on the considered dataset. It should be mentioned that we consider $FPR = 1\%$, which we assume to be adequately low enough for practical scenarios.

Experimental results are displayed in Table 3. The best antivirus detects over 90% of malware applications. Our *RoughDroid* also provides best performance with detection rate of 95.6%.

3.2.3. Detecting Malware Families. When evaluating the detection efficiency of an approach, the equilibrium of malware family members in the dataset is very important. If the number of applications of a particular malware family members is little great compared to various other families, the detection result might mostly depend on these families.

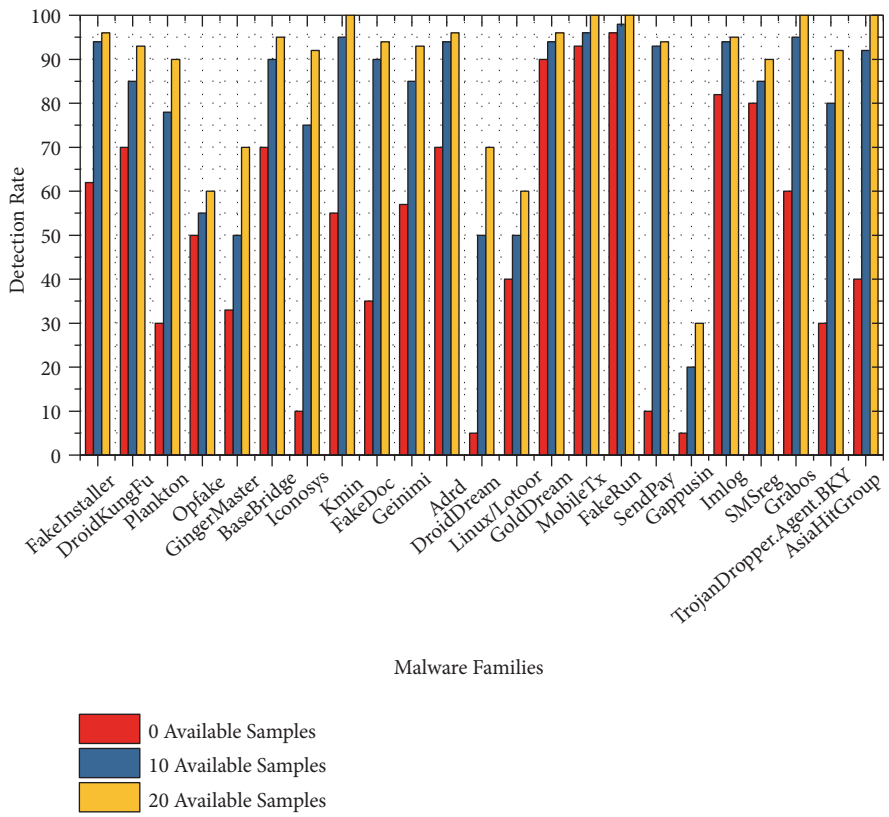


Figure 5: *RoughDroid* detection rate for 23 malware families based on three different number of applications for each family.

Table 3: Detection rates of *RoughDroid* in comparison with Drebin, SigPID, and ten anti-virus scanners.

	RoughDroid	Drebin [1]	SigPID [18]	Anti-V1	Anti-V2	Anti-V3	Anti-V4	Anti-V5	Anti-V6	Anti-V7	Anti-V8	Anti-V9	Anti-V10
Detection Rate	95.60%	93.90%	91.22%	96.41%	93.71%	84.66%	84.54%	78.38%	64.16%	48.50%	48.34%	9.84%	3.99%

An unreal solution to this problem is to make use of the same number of applications for each malware family.

We are laying more stress on 20 (*FakeInstaller, DroidKungFu, Plankton, Opfake, GingerMaster, BaseBridge, Iconosys, Kmin, FakeDoc, Geinimi, Adrd, DroidDream, Linux/Lotoor, GoldDream, MobileTx, FakeRun, SendPay, Gappusin, Imlog,* and *SMSreg*) top common malware families, plus the new three (*Grabos, TrojanDropper.Agent.BKY*, and *AsiaHitGroup*) malware families that invade Google Play Store at 2017.

We perform three more experiments, by restricting the variety of applications for a certain family in the test set. In the first experiment, we offer no applications of the family. In the second experiment, we place 10 arbitrarily picked applications of the family back right into the test set. Finally, in the third experiment, we use 20 arbitrarily picked applications of the family back right into the test set. The consequences of these three experiments are shown in Figure 5. *RoughDroid* can reliably detect all households with a typical precision of 95.6% at $FPR = 1\%$. The figure also shows that five (*Kmin, MobileTx, FakeRun, Grabos,* and *AsiaHitGroup*) families are perfectly detected.

4. Related Work and Discussion

To the best of our knowledge, Android malware detection and classification have a wide research area in the last decade. It has three basic categories, based on the detection technique, that is, static analysis, dynamic analysis, and machine learning analysis. Several methods have been proposed for statically analyzing an Android application, such as [8, 9, 18, 19]. Also, there are some contributions based on dynamic analysis, such as [5–7, 20, 21]. Regarding realizing the data placement considering both the energy consumption in private cloud and the cost for renting the public cloud services, the authors in [22] have proposed a cost- and energy-aware data placement method, for privacy-aware applications over big data in hybrid cloud.

Furthermore, the detection techniques [16, 17, 23, 24] are based on machine learning. The authors in [25] propose a new bio-key production algorithm called FVHS, which unites the benefits of the biometrics authentication and user-key authentication. Also, in [26] the authors suggest a new scheme named FREDP (File Remotely keyed Encryption and Data Protection). This strategy entails interaction between

one of the clouds that are personal and a terminal. The authors in [27] propose a new identity-based blind signature scheme based on number theorem research unit lattice.

The authors in [28, 29] are proposing a new access control for cloud infrastructure as a service. Also, a trust based access control model is proposed in [30]. In addition, cryptographic access control scheme is introduced in [31]. Also, the authors in [32] propose a new space metric optimization pushed deep-learning frame for age-invariant facial recognition. A complete review for Blockchain and intrusion detection is available in [33]. Reference [34] introduced JFCGuard for detecting juice filming charging attack and [35] enhanced network capacity. A privacy-preserving scheme based on location is introduced in [36].

Due to the sparsity of big rating data in E-commerce, both similar friends and similar product items may be absent from the user-product purchase network, which leads to a big challenge to the recommendation of appropriate product items to the target user. The authors in [37] propose a structural balance theory-based recommendation scheme. Also, protecting users' privacy is challenging when IBM releases its own data to Amazon. In addition, the recommendation efficiency and scalability are often low when the user-service quality data of Amazon and IBM are updated frequently. Thus, the authors in [38] have proposed a privacy-preserving and scalable service recommendation approach based on distributed locality-sensitive hashing.

Based on deep learning, [39] proposed a novel finger vein recognition algorithm. For social networks, [40] introduced a measure for social influence. An early detection scheme for IP traffic is introduced in [41]. A new instant encrypted transmission is proposed in [42]. Based on trusted routing, a sensitive analysis of attack-pattern is proposed in [43]. Finally, [44] presents a new scheme M-SSE that achieves both forward and backward security based on a multicloud technique.

5. Conclusion

This paper introduced *RoughDroid* that is a new broad floppy analysis malware detector on smart Android phones during the installation time by introducing robust feature extraction framework. *RoughDroid* performs a broad floppy analysis, gathering numerous features from an application's *Dex code* as well as *manifest* file. It is based on ten feature sets ($FS_1, FS_2, \ldots, FS_{10}$). It then uses the Rough Set algorithm to check the behavior of an Android application. The experimental results showed that *RoughDroid* is detecting 95.6% of the malware applications at a $FPR = 1\%$, which means that *RoughDroid* outperforms the well-known detection approaches (Drebin [1], RCP [16], Peng et al. [17], and SigPID [18]). Also, *RoughDroid* is compared with the ten most popular antivirus scanners and proved efficiency in practical scenarios. Finally, *RoughDroid* is able to perfectly detect five (*Kmin, MobileTx, FakeRun, Grabos,* and *AsiaHitGroup*) malware families.

Data Availability

The data used to support the findings of this study are available from the authors 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 Guangzhou Scholars Project for Universities of Guangzhou (No. 1201561613). Also, this work was supported by the Egyptian Ministry of Higher Education, the Arab Republic of Egypt.

References

[1] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck, "Drebin: effective and explainable detection of android malware in your pocket," in *Proceedings of the NDSS Symposium 2014*, February 2014.

[2] Android Developers Blog, *How We Fought Bad Apps and Malicious Developers in 2017*, January 2018.

[3] SC Media US, *Three M Android Malware Families Invade Google Play Store*, November 2017.

[4] Amazon's App Store Compromises Android Security, *It's dangerous to go alone outside Google's protective walled garden, but it's the price you pay for free software*, 2017.

[5] W. Enck, P. Gilbert, B.-G. Chun et al., "TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones," in *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI'10)*, pp. 393–407, February 2010.

[6] Y. Zhou, Z. Wang, W. Zhou, and X. Jiang, "Hey, You, get off of my market: Detecting malicious apps in official and alternative android markets," in *Proceedings of the Annual Symposium on Network and Distributed System Security (NDSS '12)*, 2012.

[7] L.-K. Yan and H. Yin, "Droidscope: Seamlessly reconstructing os and dalvik semantic views for dynamic android malware analysis," in *Proceedings of the USENIX Security Symposium*, 2012.

[8] A. P. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner, "Android permissions demystified," in *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS '11)*, pp. 627–638, ACM, New York, NY, USA, 2011.

[9] M. Grace, Y. Zhou, Q. Zhang, S. Zou, and X. Jiang, "RiskRanker: scalable and accurate zero-day android malware detection," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys '12)*, pp. 281–294, ACM, New York, NY, USA, June 2012.

[10] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*, MIT Press, 1989.

[11] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of the ACM*, vol. 13, no. 7, pp. 422–426, 1970.

[12] S. Rissino and G. Lambert-Torres, "Rough set theory –fundamental concepts, principals, data extraction, and applications," in *Data Mining and Knowledge Discovery in Real Life Applications*, pp. 35–59, INTECH, January 2009.

[13] Z. Pawlak, J. Grzymala-Busse, R. Slowinski, and W. Ziarko, "Rough sets," *Communications of the ACM*, vol. 38, no. 11, pp. 88–95, 1995.

[14] L. Polkowski, S. Tsumoto, and T. Y. Lin, Eds., *Rough Set Methods and Applications: New Developments in Knowledge Discovery*

in Information Systems, Physica-Verlag GmbH, Heidelberg, Germany, 2000.

[15] S. Hirano and S. Tsumoto, "Rough representation of a region of interest in medical images," *International Journal of Approximate Reasoning*, vol. 40, no. 1-2, pp. 23–34, 2005.

[16] B. P. Sarma, N. Li, C. Gates, R. Potharaju, C. Nita-Rotaru, and I. Molloy, "Android permissions: a perspective combining risks and benefits," in *Proceedings of the 17th ACM Symposium on Access Control Models and Technologies (SACMAT '12)*, pp. 13–22, ACM, New York, NY, USA, June 2012.

[17] H. Peng, C. Gates, B. Sarma et al., "Using probabilistic generative models for ranking risks of Android apps," in *Proceedings of the ACM Conference on Computer and Communications Security (CCS '12)*, pp. 241–252, ACM, New York, NY, USA, October 2012.

[18] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an, and H. Ye, "Significant permission identification for machine learning based android malware detection," *IEEE Transactions on Industrial Informatics*, vol. PP, no. 99, pp. 1-1, 2018.

[19] W. Enck, M. Ongtang, and P. McDaniel, "On lightweight mobile phone application certification," in *Proceedings of 16th ACM Conference on Computer and Communications Security (CCS '09)*, pp. 235–245, ACM, New York, NY, USA, November 2009.

[20] A. Reina, A. Fattori, and L. Cavallaro, "A system call-centric analysis and stimulation technique to automatically reconstruct android malware behaviors," in *Proceedings of the European Workshop on System Security (EUROSEC '13)*, April 2013.

[21] V. Rastogi, Y. Chen, and W. Enck, "AppsPlayground: automatic security analysis of smartphone applications," in *Proceedings of the 3rd ACM Conference on Data and Application Security and Privacy (CODASPY '13)*, pp. 209–220, ACM, New York, NY, USA, February 2013.

[22] X. Xu, X. Zhao, F. Ruan et al., "Data placement for privacy-aware applications over big data in hybrid clouds," *Security and Communication Networks*, vol. 2017, 2017.

[23] D. Barrera, H. G. Kayacik, P. C. Van Oorschot, and A. Somayaji, "A methodology for empirical analysis of permission-based security models and its application to android," in *Proceedings of the 17th ACM conference on Computer and communications security (CCS '10)*, pp. 73–84, ACM, New York, NY, USA, October 2010.

[24] D.-J. Wu, C.-H. Mao, T.-E. Wei, H.-M. Lee, and K.-P. Wu, "DroidMat: android malware detection through manifest and API calls tracing," in *Proceedings of the Seventh Asia Joint Conference on Information Security (Asia JCIS '12)*, pp. 62–69, Tokyo, Japan, August 2012.

[25] Z. Wu, L. Tian, P. Li, T. Wu, M. Jiang, and C. Wu, "Generating stable biometric keys for flexible cloud computing authentication using finger vein," *Information Sciences*, vol. 433-434, pp. 431–447, 2018.

[26] L. Yang, Z. Han, Z. Huang, and J. Ma, "A remotely keyed file encryption scheme under mobile cloud computing," *Journal of Network and Computer Applications*, vol. 106, pp. 90–99, 2018.

[27] H. Zhu, Y.-a. Tan, L. Zhu, X. Wang, Q. Zhang, and Y. Li, "An identity-based anti-quantum privacy-preserving blind authentication in wireless sensor networks," *Sensors*, vol. 18, no. 5, p. 1663, 2018.

[28] K. Riad, Z. Yan, H. Hu, and G.-J. Ahn, "AR-ABAC: A New Attribute Based Access Control Model Supporting Attribute-Rules for Cloud Computing," in *Proceedings of the 1st IEEE International Conference on Collaboration and Internet Computing, CIC 2015*, pp. 28–35, China, October 2015.

[29] K. Riad, "Blacklisting and forgiving coarse-grained access control for cloud computing," *International Journal of Security and Its Applications*, vol. 10, no. 11, pp. 187–200, 2016.

[30] K. Riad and Z. Yan, "Multi-factor synthesis decision-making for trust-based access control on cloud," *International Journal of Cooperative Information Systems*, vol. 26, no. 4, pp. 1–33, 2017.

[31] K. Riad, "Revocation basis and proofs access control for cloud storage multi-authority systems," in *Proceedings of the 3rd International Conference on Artificial Intelligence and Pattern Recognition, AIPR 2016*, pp. 118–127, Poland, September 2016.

[32] 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.

[33] 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.

[34] W. Meng, L. Jiang, Y. Wang, J. Li, J. Zhang, and Y. Xiang, "Jfcguard: Detecting juice filming charging attack via processor usage analysis on smartphones," *Computers & Security*, 2017.

[35] 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.

[36] T. Peng, Q. Liu, D. Meng, and G. Wang, "Collaborative trajectory privacy preserving scheme in location-based services," *Information Sciences*, vol. 387, pp. 165–179, 2017.

[37] L. Qi, X. Xu, X. Zhang et al., "Structural balance theory-based e-commerce recommendation over big rating data," *IEEE Transactions on Big Data*, p. 1, 2017.

[38] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[39] Y. Liu, J. Ling, Z. Liu, J. Shen, and C. Gao, "Finger vein secure biometric template generation based on deep learning," *Soft Computing*, vol. 22, no. 7, pp. 2257–2265, 2018.

[40] S. Peng, A. Yang, L. Cao, S. Yu, and D. Xie, "Social influence modeling using information theory in mobile social networks," *Information Sciences*, vol. 379, pp. 146–159, 2017.

[41] Z. Chen, L. Peng, C. Gao, B. Yang, Y. Chen, and J. Li, "Flexible neural trees based early stage identification for IP traffic," *Soft Computing*, vol. 21, no. 8, pp. 2035–2046, 2017.

[42] C. Wang, J. Shen, Q. Liu, Y. Ren, and T. Li, "A novel security scheme based on instant encrypted transmission for internet of things," *Security and Communication Networks*, vol. 2018, Article ID 3680851, 7 pages, 2018.

[43] R. H. Jhaveri, N. M. Patel, Y. Zhong, and A. K. Sangaiah, "Sensitivity analysis of an attack-pattern discovery based trusted routing scheme for mobile Ad-Hoc networks in industrial IoT," *IEEE Access*, vol. 6, pp. 20085–20103, 2018.

[44] C. Gao, S. Lv, Y. Wei, Z. Wang, Z. Liu, and X. Cheng, "M-SSE: An effective searchable symmetric encryption with enhanced security for mobile devices," *IEEE Access*, pp. 1-1, 2018.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 9081814, 12 pages
https://doi.org/10.1155/2018/9081814

Research Article

Secure Deduplication Based on Rabin Fingerprinting over Wireless Sensing Data in Cloud Computing

Yinghui Zhang,[1,2,3] Haonan Su,[1] Menglei Yang,[1] Dong Zheng,[1,3] Fang Ren,[1] and Qinglan Zhao[1]

[1] *National Engineering Laboratory for Wireless Security, Xi'an University of Posts and Telecommunications, Xi'an 710121, China*
[2] *State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China*
[3] *Westone Cryptologic Research Center, Beijing 100070, China*

Correspondence should be addressed to Yinghui Zhang; yhzhaang@163.com and Dong Zheng; zhengdong@xupt.edu.cn

Received 9 June 2018; Accepted 12 August 2018; Published 6 September 2018

Academic Editor: Lianyong Qi

The rapid advancements in the Internet of Things (IoT) and cloud computing technologies have significantly promoted the collection and sharing of various data. In order to reduce the communication cost and the storage overhead, it is necessary to exploit data deduplication mechanisms. However, existing data deduplication technologies still suffer security and efficiency drawbacks. In this paper, we propose two secure data deduplication schemes based on Rabin fingerprinting over wireless sensing data in cloud computing. The first scheme is based on deterministic tags and the other one adopts random tags. The proposed schemes realize data deduplication before the data is outsourced to the cloud storage server, and hence both the communication cost and the computation cost are reduced. In particular, variable-size block-level deduplication is enabled based on the technique of Rabin fingerprinting which generates data blocks based on the content of the data. Before outsourcing data to the cloud, users encrypt the data based on convergent encryption technologies, which protects the data from being accessed by unauthorized users. Our security analysis shows that the proposed schemes are secure against offline brute-force dictionary attacks. In addition, the random tag makes the second scheme more reliable. Extensive experimental results indicate that the proposed data deduplication schemes are efficient in terms of the deduplication rate, the system operation time, and the tag generation time.

1. Introduction

The wireless sensor network (WSN) is an ad hoc network composed of a large number of sensors, and the sensors communicate with each other over a wireless channel in a multihop manner [1–5]. Sensors are usually a low-cost, simple device with limited computing power and working batteries, which have the ability to collect, process, and transfer data. With the rapid development of Internet of Things (IoT) and cloud computing technologies, WSN has found many promising applications. As an extension to the cloud computing paradigm, fog computing makes it possible to execute the IoT applications in the network of edge. Xu et al. [6] proposed a dynamic resource allocation method for load balancing in fog environment. Cloud computing [7, 8] supports distributed data storage and parallel processing and its data processing framework handles huge amounts of data in a local computer rather than requiring to transmit these data remotely [9–11]. We know that cloud storage technology is the most common and most popular cloud computing service today. The extensive application of cloud storage motivates enterprises and organizations to outsource data storage to third-party cloud providers [12–16]. Zhang et al. [17] proposed a fine-grained access control system suitable for resource-constrained users in cloud computing. It is reported that the average size of backup data for a medium size enterprise is 285 TB and faces an annual growth rate of about 24-27%. According to the analysis report of IDC, personal user data has reached terabytes in 2006. From 2006 to 2010, global data volume continues to grow at a rate of 57% annually. In 2011, the global data volume has entered the era of ZB, and the total amount of data used globally exceeds 1.8

ZB. It is expected that the global data volume will reach 40 ZB by 2020 [18].

Data deduplication has been widely accepted as an effective technique to reduce workload and overhead of the cloud storage system [19–23]. Today's commercial cloud storage services, such as Dropbox, Google Drive, Bitcasa, Mozy, and Memopal, have been applied deduplication to save maintenance cost. However, the extensive application of data deduplication makes its security problems increasingly prominent [24, 25]. Compared with traditional information security, cloud storage security [26–28] mainly has two characteristics: users do not enjoy physical control over the data they upload to the cloud storage system and the same kind of physical resources is shared by multiple users. The confidentiality and integrity of data will be threatened. It is noted that cloud storage security has drawn many attentions [29, 30]. Xu et al. [31] proposed a cost and energy aware data placement method for privacy-aware applications over big data in hybrid cloud. Harnik et al. [32] pointed out that there were security vulnerabilities in the deduplication technology used by the provider. Douceur et al. [33] introduced convergent encryption (CE) that uses the hash value of the data itself as a secret key to solve the problem of contradiction between deduplication and confidentiality. Bellare et al. [34] defined a cryptographic primitive called message-locked encryption. Li et al. [35] implemented Dekey using the Ramp secret sharing scheme to manage the CE keys. Literature [21] pointed out that, in the data deduplication, simply using the hash value of the file represents the entire file, making the data deduplication process vulnerable to hacking, and the hash value is not confidential, and the attacker can obtain the entire file content by obtaining the hash value. Abadi et al. [36] proposed two schemes, including a completely random scheme and a deterministic scheme, which support the randomization of tags to ensure the security of the data deduplication system. In the schemes, CE directly uses the data fingerprint as the key derivation function and hence only achieves security for unpredictable data. In fact, offline brute-force dictionary attacks can be easily launched because of the determination of CE keys [37]. Moreover, current deduplication schemes [35, 37] directly deduplicate the encrypted data, which increases the computational overhead. In the future, it is possible to realize decentralized data deduplication schemes via blockchain technologies, which have been used to realize decentralized outsourcing computation [38, 39] and searchable encryption with two-side verifiability [40] in cloud computing.

Deduplication can be defined based on different granularities [41]: file-level deduplication and block-level deduplication (fine-grained fixed-size or variable-size data block). File-level deduplication is the easiest but inefficient method. Fixed-size block-level deduplication refers to blocking the file into fine-grained fixed-size (such as 4MB, 512KB) data blocks and then deleting the duplicate blocks [42]. However, it is difficult for fixed-size block-level deduplication to deal with the situation of insertion of data in the file. Abadi et al. [36] propose a completely random scheme that avoids deterministic messages to generate tags directly and better guarantees the security of the data deduplication process. On the basis of [36], Jiang et al. [43] added static data deduplication decision trees and dynamic data deduplication decision trees and optimized duplicate detection operations. However, most of previous schemes realize data deduplication after the data is encrypted by users, and hence the computation and communication efficiencies remain to be improved. In [44], the authors proposed a data deduplication scheme based on Rabin fingerprinting, which is a preliminary version of the work given in Section 4.2 of this paper. In this paper, we significantly revise the preliminary scheme [44] and add more technical details as compared to the preliminary abstract [44]. First, we add Section 3 to describe a system architecture of secure deduplication based on Rabin fingerprinting over wireless sensing data in cloud computing. Second, we improve the basic construction to support randomized tags and provide detailed procedures of data deduplication using randomized tags in Section 4.3. Third, we present security analysis of both schemes in Section 5 and do extensive experiments to evaluate the proposed deduplication schemes in Section 6.

Our Contribution. The contributions of this paper can be summarized as follows. In order to tackle the security and efficiency drawbacks in the existing data deduplication technologies, we propose two secure data deduplication schemes based on Rabin fingerprinting over wireless sensing data in cloud computing. The first scheme is based on deterministic tags and the other one adopts random tags. Note that the randomized tag achieves more reliable security guarantees than the deterministic tag. In order to reduce the communication cost and the computation cost, data deduplication in the proposed schemes is realized before the data is outsourced to the cloud storage server. For the sake of practicability, we realize variable-size block-level deduplication of the data, which is enabled based on the technique of Rabin fingerprinting. In order to protect the outsourcing data from being accessed by unauthorized users, the data is encrypted by users based on convergent encryption technologies before outsourcing data to the cloud. Our security analysis shows that the proposed schemes are secure against both external attacks and internal attacks. Extensive experimental results indicate that the proposed data deduplication schemes are efficient in terms of the deduplication rate, the system operation time, and the tag generation time.

Organization. The rest of this paper is organized as follows. Notations and cryptographic backgrounds are reviewed in Section 2. The system model, the threat model and security requirements of a secure deduplication scheme are described in Section 3. We present the proposed two data deduplication schemes in Section 4. Section 5 gives the security analysis of the proposed schemes and Section 6 shows the performance evaluation. Finally, our concluding remarks are made in Section 7.

2. Preliminaries

In this section, we first explain notations used throughout this paper and then simply review some cryptographic

Table 1: Notation description.

Notation	Meaning
q	A prime.
$s \in_R S$	s is randomly chosen from the set S.
g	A generator of a cyclic group of order q.
$(K_{\text{pub}}, K_{\text{pri}})$	The public and secret key pair of a user.
B_i	A data block.
$f(B_i)$	The Rabin fingerprinting of B_i.
K_i	The convergent key corresponding to B_i.
C_i	The ciphertext corresponding to B_i.
τ_i	The random tag corresponding to B_i.
$\mathsf{B}(X)$	The bitwise exclusive of X.

backgrounds involved in the proposed data deduplication schemes.

2.1. Notations. In Table 1, we list notations mainly used in the description of the proposed data deduplication schemes.

2.2. Rabin Fingerprinting. The technique of Rabin fingerprinting is widely used for quick comparison and recognition of duplicate data. It is based on arithmetic modulo an irreducible polynomial over $\mathbb{Z}_2$ [45]. Let $S = [a_1, a_2, \ldots, a_n]$ be a bit string. We define a polynomial $S(t)$ of degree $n-1$ over $\mathbb{Z}_2$ as

$$S(t) = a_1 t^{n-1} + a_2 t^{n-2} + \cdots + a_{n-1} t + a_n. \quad (1)$$

Let $p(t) = b_1 t^k + b_2 t^{k-1} + \cdots + b_k t + b_{k+1}$ be an irreducible polynomial of degree k over $\mathbb{Z}_2$. Given a fixed $p(t)$, the Rabin fingerprinting of $S(t)$ is defined as the polynomial $r(t) = S(t) \bmod p(t)$. The computation of Rabin fingerprinting is illustrated in Figure 1, where $[X_1, X_2, \ldots, X_\omega, X_{\omega+1}, X_{\omega+2}, \ldots]$ is a continuous string and each character X_i is a tuple of 8 bits.

Note that a sliding window of width ω is used. Assume the starting point is X_i which is represented by a polynomial $X_i(t)$; thus the Rabin fingerprinting value of the string $[X_i, X_{i+1}, \ldots, X_{i+\omega-1}]$ in the window is

$$r_i(t) = \left(\sum_{j=1}^{\omega} X_{i+j-1}(t)\, t^{8\omega-j} \right) \bmod p(t). \quad (2)$$

When the window slides forward 8 bits, X_{i+1} becomes the starting point and then the Rabin fingerprinting value of the string $[X_{i+1}, X_{i+2}, \ldots, X_{i+\omega}]$ in the window is

$$r_{i+1}(t) = \left(\sum_{j=1}^{\omega} X_{i+j}(t)\, t^{8\omega-j} \right) \bmod p(t). \quad (3)$$

In fact, the Rabin fingerprinting algorithm computes a rolling checksum of the data [46]. The window of the data is configurable, but it is typically a few dozen bytes long. The Rabin module will read through a file and let the window slide over the data. When a byte is read, the fingerprint is

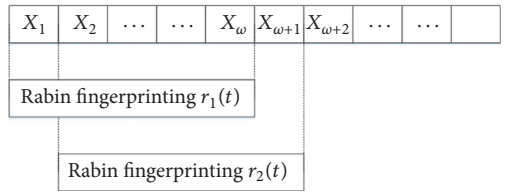


Figure 1: The computation of Rabin fingerprinting.

recalculated. If the fingerprint is a special value, the Rabin module considers the corresponding window position to be a boundary. The data preceding this window position is taken to be a "block" of the file. For $1 \le i \le n$, let B_i be a "block," and the fingerprint of the data block B_i is defined as.

2.3. Proof of Ownership. A proof of ownership (PoW) protocol [47] enables a client to prove to the server that they own a given file. The server can derive a small metadata $T(M)$ from the data M. To prove the ownership of the data M, the user needs to send T' and run a proof algorithm with the sever. Its ownership is accepted if and only if $T' = T(M)$ and the proof is correct [48].

2.4. Convergent Encryption. The notion of convergent encryption was proposed by Douceur et al. [33]. In order to ensure the confidentiality of outsourcing data in the data deduplication process, users first encrypt data and then upload ciphertexts. In practice, if traditional encryption mechanisms are adopted, different users have diverse encryption keys, which leads to that the same file will be encrypted to different ciphertexts by diverse users. This property poses a serious challenge to data deduplication form the point of efficiency. In convergent encryption, the key is derived from the outsourcing data, and hence the same data corresponds to the same ciphertext even if users are different. Therefore, CE makes it possible to realize secure data deduplication in ciphertexts. Figure 2 illustrates the process of a convergent encryption. A convergent encryption scheme consists of the following algorithms:

(i) $\mathsf{KenGen}_{\text{CE}}(M) \longrightarrow K$. The key generation algorithm generates a convergent key K based on data M. For a secure use of convergent encryption, the convergent key should be unpredictable, which can be realized by introducing randomness based on the message authentication code (MAC). MAC is also known as the keyed hash function. It is a value obtained based on a secret key and a message digest, which is usually used to data source authentication and integrity checking. A MAC is defined as below.

(a) $\mathsf{Hash}(M) \longrightarrow H_b$ is a hash algorithm, such as SHA-1 and SHA-256, which takes as input the message M and outputs the hash value.

(b) $\mathsf{keyHmac}(secret, H_b) \longrightarrow K$ is a message authentication code that takes as inputs the hash value H_b and a random parameter *secret* and outputs a randomized convergent key K.

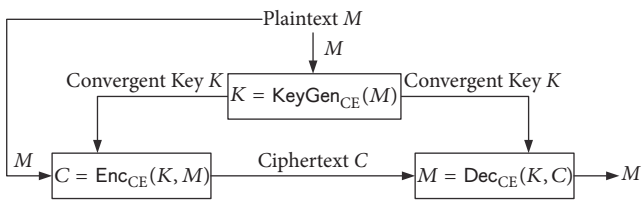


Figure 2: The convergent encryption process.

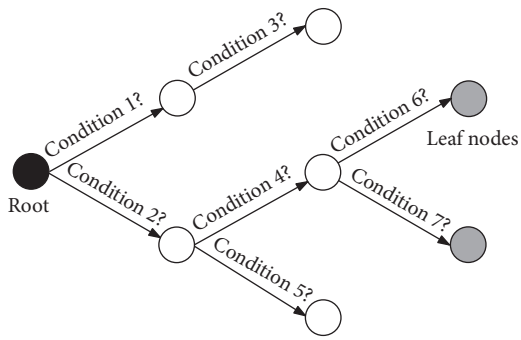


Figure 3: An example of decision tree.

(ii) $\mathsf{Enc}_{\mathrm{CE}}(K, M) \longrightarrow C$. It is a symmetric encryption algorithm that takes the convergent key K and the data M as inputs and outputs a ciphertext C.

(iii) $\mathsf{Dec}_{\mathrm{CE}}(K, C) \longrightarrow M$. It is the corresponding decryption algorithm that takes the convergent key K and the ciphertext C as inputs and outputs the original data M.

(iv) $\mathsf{TagGen}_{\mathrm{CE}}(M) \longrightarrow T_M$. The tag generation algorithm maps the original data M to a tag T_M. Essentially, the Rabin fingerprinting of data is used as the tag in the deterministic tag based scheme and is used to generate tags for the random tag based scheme.

2.5. Decision Trees. As a predictive model, a decision tree is a tree-like structure, in which each internal node denotes a test on an attribute, each branch represents the test output, and each leaf node means a category. For example, as shown in Figure 3, a decision tree consists of nodes and branches. Typically, a decision tree begins with the root node and branches connect the nodes. A branch that originates from a decision node is called a decision branch. Note that different conditions are associated with different branches. A leaf node acts as a termination node, which indicates the final outcome of the branch.

3. Models and Security Goals

In this section, we first introduce the system model and then describe the threat model and security goals.

3.1. System Model. The system model is illustrated in Figure 4, in which three entities are involved, including a management server (MS), users, and a cloud storage server (CSS). In the model, users outsource their data to CSS and access the data later with the help of MS, while keeping the ability of data deduplication. The details are described as follows.

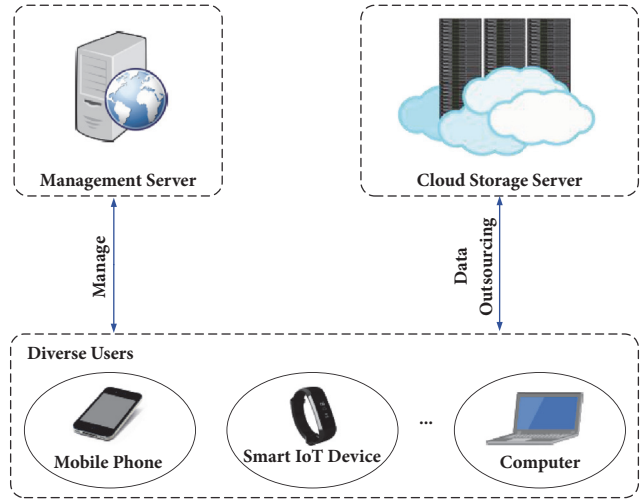


Figure 4: The system model.

(i) MS. It is trusted by users and manages secret keys and users' information. MS introduces a random secret parameter to generate randomized convergent keys for users.

(ii) Users. Users can compute the block fingerprints before data deduplication. They encrypt data and then upload ciphertexts to CSS. For recovering the data, they decrypt the corresponding ciphertext from CSS.

(iii) CSS. It is honest but curious and provides data storage service to users. It stores and manages user's unique data copies in the form of ciphertexts. In the subsequent random tag based deduplication scheme, CSS checks duplicate data based on a decision tree.

3.2. Threat Model and Security Goals. We consider both external attackers and internal attackers for the security of outsourcing data storage with data deduplication. For one thing, in the public channels, the external attackers are able to achieve partial of information on the data. An external attacker can access CSS by disguising as a legitimate user. For another, the internal attackers are honest but curious. They will follow the procedures of the proposed scheme and try to get confidential information as much as possible. The goal of the internal attackers is to obtain the contents of the data from CSS and obtain the randomized convergent keys from MS.

Considering the above threat model, we specify the following security goals. First, we need to ensure that the semantic security of encrypted data blocks. This requirement has been formalized in [49]. Therefore, the adversary does not have the ownership of the data because there is no convergent key to encrypt. Second, the convergent keys should be kept secure. The goal of the attackers is to get the other users' keys and the data block ciphertexts. We aim to guarantee the security of the keys' transmission and storage. Neither

external attackers nor internal attackers can obtain other convergent keys.

4. Data Deduplication Schemes Based on Rabin Fingerprinting

In this section, we propose two data deduplication schemes based on Rabin fingerprinting, including a deterministic tag based scheme and a random tag based scheme. In each scheme, three phases, system setup, file uploading, and file downloading, are performed for data outsourcing storage with deduplication. The proposed deduplication schemes perform block-level data deduplication before users' data encryption, in which the file blocks are generated based on Rabin fingerprinting.

4.1. Overview of Our Schemes. In the first scheme, the outsourcing data is first divided into many data blocks based on the Rabin fingerprinting technique. For each data block, a deterministic tag is generated based a hash function. With the tag, the cloud storage server can check whether the corresponding data block has already existed. If it exists, the user proves to the cloud server that it indeed has the ownership of the data block. Otherwise, the user encrypts the data block and uploads the generated ciphertext to the cloud server, in which the ciphertext is based on a convergent encryption and the convergent key is generated by the management server. The security of data in the deduplication process is ensured based on encryption techniques, and the convergent keys are also effectively managed. However, deterministic tags fail to meet the standard confidentiality requirement, such as semantic security. To be specific, if the plaintext can be listed, the attacker can learn the content of the plaintext by computing the tags and comparing the ciphertexts. If the tag is unpredictable, the above security drawback can be avoided. In the second scheme, the tag is randomly generated by the management server. The new scheme can support randomized tags and also allows decision tree based data duplicate detection. The decision tree supports the deletion and updating without needing expensive bilinear pairing operations. The randomized tags sacrifice efficiency to some extent but provide more reliable protection for data confidentiality in data deduplication systems.

4.2. Data Deduplication with Deterministic Tags

4.2.1. System Setup. In the system setup phase, necessary parameters are generated based on the following procedures:

(S1) Given a security parameter 1^λ, MS specifies a convergent encryption scheme ($\mathsf{KeyGen}_{\mathrm{CE}}$, $\mathsf{Enc}_{\mathrm{CE}}$, $\mathsf{Dec}_{\mathrm{CE}}$, and $\mathsf{TagGen}_{\mathrm{CE}}$), an asymmetric encryption scheme ($\mathsf{KeyGen}_{\mathrm{AE}}$, $\mathsf{Enc}_{\mathrm{AE}}$, and $\mathsf{Dec}_{\mathrm{AE}}$), and a PoW algorithm. MS runs $\mathsf{KeyGen}_{\mathrm{AE}}$ to generate an asymmetric public and secret key pair (K_{pub}, K_{pri}) for each user. Note that $\mathsf{KeyGen}_{\mathrm{CE}}$ is realized based on $\mathsf{keyHmac}$ and $\mathsf{TagGen}_{\mathrm{CE}}$ is computed based on the Rabin fingerprinting.

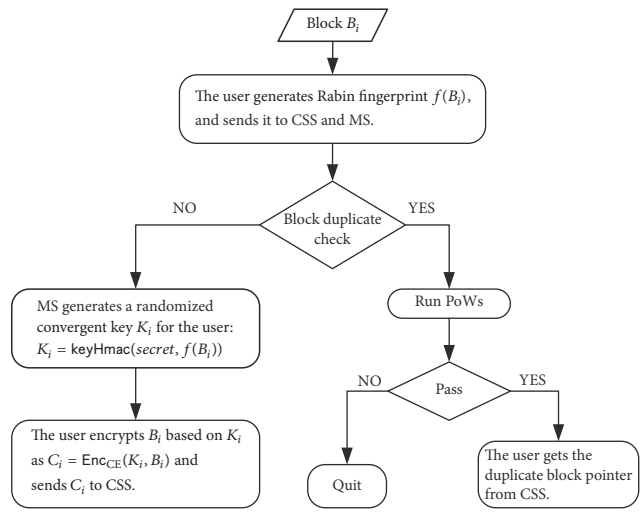


Figure 5: The uploading phase of deterministic tag based data deduplication.

(S2) The CSS initializes two types of storage systems: a fast storage system for efficient detection of duplicate data tags and a file storage system for storing encrypted outsourcing data.

(S3) MS initializes its local storage system for storing users' metadata and randomized convergent keys.

4.2.2. File Uploading. The uploading phase is shown in Figure 5. Suppose that a user uploads a file F and then performs the block-level deduplication below:

(S1) The user sends a file-backup request to MS, including its authentication information. Then, MS performs an identity authentication. If passed, the following steps are performed.

(S2) Based on the Rabin fingerprinting technique, the user divides F into a set of blocks denoted by $\{B_i\}_{1\le i\le n}$. The user computes each block fingerprint $f(B_i)$ and sends $f(B_i)$ as tags to CSS for duplicate checking.

(S3) In addition, the fingerprints $\{f(B_i)\}$ are sent to MS for generating convergent keys later.

(S4) Once the data block fingerprints $\{f(B_i)\}$ are received, CSS computes the data block signal vector σ_B as follows:

(i) For each i, if an existing block fingerprint matches $f(B_i)$, CSS sets $\sigma_B[i] = 1$ to indicate "block duplicate."

(ii) Otherwise, CSS sets $\sigma_B[i] = 0$ to indicate "no block duplicate." CSS stores $f(B_i)$ into the fast storage system.

After the data deduplication is fulfilled, CSS returns the signal vector σ_B to the user.

(S5) After receiving σ_B, the user checks if $\sigma_B[i] = 1$. If it is, the user runs a PoW algorithm to prove to CSS

that it owns the data block B_i. If CSS accepts the proof, it directly returns the corresponding pointer of B_i to the user. At the same time, the user stores the block pointer of B_i which is not needed to upload. In the other cases, the protocol is terminated and the involved entities quit the protocol.

(S6) Otherwise, the user sends σ_B to MS. Upon receiving σ_B, MS checks if $\sigma_B[i] = 0$. If it is, MS generates the convergent key $K_i = \mathsf{keyHmac}(secret, f(B_i))$, where *secret* is a randomly chosen parameter. MS sends the randomized convergent key K_i corresponding to the nonduplicate block to the user. The user computes a ciphertext $C_i = \mathsf{Enc}_{\mathrm{CE}}(K_i, B_i)$ and uploads C_i to CSS.

4.2.3. File Downloading. Suppose that a user intends to download a file F. The user first sends a downloading request to MS, including its authentication information. If the authentication is successfully verified, the following procedures are performed:

(S1) MS encrypts the randomized convergent key K_i by computing $C_{k_i} = \mathsf{Enc}_{\mathrm{AE}}(K_{\mathrm{pub}}, K_i)$, which is then sent to the user.

(S2) Upon receiving the ciphertext C_{k_i}, the user decrypts it based on its secret key K_{pri} to get the randomized convergent key K_i, that is, $K_i = \mathsf{Dec}_{\mathrm{AE}}(K_{\mathrm{pri}}, C_{k_i})$. Subsequently, the user obtains the encrypted data block $\{C_i\}$ from CSS.

(S3) The user decrypts the corresponding ciphertext C_i by computing $B_i = \mathsf{Dec}_{\mathrm{CE}}(K_i, C_i)$, based on K_i, and then restores the file F.

4.3. Data Deduplication with Randomized Tags

4.3.1. System Setup. The details are the same to those in the deterministic tag based scheme. Besides, MS specifies a cyclic group of prime order q with generator g.

4.3.2. File Uploading. Suppose that a user intends to outsource the file F. The tag corresponding to the data block B_i is $\tau_i = (g^{r_i}, g^{r_i f(B_i)}, s_i)$, where r_i is randomly chosen from $\mathbb{Z}_q^*$ and $f(B_i)$ is the data fingerprint. The value of s_i can be 0 or 1. If $s_i = 1$, it means the corresponding data block of the tag in the decision tree has not been deleted. When the data block is deleted, s_i is set to be 0, which means there is no corresponding data block in CSS. The data uploading process is illustrated in Figure 6.

(S1) The user sends a file-backup request to MS, including its authentication information. Then, MS performs an identity authentication. If passed, the following steps are performed:

(S2) Based on the Rabin fingerprinting technique, the user divides F into a set of blocks denoted by $\{B_i\}_{1\le i\le n}$. The user computes each block fingerprint $f(B_i)$ and sends $f(B_i)$ to MS.

(S3) Upon receiving the data backup request, CSS first iterates through the tag nodes in the order of the decision tree. If $s_i = 0$, it traverses the next node's tag until $s_i = 1$ or a leaf node. If $s_i = 1$, CSS returns the tag $\tau_i = (g^{r_i}, g^{r_i f(B_i)}, s_i)$ to the user. Note that the root node tag of the decision tree is $\tau_0 = (g^{r_0}, g^{r_0 f(B_0)}, s_0)$ and s_i has a default value 1.

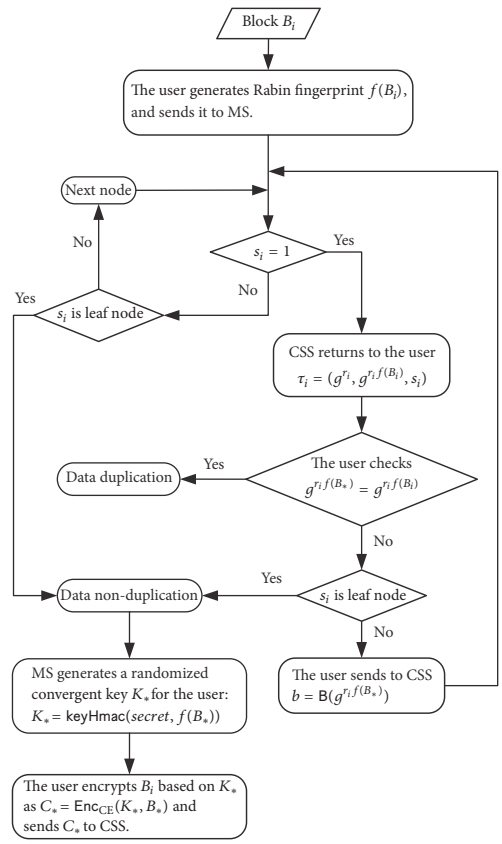


Figure 6: The uploading phase of random tag based data deduplication.

(S4) Once the user receives the tag sent by CSS, the user calculates $g^{r_i f(B_*)}$ and verifies that $g^{r_i f(B_*)}$ is equal to $g^{r_i f(B_i)}$.

(i) If $g^{r_i f(B_*)} = g^{r_i f(B_i)}$, the user sends "data duplication" to CSS and skips to the step (S6).

(ii) Otherwise, the user calculates $b = \mathsf{B}(g^{r_i f(B_*)})$ and sends it to CSS.

(S5) The server moves the pointer to the next node in the decision tree based on the result of $b = \mathsf{B}(g^{r_i f(B_*)})$.

(i) If $b = 0$, CSS will move the pointer to the left node.

(ii) If $b = 1$, the pointer will be moved to the right node and the above step (S3) is performed again.

If the decision tree pointer has not found a duplicate node after moving to the leaf node, CSS will send "data non-duplication" instruction to the user and skips to the following step (S7).

(S6) Once the user receives a "data duplication" instruction for a block B_*, it runs a PoW protocol with CSS to prove its ownership of the block. If passed, then CSS will return to the user a pointer to the duplicate data block B_*. The user then stores the pointer and the data block B_* does not need to be uploaded.

(S7) Once the user receives the "data non-duplication" associated with the data block B_*, MS generates the convergent key $K_* = \mathsf{keyHmac}(secret, f(B_*))$, where *secret* is a randomly chosen parameter, and sends the randomized convergent key K_* to the user. The user will run the encryption algorithm $C_* = \mathsf{Enc}_{\mathrm{CE}}(K_*, B_*)$ to compute the ciphertext C_* and upload it to CSS. At the same time, the user chooses $r_* \in_R \mathbb{Z}_q^*$, generates a corresponding tag $\tau_* = (g^{r_*}, g^{r_* f(B_*)}, s_*)$, and sends the tag to CSS.

(S8) Upon receiving the tag $(g^{r_*}, g^{r_* f(B_*)}, s_*)$ of the block B_* from the user, CSS computes $b = \mathsf{B}(g^{r_i f(B_*)})$. If $b = 0$, the tag $(g^{r_*}, g^{r_* f(B_*)}, s_*)$ will cover the left node with $s_* = 0$, or be placed on the left leaf node. If $b = 1$, the tag $(g^{r_*}, g^{r_* f(B_*)}, s_*)$ will cover the right node with $s_* = 0$ or be placed on the right leaf node.

4.3.3. File Downloading. Suppose that a user intends to download a file F. The user first sends a downloading request to MS, including its authentication information. If the authentication is successfully verified, the following procedures are performed:

(S1) MS encrypts the randomized convergent key K_i by computing $C_{k_i} = \mathsf{Enc}_{\mathrm{AE}}(K_{\mathrm{pub}}, K_i)$, which is then sent to the user.

(S2) Upon receiving the ciphertext C_{k_i}, the user decrypts it based on its secret key K_{pri} to get the randomized convergent key K_i, that is, $K_i = \mathsf{Dec}_{\mathrm{AE}}(K_{\mathrm{pri}}, C_{k_i})$. Subsequently, the user obtains the encrypted data block $\{C_i\}$ from CSS.

(S3) The user decrypts the corresponding ciphertext C_i by computing $B_i = \mathsf{Dec}_{\mathrm{CE}}(K_i, C_i)$, based on K_i, and then restores the file F.

5. Security Analysis of the Proposed Schemes

The proposed two data deduplication schemes differ in the tags. The first scheme adopts deterministic tags and the second scheme uses random tags. The involvement of random parameters in the tag generation makes the second scheme more secure. In the following, we only show that the deterministic tag based scheme is secure against both external attacks and internal attacks.

5.1. Security against External Attacks. In data deduplication systems, external attackers must be prevented from accessing data. For instance, the transmitted data between the user and CSS may be obtained by an external attacker. After selecting the range of a dictionary, the attacker can obtain data corresponding to metadata by the way of brute-force dictionary attack. In particular, an external attacker may maliciously modify and destroy users' transmitted data in order to compromise both the integrity and the availability of the data. In the proposed deduplication scheme, random information is added to the convergent key by MS, which randomizes the convergent key and alleviate the key compromise risk. The randomization of the convergent key makes offline brute-force attack very difficult. Because each user first encrypts outsourcing data and then transmits data ciphertexts in the system, it is impossible for external attackers to get the original data without needing the relevant key.

5.2. Security against Internal Attacks. In order to issue cross-border operations, attackers often try to hide their own identities. For example, an attacker may disguise as other legitimate users to violate the privacy of other users. To prevent the internal attackers, the secure deduplication system realizes the identity authentication when a user initially communicate with MS which stores and manages the convergent keys to prevent unauthorized information read. At the side of CSS, if a user aims to access a file, a PoW protocol is required to be performed between the user and CSS. The user can prove to CSS its ownership of the file. The proposed deduplication scheme can effectively prevent attackers from accessing any files and keys beyond their ownership. In the random tag based scheme, besides the security of the first scheme, it also avoids the use of deterministic tags during duplicate data detection. Accordingly, even if an attacker obtains a tag, the randomness of the tag makes it possible to obtain the corresponding convergent key, which further improves the system security.

6. Performance Evaluation

In this section, we evaluate the performance of the proposed Rabin fingerprinting based data deduplication systems. We also compare the trivial deduplication rate of the fixed-size block scheme and our Rabin fingerprinting based schemes.

6.1. Simulation Environment. The hardware used in the simulation is a 64-bit Lenovo 80ER laptop with Windows 7 Home Basic operating system, and its CPU is Intel(R) Core(TM) i5-5300U CPU @2.30GHz. The simulation code is written with Java language by using the MyEclipse development platform. In our experiments, the data samples are 5400 different journal articles from the China national knowledge infrastructure, and they are about 15.9 GB in size.

6.2. Experimental Results and Analysis

6.2.1. The Optimized Sliding Step Size. In this section, we aim to find the optimized sliding step size of the data deduplication scheme based on the Rabin fingerprinting. The optimized sliding step size enables a better performance of the data deduplication system. Specifically, given a fixed-size data set, we set the upper bound of the data block size as 8 KB and the sliding window size of the Rabin fingerprinting as 64 KB. Then, when the window sliding step size varies from 1 B to 20 B, we test the running time and the

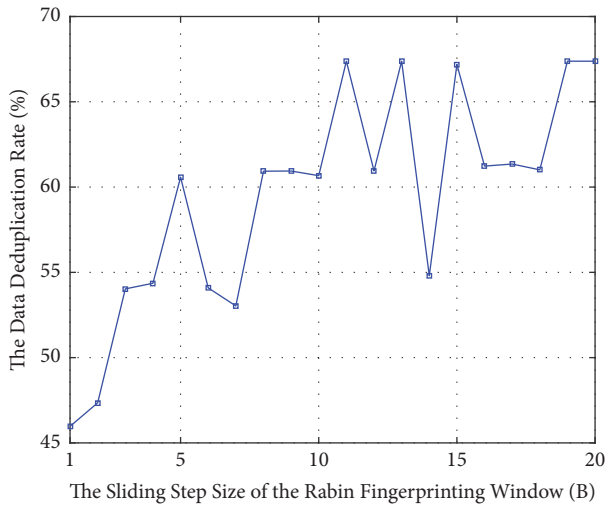


Figure 7: The variation of the deduplication rate with the sliding step size.

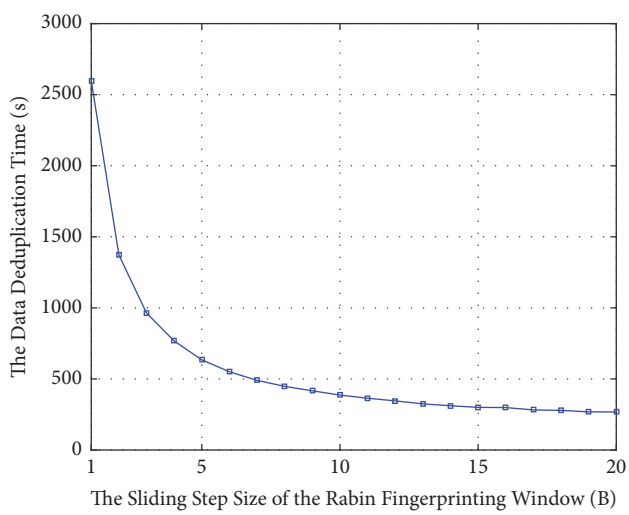


Figure 8: The variation of the data deduplication time with the sliding step size.

deduplication rate of the Rabin fingerprinting, respectively. Note that the data deduplication rate is defined as the ratio of the remaining nonduplicate data size after data deduplication to the total data size. The smaller the ratio, the better the data deduplication effect.

Figure 7 illustrates the variation of the data deduplication rate with the sliding step size of the window. Figure 8 shows the variation of the data deduplication time with the sliding step size of the window. We can see from Figure 7 that the deduplication rate has the optimal value when the Rabin fingerprinting has a window sliding step of 1 B and more than 50% of the duplicate data is removed. In this case, however, the deduplication time is the longest as shown in Figure 8. As the sliding step size increases, the data deduplication rate fluctuates between fixed values and it tends to be steady between several given sliding step size. When the window sliding step size is 1 B, the time of the deduplication

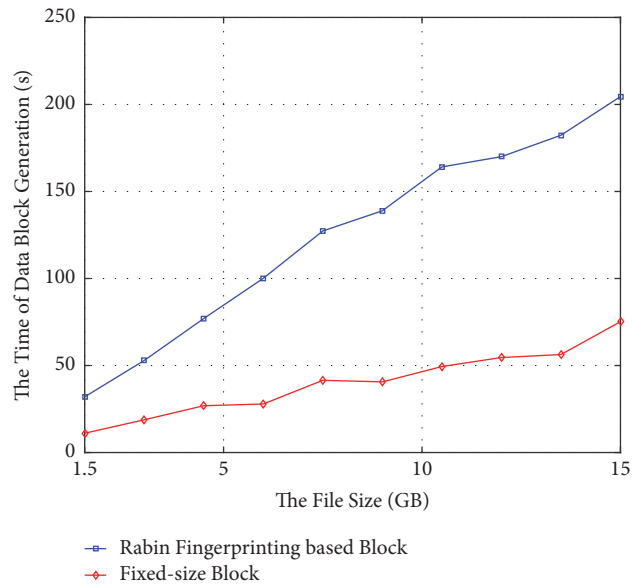


Figure 9: The time of data block generation with the file size.

based on Rabin fingerprinting is the longest. The longer the sliding window moves, the less time it takes for data to be deduplicated. Generally, in order to ensure the effect of data deduplication based on Rabin fingerprinting and reduce the system operation time, we exploit a sliding window of 64 KB and a sliding step size of 18 B in the following experiments.

6.2.2. The Performance Comparison of Rabin Fingerprinting Based Scheme and Fixed-Size Block Scheme. Figure 9 shows that the time for the data block generation varies with the file size. At the same time, we compare the block generation time of the fixed-size block deduplication scheme and the Rabin fingerprinting based scheme. Given the test file of the same size, the time required for the fixed-size block scheme is smaller than that based on the Rabin fingerprinting. Nevertheless, we will show that the total system operation time of the Rabin fingerprinting based scheme is optimal, later. Figure 10 compares the time of data deduplication based on the Rabin fingerprinting and data deduplication based on fixed-size blocks. The deduplication time does not include the block generation time, and the performance of these two schemes is compared from the perspective of data deduplication. It can be seen from Figure 10 that the data deduplication time of both schemes increases with the file size. If the test file is given, the deduplication efficiency of the Rabin fingerprinting based scheme is obviously better than that of the fixed-size block scheme. The comparison reflects the advantage of the Rabin fingerprinting in data deduplication.

In the subsequent simulation, the fixed-size block algorithm is first used to divide the test files into fixed-size data blocks of sizes 4, 8, 16, 32, 64, 128, 256, 512, 1024, and 2048 KB, respectively. Then, the Rabin fingerprint algorithm is used to divide the same files into variable-size data blocks with the upper bound limit of sizes 4, 8, 16, 32, 64, 128, 256, 512, 1024,

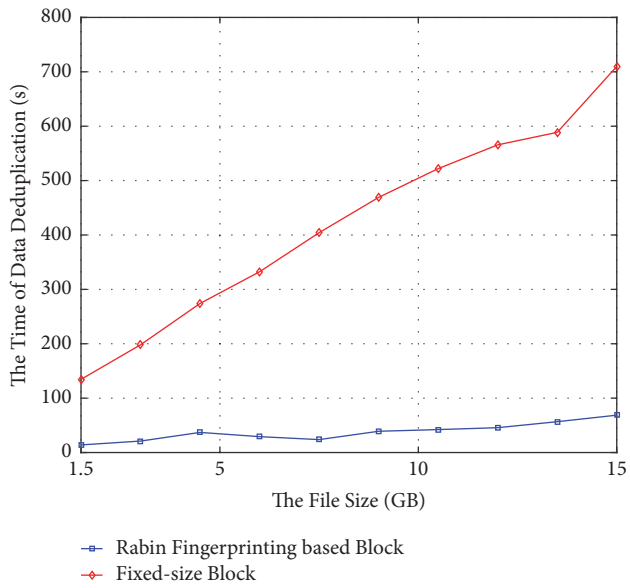


Figure 10: The time of data deduplication with the file size.

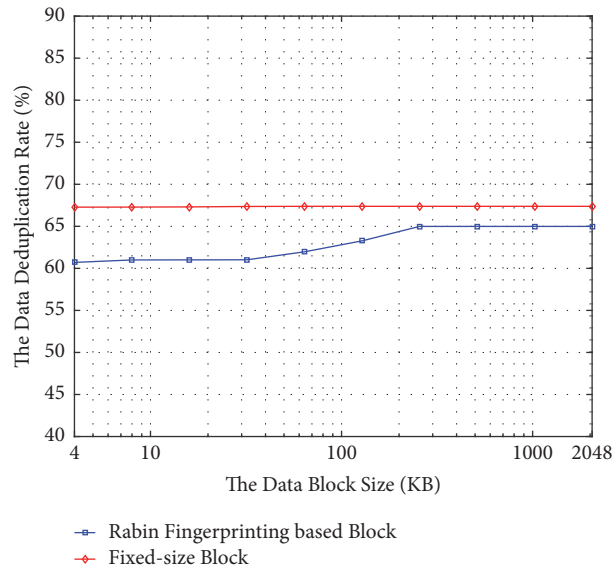


Figure 12: The data deduplication rate with the block size.

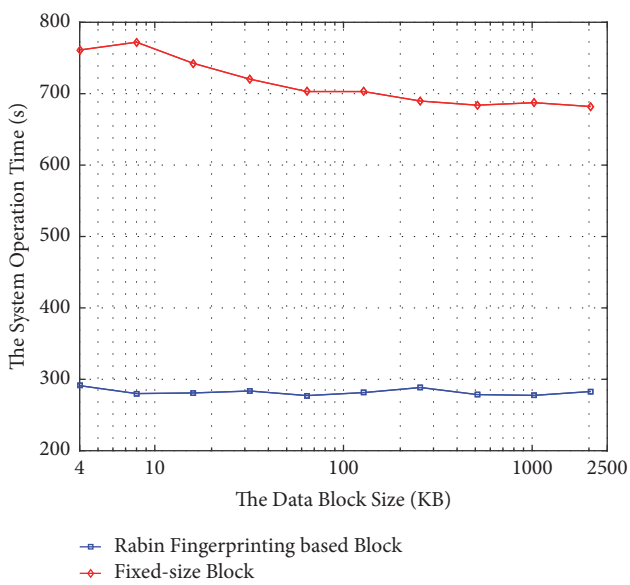


Figure 11: The system operation time with the block size.

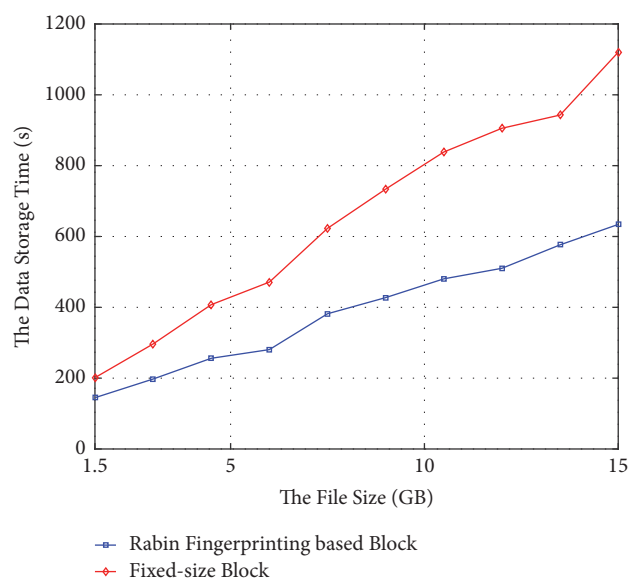


Figure 13: The data storage time with the file size.

and 2048KB, respectively. Finally, the data deduplication time and the deduplication rate are tested and compared.

Figure 11 shows the comparison of the total running time of the data deduplication system based on the Rabin fingerprinting and the fixed-size block-level data deduplication system. For the sake of clarity, the horizontal axis adopts a logarithmic scale. We can see that the performance of the data deduplication system based on the Rabin fingerprinting is better. Therefore, the use of the characteristics of duplicate data can be quickly found by using the Rabin fingerprinting, which makes the data deduplication more efficient. Figure 12 shows the data deduplication rate of the Rabin fingerprinting based scheme and the fixed-size block scheme. It can be seen that the duplicate data detection rate of the former is better than the latter. With the increase of the data block size, the deduplication rate becomes inferior. However, the data deduplication rate of the system based on the Rabin fingerprinting is always better than the fixed-size block scheme. Figure 13 is a comparison of the overall system data storage performance based on the Rabin fingerprinting and fixed-size block level, respectively. From this figure, we can see that the overall storage performance of the data deduplication scheme based on Rabin fingerprinting is better than that of the fixed-size block data deduplication system. And with the increase of the data volume of the file, the increase trend of the storage time of the fixed-size block-level data deduplication system is faster than that of the Rabin fingerprinting deduplication system.

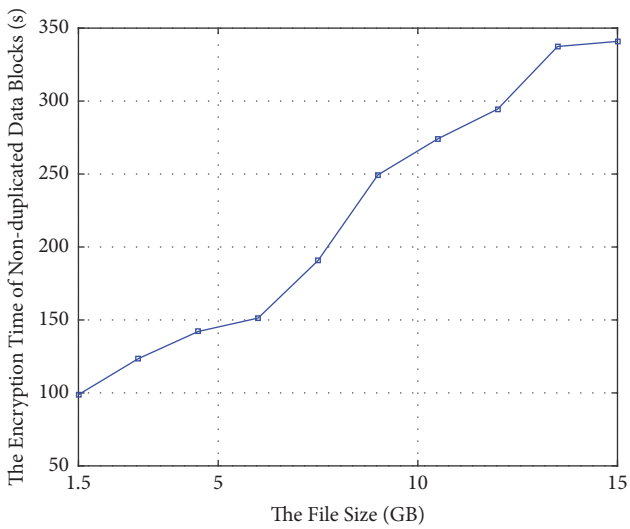


Figure 14: The encryption time of the nonduplicated data blocks with the file size.

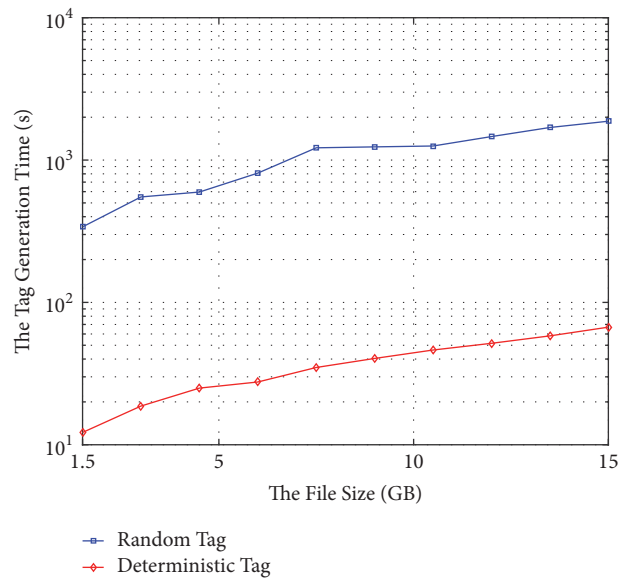


Figure 15: The tag generation time with the file size.

6.2.3. The Encryption Time of Nonduplicated Data Blocks. Besides the deduplication performance, we also consider the cost of encryption. As shown in Figure 14, the encryption time of the nonduplicated data after deduplication operation increases with the file size. Based on Figure 10, we know that the fixed-size block-level data deduplication scheme will generate more data blocks, and hence the data block encryption time will be longer than that of the Rabin fingerprinting based scheme. In particular, the fixed-size block-level data deduplication scheme needs to encrypt data before performing data deduplication. Put another way, duplicate data is also encrypted, which further increases the encryption overhead of the system.

6.2.4. The Performance Comparison of Deterministic Tags and Random Tags. In the above analysis, the schemes are of deterministic tags. In the following, we test and analyze the performance of the data deduplication systems based on the Rabin fingerprint algorithm with deterministic tags and random tags. In Figure 15, we show the performance comparison of tag generation in the data deduplication scheme based on deterministic tags and random tags.

It can be seen from Figure 16 that the time for generating random tags is much longer than that of the deterministic tags. With the increase of the number of uploaded files, the time cost of generating random tags will also increase and its rising trend is obvious. In Figure 16, we compare the storage performance of the two types of schemes. As can be seen from Figure 16, the larger the number of uploaded files, the greater the total data deduplication time of the two deduplication schemes. Generally, the random tag based deduplication system is more secure and the deterministic tag based scheme is more efficient.

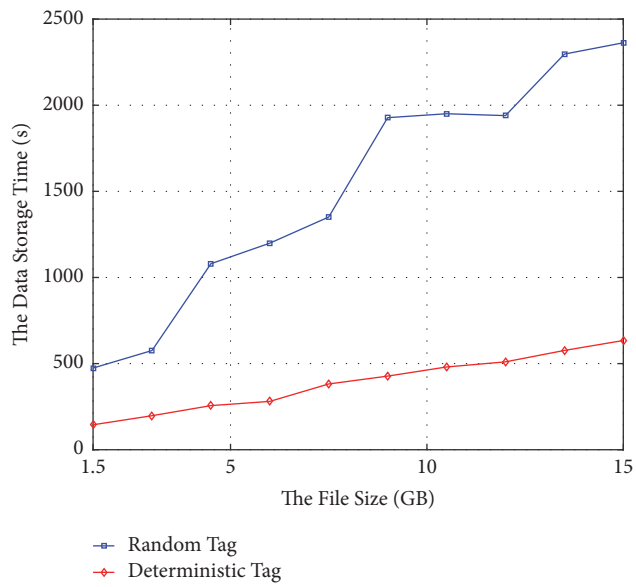


Figure 16: The storage cost comparison of random tag scheme and deterministic tag scheme.

7. Conclusions and Future Work

In this paper, we proposed two secure data deduplication schemes based on Rabin fingerprinting. The schemes are realized, respectively, based on deterministic tags and random tags. In our schemes, data deduplication is enabled before the data is outsourced to the cloud storage server, and hence both the communication cost and the computation cost are reduced. In particular, we realized variable-size block-level deduplication by using Rabin fingerprinting. The data confidentiality is kept based on convergent encryption technologies. Our security analysis showed that the proposed schemes can resist offline brute-force dictionary attacks. Our simulation results indicated that the proposed schemes are practical in terms of the efficiency.

In the future research, it would be interesting to design decentralized block-level data deduplication schemes with fine-grained access control.

Data Availability

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

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), the National Natural Science Foundation of China (nos. 61772418, 61472472, and 61402366), and the Natural Science Basic Research Plan in Shaanxi Province of China (nos. 2018JZ6001 and 2015JQ6236). Yinghui Zhang is supported by New Star Team of Xi'an University of Posts and Telecommunications (2016-02).

References

[1] H. Huang, T. Gong, N. Ye, R. Wang, and Y. Dou, "Private and secured medical data transmission and analysis for wireless sensing healthcare system," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1227–1237, 2017.

[2] Y. Zhang, X. Chen, J. Li, and H. Li, "Generic construction for secure and efficient handoff authentication schemes in EAP-based wireless networks," *Computer Networks*, vol. 75, pp. 192–211, 2014.

[3] Q. Han, Y. Zhang, X. Chen, H. Li, and J. Quan, "Efficient and robust identity-based handoff authentication in wireless networks," in *Proceedings of the in International Conference on Network and System Security*, pp. 180–191, Springer, 2012.

[4] Y. Zhang, J. Li, D. Zheng, P. Li, and Y. Tian, "Privacy-preserving communication and power injection over vehicle networks and 5G smart grid slice," *Journal of Network and Computer Applications*, 2018, http://dx.doi.org/10.1016/j.jnca.2018.07.017.

[5] 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.

[6] X. Xu, S. Fu, Q. Cai et al., "Dynamic resource allocation for load balancing in fog environment," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 6421607, 15 pages, 2018.

[7] 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.

[8] Y. Zhang, A. Wu, and D. Zheng, "Efficient and privacy-aware attribute-based data sharing in mobile cloud computing," *Journal of Ambient Intelligence & Humanized Computing*, vol. 9, no. 4, pp. 1039–1048, 2018.

[9] Z. Wu, L. Tian, P. Li, T. Wu, M. Jiang, and C. Wu, "Generating stable biometric keys for flexible cloud computing authentication using finger vein," *Information Sciences*, vol. 434, pp. 431–447, 2016.

[10] L. Yang, Z. Han, Z. Huang, and J. Ma, "A remotely keyed file encryption scheme under mobile cloud computing," *Journal of Network and Computer Applications*, vol. 106, pp. 90–99, 2018.

[11] C. Xiang, C. Tang, Y. Cai, and Q. Xu, "Privacy-preserving face recognition with outsourced computation," *Security & Communication Networks*, vol. 20, no. 9, pp. 3735–3744, 2016.

[12] Y. Zhang, M. Yang, D. Zheng, P. Lang, A. Wu, and C. Chen, "Efficient and secure big data storage system with leakage resilience in cloud computing," *Soft Computing*, 2018, http://dx.doi *Soft Computing*, 2018, .org/10.1007/s00500-018-3435-z.

[13] Y. Zhang, P. Lang, D. Zheng, M. Yang, and R. Guo, "A secure and privacy-aware smart health system with secret key leakage resilience," *Security and Communication Networks*, vol. 2018, Article ID 7202598, pp. 1–13, 2018.

[14] C. Esposito, M. Ficco, F. Palmieri, and A. Castiglione, "Smart cloud storage service selection based on fuzzy logic, theory of evidence and game theory," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 65, no. 8, pp. 2348–2362, 2016.

[15] Y. Zhang, D. Zheng, Q. Li, J. Li, and H. Li, "Online/offline unbounded multi-authority attribute-based encryption for data sharing in mobile cloud computing," *Security and Communication Networks*, vol. 9, no. 16, pp. 3688–3702, 2016.

[16] Z. Li, Y. Dai, G. Chen, and Y. Liu, "Toward network-level efficiency for cloud storage services," in *Content Distribution for Mobile Internet: A Cloud-based Approach*, pp. 167–196, Springer, 2016.

[17] Y. Zhang, D. Zheng, R. Guo, and Q. Zhao, "Fine-Grained Access Control Systems Suitable for Resource-Constrained Users in Cloud Computing," *Computing and Informatics*, vol. 37, no. 2, pp. 327–348, 2018.

[18] 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.

[19] H. Kwon, C. Hahn, D. Koo, and J. Hur, "Scalable and reliable key management for secure deduplication in cloud storage," in *Proceedings of the IEEE International Conference on Cloud Computing*, pp. 391–398, 2017.

[20] 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.

[21] J. Xiong, J. Ren, L. Chen et al., "Enhancing privacy and availability for data clustering in intelligent electrical service of IoT," *IEEE Internet of Things Journal*, 2018, http://dx.doi.org/10.1109/JIOT.2018.2842773.

[22] J. Xiong, Y. Zhang, X. Li, M. Lin, Z. Yao, and G. Liu, "RSE-PoW: a role symmetric encryption PoW scheme with authorized deduplication for multimedia data," *Mobile Networks and Applications*, vol. 23, no. 3, pp. 650–663, 2018.

[23] Y. Zhang, J. Shu, X. Liu, J. Li, and D. Zheng, "Comments on a large-scale concurrent data anonymous batch verification scheme for mobile healthcare crowd sensing," *IEEE Internet of Things Journal*, Article ID 2862381, 2018, http://dx.doi.org/10.1109/JIOT.2018.2862381.

[24] 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.

[25] X. Li, J. Li, and F. Huang, "A secure cloud storage system supporting privacy-preserving fuzzy deduplication," *Soft Computing*, vol. 20, no. 4, pp. 1437–1448, 2016.

[26] 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.

[27] 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.

[28] J. Li, J. Li, X. Dongqing, and Z. Cai, "Secure auditing and deduplicating data in cloud," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 65, no. 8, pp. 2386–2396, 2016.

[29] 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.

[30] 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.

[31] X. Xu, X. Zhao, F. Ruan et al., "Data placement for privacy-aware applications over big data in hybrid clouds," *Security and Communication Networks*, vol. 2017, pp. 1–15, 2017.

[32] D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: Deduplication in cloud storage," *IEEE Security & Privacy*, vol. 8, no. 6, pp. 40–47, 2010.

[33] J. R. Douceur, A. Adya, W. J. Bolosky, D. Simon, and M. Theimer, "Reclaiming space from duplicate files in a serverless distributed file system," in *Proceedings of the 22nd International Conference on Distributed Systems*, pp. 617–624, IEEE, Austria, July 2002.

[34] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-locked encryption and secure deduplication," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 296–312, Springer, 2013.

[35] 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.

[36] M. Abadi, D. Boneh, I. Mironov, A. Raghunathan, and G. Segev, "Message-locked encryption for lock-dependent messages," in *Advances in Cryptology – CRYPTO 2013*, vol. 8042 of *Lecture Notes in Computer Science*, pp. 374–391, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[37] J. Li, C. Qin, P. P. Lee, and J. Li, "Rekeying for encrypted deduplication storage," in *Proceedings of the 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 618–629, Toulouse, France, June 2016.

[38] Y. Zhang, R. H. Deng, X. Liu, and D. Zheng, "Blockchain based efficient and robust fair payment for outsourcing services in cloud computing," *Information Sciences*, vol. 462, pp. 262–277, 2018.

[39] Y. Zhang, R. H. Deng, X. Liu, and D. Zheng, "Outsourcing service fair payment based on blockchain and its application in cloud computing," *IEEE Transactions on Services Computing*, Article ID 2864191, 2018, http://dx.doi.org/10.1109/TSC.2018.2864191.

[40] Y. Zhang, R. H. Deng, J. Shu, K. Yang, and D. Zheng, "TKSE: Trustworthy Keyword Search Over Encrypted Data With Two-Side Verifiability via Blockchain," *IEEE Access*, vol. 6, pp. 31077–31087, 2018.

[41] J. Li, Y. K. Li, X. Chen, P. P. 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.

[42] S. Wang, "Use of gpu architecture to optimize rabin fingerprint data chunking algorithm by concurrent programming," Tech. Rep., California State University, Long Beach, ProQuest Dissertations Publishing, 2016, https://books.google.com.sg/books?id=GyttnQAACAAJ.

[43] T. Jiang, X. Chen, Q. Wu, J. Ma, W. Susilo, and W. Lou, "Secure and efficient cloud data deduplication with randomized tag," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 3, pp. 532–543, 2017.

[44] H. Su, D. Zheng, and Y. Zhang, "An efficient and secure deduplication scheme based on rabin fingerprinting in cloud storage," in *Proceedings of the IEEE International Conference on Computational Science and Engineering*, pp. 833–836, 2017.

[45] A. Z. Broder, "Some applications of Rabin's fingerprinting method," in *Sequences II*, pp. 143–152, Springer, 1993.

[46] K. R. Jayaram, C. Peng, Z. Zhang, M. Kim, H. Chen, and H. Lei, "An empirical analysis of similarity in virtual machine images," in *Proceedings of the the Middleware 2011 Industry Track Workshop*, pp. 1–6, Lisbon, Portugal, December 2011.

[47] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems," in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, pp. 491–500, Chicago, Illinois, USA, October 2011.

[48] Y. Zhao and S. S. Chow, "Towards proofs of ownership beyond bounded leakage," in *Proceedings of the International Conference on Provable Security*, Provable Security, pp. 340–350, Springer, 2016.

[49] L. P. Cox, C. D. Murray, and B. D. Noble, "Pastiche: Making backup cheap and easy," *ACM SIGOPS Operating Systems Review*, vol. 36, pp. 285–298, 2002.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 1937670, 12 pages
https://doi.org/10.1155/2018/1937670

Research Article

Enhanced Adaptive Cloudlet Placement Approach for Mobile Application on Spark

Yiwen Zhang,[1] Kaibin Wang,[1] Yuanyuan Zhou,[1] and Qiang He[2]

[1]*Key Laboratory of Intelligent Computing & Signal Processing, Ministry of Education, Anhui University, Hefei, Anhui 230031, China*
[2]*School of Software and Electrical Engineering, Swinburne University of Technology, Melbourne, Australia*

Correspondence should be addressed to Yuanyuan Zhou; yyzhouahu@qq.com

Received 6 June 2018; Accepted 9 August 2018; Published 2 September 2018

Academic Editor: Yuan Yuan

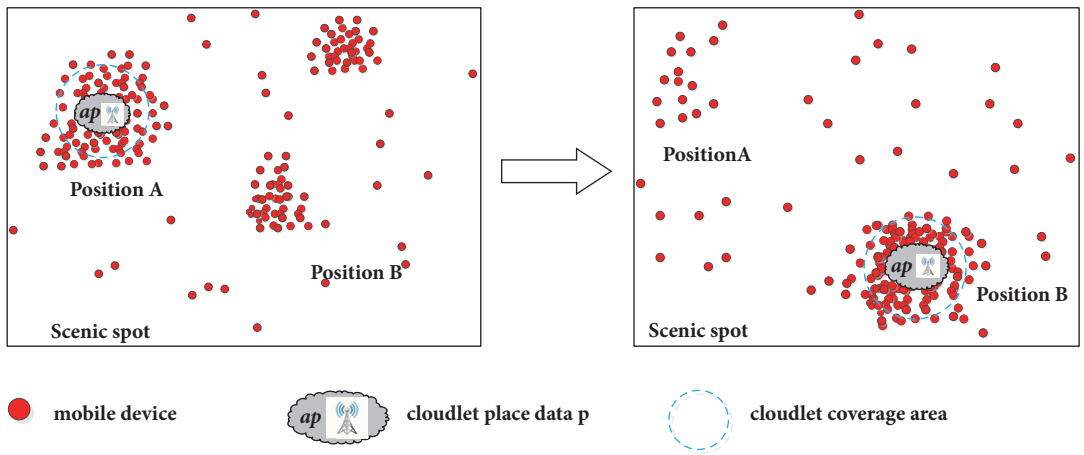
The applications of mobile devices are increasingly becoming computationally intensive while the computing capability of the user's mobile device is limited. Traditional approaches offload the tasks of mobile applications to the remote cloud. However, the rapid growth of mobile devices has made it a challenge for the remote cloud to provide computing and storage capacities with low communication delays due to the fact that the remote cloud is geographically far away from mobile devices. Reducing the completion time of applications in mobile devices through the technical expending mobile cloudlets which are moving collocated with Access Points (APs) is necessary. To address the above issues, this paper proposes EACP-CA (**E**nhanced **A**daptive **C**loudlets **P**lacement approach based on **C**overing **A**lgorithm), an enhanced adaptive cloudlet placement approach for mobile applications in a given network area. We apply the CA (Covering Algorithm) to adaptively cluster the mobile devices based on their geographical locations, the aggregation regions of the mobile devices are identified, and the cloudlet destination locations are also confirmed according to the clustering centers. In addition, we can also obtain the traces between the original and destination locations of these mobile cloudlets. To increase the efficiency, we parallelize CA on Spark. Extensive experiments show that the proposed approach outperforms the existing approach in both effectiveness and efficiency.

1. Introduction

The rapid development of the mobile Internet and the Internet of Things has promoted the emergence of various new types of services, which has led to explosive growth in mobile communication traffic over the past few years. Mobile devices have gradually replaced personal computers as one of the main tool, which people can use in daily work, socialization, and entertainment. Simultaneously, the mobile applications are also increasingly becoming computationally intensive [1–3]. These vast amounts of mobile applications accordingly bring huge computing capacity, storage capacity requirements, and low latency requirements. The previous methods allow these mobile devices to directly access the remote cloud (e.g., Amazon) [4–6], deploying all services to the remote cloud which will not only lead to the great increase of the network load and cause a long delay in the network, but also put higher demands on bandwidth and delay performance of network. The remote cloud is far away from its users and the network delay incurred by processing the requirements can be very costly. These above situations are especially intolerable in real-time demand of applications where the rapid response time is vital for users who take the mobile devices.

In order to cope with the long latency problem, recent works have proposed the cloudlets with strong computing and storage capacity, which are typically collocated at the AP in a network and can be accessed by users via wireless connection [3, 7, 8]. The critical advantage of the cloudlets is that the cloudlets can be deployed physical proximity close to users which can shorten the transmission latency and improve the user experience of using interactive applications.

Although there is an increasing number of researches in mobile cloudlet offloading technology [9–11], fewer researches about how to place cloudlets in a given network region are conducted [3, 7, 12–14]. It is necessary to pay



(a) The original mobile devices distribution and the cloudlet placement position

(b) The new generated cloudlet placement position after the movement of the mobile devices

Figure 1: Motivation example of adaptive cloudlet placement.

attention to how cloudlets should be placed in a given network since the suitable cloudlets placements contribute to enhancing the cloud service for dynamic context-aware mobile applications. The price of the cloudlets is so expensive that the number of the cloudlets is limited, so we should maximize the utilization of cloudlets by a suitable cloudlets placement. Some current researches pay attention to the cloudlets placement to alleviate the low utilities of cloudlets [3, 12–14], the cloudlets placement in their studies are fixed, but, in practice, users are constantly moving. Fixed-placed cloudlets cannot maintain efficiency in responding to the application requirements for the constant mobile devices. In order to explain this problem more vividly, we use Figure 1 to illustrate the issue. Figure 1 presents an example with many mobile devices and a cloudlet. In Figure 1(a), those mobile devices with GPS location are randomly distributed in the scenic spot and a cloudlet with AP being in position A. Those mobile devices are constantly moving over time. And at time t', the crowd moves close to position B. If the cloudlet is still deployed at position A, the coverage range of cloudlet is so small which leads to the low utility of the cloudlet. But if the cloudlet moves to position B described in Figure 1(b), the scope of cloud coverage has increased significantly. Therefore, to improve the performance of the cloudlet service in the dynamic scenario, it is significant to propose an enhanced adaptive mobile cloudlets placement according to practical distributions of mobile devices.

To address the above issue, this paper proposes EACP-CA (**E**nhanced **A**daptive **C**loudlets **P**lacement approach based on **C**overing **A**lgorithm), a novel approach for cloudlets placement. Given a known mobile devices activity area, according to the characteristics of the data and being independent of the initial centers [8, 15], EACP-CA first employs CA with "blind" features to adaptively partition mobile devices into clusters based on the physical distance between their positions. Then we adjust the central positions obtained by CA through the obtained graph path. Next, we execute *confirmation of the center positions operation* and get the desired cloudlets central positions according to the distribution of the mobile devices. We also need to obtain the pair of the original mobile cloudlet positions and the destination position of them. Finally, we conduct the enhanced adaptive cloudlets placement approach to finish the mobile cloudlets placement. To improve the efficiency, we parallelize CA on Spark.

The major contributions of this paper are as follows:

(1) We employ the adaptive clustering algorithm CA to cluster the mobile devices in given regions. CA based on the quotient space theory has "blind" features without determining the number of clusters in advance; the algorithm can automatically identify the number of clusters based on the characteristics of the data and is independent of the initial centers. And we parallelize the CA on Spark to increase the efficiency of EACP-CA in processing big data.

(2) We implement the enhanced adaptive cloudlets placement and obtain the traces of mobile cloudlets any distribution of the mobile devices with changing over time.

(3) We conduct extensive experiments to comprehensively compare EACP-CA with existing k-means based approach.

This paper is organized as follows. Section 2 reviews the related work. Section 3 overviews the preliminary knowledge. Section 4 introduces enhanced adaptive cloudlet placement approach for mobile applications. Section 5 presents the experimental results and analysis, and Section 6 concludes this paper.

2. Related Work

Mobile cloud computing provides information technology service environment and cloud computing capabilities within the radio access network closest to the user's mobile devices,

aiming to further reduce latency, enhance network operating efficiency, boost service distribution, and improve user experience, which can also increase the computing capacity of mobile devices by offloading the workload to clouds [1, 2, 9, 16–19]. Many kinds of researches have been conducted on mobile cloud computing [2, 16, 20–22], but most of them are about remote clouds which are physically far away from the uses and cloudlet offloading technology. The remote clouds lead to a long latency, which is negative for improving the user experience and the performance of cloud services [4, 5]. In view of the above issues, researchers present the cloudlets with computing and storage resource-rich which can deploy at APs in a network [3, 7]. The cloudlets are deployed physically close to the users and act as offloading destinations of the mobile user which can significantly short the response time for the users' requirements and also can reduce the energy consumption of mobile devices.

There are some studies investigated the cloudlets. To name a few, Jia et al. [3] study the cloudlet placement and mobile user allocation to the cloudlets at user dense region of the wireless metropolitan area network (WMAN) and assign mobile users to the placed cloudlets while balancing their workload. Xu et al. [12] focused the cloudlet placement problem in a large-scale WMAN consisting of many wireless APs, in which capacitated cloudlets need to find the best deployment locations within a given set of candidate locations in WMAN. The objective is to minimize the average access delay between the mobile users and the activated cloudlets serving the users. Ma et al. [13] propose a New Heuristic Algorithm (NHA) and a Particle Swarm Optimization (PSO) algorithm for the delaying problem. In the above approaches, cloudlets are fixed in the placement, which cannot maintain efficiency in response to the requirements of applications for frequently moving devices. There are some researchers proposed the movable cloudlets to enhance the performance of the cloud service with the mobile users. Xiang et al. [7] propose self-adaptive edge cloud placement based on the locations of mobile applications. The core idea of the method is to maximize the number of mobile devices that are covered by the known active area of the edge cloud. They use the k-means algorithm to conduct the clustering process. However, k-means clustering algorithm has some drawbacks that the number of clusters k cannot be easily determined, and the clustering results heavily rely on the initial centers of random selection. And the selection of the initial center has a significant impact on the final clustering result and is easily disturbed by outliers [15, 23, 24].

In this paper, we propose EACP-CA, a novel approach for enhanced adaptive cloudlets placement. EACP-CA efficiently addresses the limitations and issues of most existing clustering approaches with a novel covering-based clustering technique. The CA algorithm we employed has "blind" features, without determining the number of clusters in advance, which can automatically identify the number of clusters based on the characteristics of the data and is independent of the initial centers. And EACP-CA can efficiently mitigate the issue of data scalability on Spark.

3. Preliminaries

In this section, we formally define the problem accurately to facilitate further introduction on enhanced adaptive cloudlet placement for mobile applications. Before conducting the introduction, we summarize the notions used throughout this paper in Table 1 to simplify the discussion.

Most existing studies in mobile edge computing focus on the limited of computing capacity, storage, and energy savings of mobile devices by offloading high-complexity and computing-intensive tasks from mobile devices to remote clouds [2, 9, 10]. However, the approach of offloading computing tasks to the cloud computing center not only brings about a large amount of data transmission and increases the network load, but also increases the transmission delay, which has a certain impact on the delay-sensitive service and the user experience [15]. Therefore, to effectively solve the requirements of high bandwidth and low latency brought about by the rapid development of the mobile Internet and the Internet of Things, the mobile edge computing has received extensive attention from the academic community and the industry. The cloudlets play an important role in the mobile edge computing which can significantly enhance the performance of mobile devices and meet the mobile users' response time requirements simultaneously, as shown in Figure 1. The mobile cloud framework consists of three main parts presented in Figure 2. Part 1 is mobile device clients, Part 2 is cloudlets, and Part 3 is the remote cloud. The mobile device clients can directly access the cloud service through the APs or can connect to the network through the wireless network. What is more, there is also a remote cloud which can be accessed from APs through the Internet. When the cloudlets cannot meet the request of the mobile devices, some computationally intensive tasks and data can be offloaded to the remote cloud for processing. Mobile devices can quickly access nearby cloudlets through APs to obtain cloud storage and cloud computing resources, so cloudlets can help reduce the access latency.

Effectively dealing with the cloudlet placement problem in WMAN consisting of many APs can contribute to enhancing the cloud services for dynamic mobile computationally intensive applications and improve the user experience. In this paper, we apply a rectangle activity area to place the cloudlets. In addition, other shaped areas can also be split by multiple different size rectangles.

3.1. Cloudlet Placement Strategy. If cloudlet l_m with the central position $p(x, y)$ is providing service to the users with the mobile devices at time t, the number of the mobile devices covered by l_m at time t should meet the condition that $dc\_l_m^{p_{(t)}} \geq \sigma$. Our goal is to find optimal cloudlet placement in A and maximize the total number of covered mobile devices which is calculated by the following objective function:

$$AN(t) = \left| \bigcup_{m=1}^{M} dc\_l_m^{p_{(t)}} \right| \tag{1}$$

Assume cloudlet l_m is placed at $p(x, y)$ at time t, and, in the time range $(t, t']$, the mobile devices in A move randomly.

Table 1: Mathematical notation.

Symbol	Explanation
A	Denotes the set of points in mobile devices activity area, $A = \{(x, y) \mid 0 \le x \le W,\ 0 \le y \le H\}$
MD	Denotes the set of mobile devices, $D = \{d_1, d_2, \ldots, d_N\}$
MDP	Denotes the set of mobile devices positions
d_n	Denotes the nth $(1 \le n \le N)$ mobile device in D
$d_n^{p(t)}$	Denotes the position of dn at time t, $d_n^{p(t)} = (d_n^{p(t)}\_x, d_n^{p(t)}\_y)$, where $d_n^{p(t)}\_x$ and $d_n^{p(t)}\_y$ represents the latitude and longitude of the location
L	Denotes the set of the cloudlets, $L = \{l_1, l_2, \ldots, l_N\}$
l_m	Denotes the mth cloudlet
CL	Denotes the set of the central position of the cloudlets
$l_m^{p(t)}$	Denotes the central position of l_m at time t, $l_m^{p(t)} = (l_m^{p(t)}\_x, l_m^{p(t)}\_y)$
AP	Denotes the set of the Aps, AP $= \{ap_1, ap_2, \ldots ap_m\}$
rm	Denotes the converage radius for l_m
$dc\_l_m^{p(t)}$	Denotes the mobiles devices collection of l_m^p at time t, $dc\_l_m^{p(t)} = \left\{d_n^{p(t)} \mid dis\left(mdp_n^{p(t)}, l_m^{p(t)}\right) \le r,\ 1 \le\le N\right\}$
$dis\left(mdp_n, l_m^{p(t)}\right)$	Denotes the Euclidean distance between $mdp_n^{p(t)}$ and $l_m^{p(t)}$, $dis(mdp_n^{p(t)}, l_m^{p(t)}) = \sqrt{(mdp_n^{p(t)}\_x - l_m^{p(t)}\_x)^2 + (mdp_n^{p(t)}\_y - l_m^{p(t)}\_y)^2}$
dc_cp_i	Denotes the mobile devices collection of cp_i
σ	Denotes the density threshold for cloudlet placement judgment
$AN(t)$	Denotes the number of covered mobile devices by all cloudlets
$dc\_l_m^{p(t')}$	Denotes the mobile device collection of l_m^p at time t'
V	Denotes the available positions that cloudlets can only stay
v_u	Denotes the uth available position that cloudlets can stay
E	Denotes the edges between adjacent position in V
e_{uv}	Denotes the edge between two APs v_u and v_v
W	Denotes the weight of all cloudlet available positions
G	Denotes the paths graph in A, $G = \{E, V, W\}$
$p(x, y)$	Denotes the coordinates of position

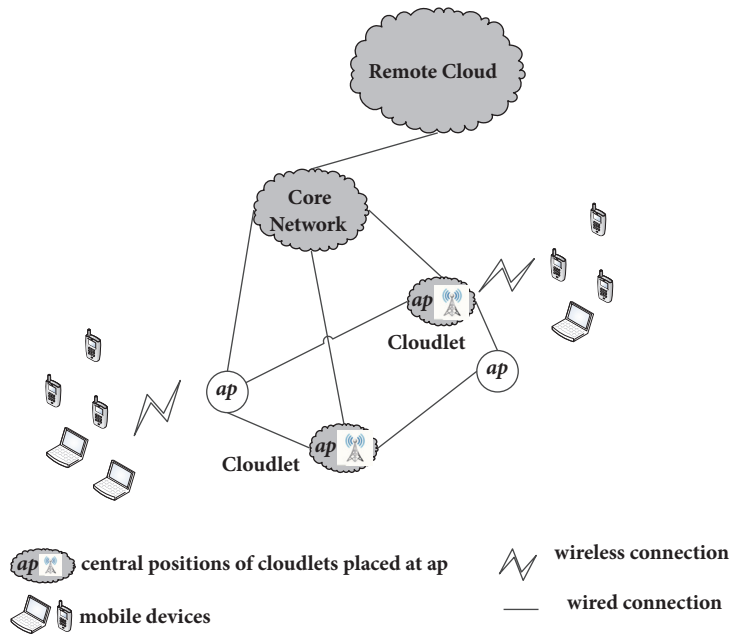


Figure 2: The mobile cloud framework.

The place position of cloudlet l_m at time t may not suitable for distribution of mobile devices after moving at time t'. In order to maintain the cloudlets working efficiently, we should propose a cloudlet movement strategy as the move reference for the cloudlets.

3.2. Cloudlet Movement Strategy. After mobile devices have moved at time t', if the cloudlet l_m is still placed at the position $p(x, y)$ at the time t, the mobile devices' collection of position $p(x, y)$ at time t' is denoted as $dc\_l_m^{p_{(t')}}$, and another place position $p'(x', y')$ for cloudlet l_m at time t' is obtained by adaptive cloudlet placement approach, and satisfying the cloudlet placement strategy is denoted as $dc\_l_m^{p'_{(t')}}$. If meet the condition that $dc\_l_m^{p'_{(t')}} \geq dc\_l_m^{p_{(t')}}$ is met, and there are no other cloudlets placing around p' at time t' within radius r, cloudlet l_m will move from $p(x, y)$ to $p'(x', y')$.

4. Enhanced Adaptive Cloudlet Placement Approach for Mobile Applications

The placement of the cloudlets has a significant impact on the resource utilization of the cloudlets. Inappropriate placement of cloudlets can cause the severe imbalance in the edge cloud load. Some cloudlets are overloaded, while others are underloaded or even idled and reducing the mobile users' response time requirements.

To obtain the proper cloudlet placement, now we introduce our main approach to the cloudlet placement. The enhanced adaptive cloudlet placement consists of three stages. Stage 1 performs the parallel CA clustering algorithm to obtain the central positions of mobile devices gathering place. Stage 2 confirms the cloudlet locations. Stage 3 realizes the adaptive cloudlet placement. The three stages are discussed in detail as follows.

Stage 1. We observe that mobile devices-dense regions of A areas are suitable to place the cloudlets, which means that cloudlets are situated close to a large number of devices and can reduce the average network latency between mobile devices and cloudlets. Therefore, we propose a clustering algorithm named CA to adaptive identify the central positions of device gathering place. Algorithm 1 presents the pseudocode for the CA algorithm and we use Table 2 to simply introduce these functions in Algorithm 1. The detailed description of these functions is in Section 3.3 in [23]. In order to better illustrate the effectiveness, we also discuss the time complexity of the CA.

4.1. Time Complexity Analysis of the Algorithm 1. In Algorithm 1, the computational complexity of line (5) is $O(n)$ because dataset *MDP* contains a maximum of n points. Similarly, the computational complexities of lines (5) and (6) are also $O(n)$, and those of lines (7), (8), and (9) are also $O(n)$ because the number of clusters is smaller than n. Lines (10)-(15) will be repeated until the data points in the cluster do not change. Lines (3)-(15) must also be repeated until all of the data points in *MAP* are covered, and the number of repetitions *num_C* is much smaller than n. In line (3), the radius of a cluster is the average distance between the center of the cluster and all of the data points that are not covered by any clusters. On average, each newly created cluster covers half of the uncovered data points, and the computational complexity is $O(\log n)$. In line (17), the computational complexity is $O(p)$ because there is a maximum of p clusters after the initial covering process. Similarly, the computational complexity of line (18) is $O(p)$

Input: *MDP*
Output: Results of parallel covering with granularity analysis–A
set of clusters $CP = \{CP_1, CP_2, \ldots\}$
Begin
center c = null
Set $C_u = MDP$
do
center $c \longleftarrow$ get_center(C_u)
radius $r \longleftarrow$ get_weight_radius(c,C_u)
Covering C_{form} = get_covering(c,r,C_u)
c $\longleftarrow$ get_centroid(C_{form})
$r \longleftarrow$ get_weight_radius(c,C_u)
Covering C_{last} = get_covering(c,r,C_u)
while C_{last}.subtractByKey(C_{form}) > 0
$C_{form} \longleftarrow C_{last}$
c $\longleftarrow$ get_centroid(C_{form})
$r \longleftarrow$ get_ radius_centroid(c, C_u)
C_{last} = get_covering(c,r,C_u)
end while
while($C_u \neq \emptyset$)
Do Split Operation
Do Merge Operation
return $CP = \{CP_1, CP_2, \ldots\}$
End

Algorithm 1: CA (*MDP*).

Input: *MDP*, *V*
Output: A set of mobile device central positions *CCP*
Begin
centers $\longleftarrow$ CA(*MD*)
$CP \longleftarrow \emptyset$
for $i = 1$ to k
do
$Pos \longleftarrow \underset{v_j}{\text{argmin}}\, dis(cp_i, v_j)$ (j=1 to U)
add *Pos* to *CCP*
end for
return *CCP*
End

Algorithm 2: Center position adaptive identification (*MDP*, *V*).

because there is a maximum of p clusters. The number of clusters is much smaller than n. Thus, the computational complexity of Algorithm 1 is $O(n) \times O(\log n) + O(p) = O(n \log n)$.

Therefore, we obtain the central positions of mobile devices gathering place through the CA clustering algorithm.

Considering the security and efficiency of path researching in adaptive cloudlet placement, the whole cloudlets can only place to APs [25]. Then we employ the path graph G to adjust the obtained central positions and generate moving traces of the mobile cloudlets. Algorithm 2 presents the pseudocode for the adjusting of the central positions obtained by CA.

Stage 2. We obtained several mobile device central positions after performing Stage 1. Then we will conduct the confirmation operation to filter the undesired center positions. The confirmation of center positions operation is introduced as follows.

4.2. Confirmation of Center Positions Operation. Through Stage 1, we get the mobile device central positions set *CCP*. For each center position cp_i in *CCP*, the confirmation of center positions operation filters the mobile devices that the distances between cp_i and each mobile device position are shorter than radius r. And if cp_i center position contains more than σ mobile devices, then add the cp_i center position to the set of cloudlet central position denoted by *CP*. After the confirmation operation, we get the set of cloudlet central position *CP*. We use the following formula for a brief explanation.

$$CP = \left\{ cp_i \mid cp_i \in CCP \text{ and } \left| dc_{cp_i} \right| \geq \sigma \right\} \tag{2}$$

Stage 3. After the above confirmation operation, we get the desired mobile devices central positions. According to the distribution of the mobile devices in A, if each cloudlet moves to the center position which is surrounded by dense mobile devices, these cloudlets will achieve high utilities. Therefore we will propose a method to choose a suitable center position in *CP* as its destination position. Assuming we have already known the previous cloudlets, we describe the selection strategy for cloudlets location to introduce the selection of each cloudlet. For the subproblem of the cloudlet selection, a straightforward idea is to greedily select the

Table 2: Explanations of functions.

Functions	Explanation
get_center(C_u)	Denotes a function has the function of obtaining the center of the data set C_u. The specific acquisition process of the center of the circle is to calculate the data point closest to the center of gravity of the data set C_u.
get_weight_radius(c,C_u)	Denotes a function has the function of obtaining the weighted radius based on center c and data set C_u.
get_covering(c,r,C_u)	Denotes a function has the function of obtaining the centroids of the current spheres continually according to the obtained center and radius and obtain new clusters until the number of clusters in the data points does not increase.
get_centroid(C_{form})	Denotes a function has the function of obtaining the center of gravity of data set C_u.
Split Operation	Denote a function has the function of combing the most similar pair of clusters into a new cluster.
Merge Operation	Denote a function has the function of splitting the clusters with more data points.

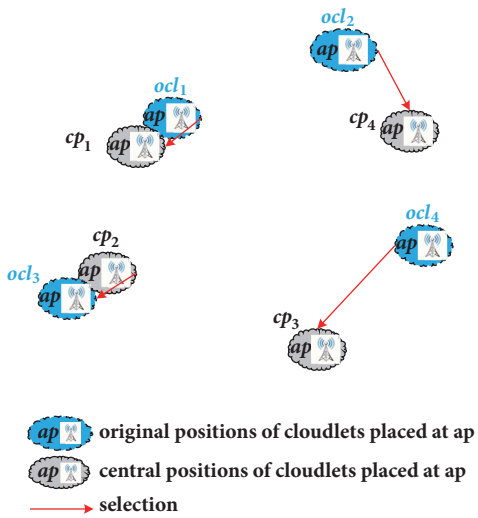


FIGURE 3: The example of locations selection mechanism.

cloudlet with the minimum distance. The center position selected by each cloudlet is according to the distance between cloudlet previous location and the mobile devices central positions.

4.3. Locations Selection Mechanism. Supposing that we already know the original positions and the destination positions of cloudlets, i.e., the central positions, the cloudlets denoted by $L = \{l_1, l_2, \ldots, l_m\}$ and the original positions denoted by $OCL = \{ocl_1, ocl_2, \ldots, ocl_m\}$, the destination positions of cloudlets are obtained by the above stages which are denoted by $CP = \{cp_1, cp_2, \ldots, cp_m\}$. We compute all the distances between cl_i and cp_j, and get distance set $OCP = \{ocp_1, ocp_2, \ldots, ocp_m \mid ocp_i = (cp_1, \ldots, cp_p \ldots, cp_q, \ldots, cp_m)$ where $1 \le p \le q \le m$ and $dis(ocl_i, cp_p) < dis(ocl_i, cp_q)\}$. While not all cloudlets have selected a central position, we add ocl_i to the intermediate set i_c_j that cp_j is the nearest central position of l_i in ocp_i, and then for each i_c_j that is not empty. If each cp_i corresponding to i_c_j have not selected a cloudlet or a nearest cloudlet selected cp_i this time, we obtain cp_i' nearest l_j in i_c_u and cp_i selects ocl_j. If there are some existing cloudlets in i_c_j are not selected, we delete the cp_i from ocp_a that ocl_a in i_c_j is not selected. We repeat those operations until all cloudlets have selected a central position. In order to explain the locations select mechanism process visually, we use a specific example to explain which are presented in Figure 3.

As we can see from Figure 3, there are 4 cloudlets, l_1, l_2, l_3, l_4, 4 original cloudlet positions, ocl_1, ocl_2, ocl_3, ocl_4, and 4 central positions, cp_1, cp_2, cp_3, cp_4. For ocp_1, ocp_2, ocp_3, *and* ocp_4, *we, respectively, get* $ocp_1 = (cp_1, cp_2, cp_4, cp_3)$, $ocp_2 = (cp_4, cp_1, cp_2, cp_3)$, $ocp_3 = (cp_2, cp_1, cp_3, cp_4)$, and $ocp_4 = (cp_4, cp_3, cp_1, cp_2)$. Now, each of cloudlets does not select a *cp*. For all cloudlets, we figure out that $i\_c = \{i\_c_1 = \{ocl_1\}, i\_c_2 = \{ocl_3\}, i\_c_4 = \{ocl_2, ocl_4\}\}$. Then we will deal with the unempty i_c_i. For each cloudlet in i_c_1, cp_1 has not selected a cloudlet, we get the cp_1' nearest cloudlet ocl_1 in i_c_1 and cp_1 selects ocl_1. For each cloudlet in i_c_2, cp_2 has not choose a cloudlet, we obtain the cp_2' nearest cloudlet ocl_3 in i_c_2, and cp_2 selects ocl_3. For each cloudlet in i_c_4, cp_4 has not selected a cloudlet, we acquire the cp_4' nearest cloudlet ocl_2 in i_c_4 and cp_4 chooses ocl_2, ocl_4 in i_c_4 is not selected. So we delete the cp_4 in ocp_4, now the $ocp_4 = (cp_3, cp_1, cp_2)$. We can find that ocl_4 is still not selected a *cp*, so we will repeat these operations. For each cloudlet in *OCL* which has not selected a *cp*, we come to the updated $i\_c = \{i\_c_1 = \{ocl_1\}, i\_c_2 = \{ocl_3\}, i\_c_3 = \{ocl_4\}, i\_c_4 = \{ocl_2\}\}$. For each cloudlet in i_c_1, cp_1' nearest cloudlet ocl_1 chooses cp_1, we obtain the cp_1' nearest cloudlet ocl_1 in i_c_1 and cp_1 selects ocl_1. For each cloudlet in i_c_2, cp_2' nearest cloudlet ocl_3 chooses cp_2, we get the cp_2' nearest cloudlet ocl_3 in i_c_2 and cp_2 selects ocl_3. For each cloudlet in i_c_3, cp_3 has not choose a *cp*, we acquire the cp_3' nearest cloudlet ocl_4 in i_c_3 and cp_3 selects ocl_4. For each cloudlet in i_c_4, cp_4' nearest cloudlet ocl_2 chooses cp_4, we get the cp_4' nearest cloudlet ocl_2 in i_c_2 and cp_4 chooses ocl_2. So far, all the cloudlets have selected a *cp*.

After the selection, we will employ the enhanced adaptive cloudlet placement approach to obtain the moving trace of each cloudlet. As introduced above, the previous cloudlet locations and the destination positions of all cloudlets all belonged to *V*. Consequently, the problem of the generation of moving trace can transform to the shortest path problem. We use the Dijkstra algorithm for generation of moving trace and then the moving traces will be transmitted to corresponding cloudlet moving. The ultimate overall process of enhanced cloudlets placement is presented in Algorithm 3.

5. Experience Evaluation

In this section, we present the performance evaluation of our proposed EACP-CA approach. And we also experimentally compare our proposed approach with k-means based approach.

The Spark cluster used to implement CA is built on a cluster with three connected nodes. Master/Slave is ThinkCentre M8500t-N000 Intel(R) Core(TM) i7-4790 CPU 3.60GHZ, 4cores, 8CPUs, DDR3 1600MHz SDRAM; 1 disk on the master and 2 disks on the slave: 1TB, 7.2K RPM SATA Hard Drive.

5.1. Experimental Setup. The experiments are conducted on the randomly generated location datasets deferring to the Gaussian distribution which can better simulate real-world applications. The parameters of Gaussian distribution used for generating location datasets are illustrated in Table 3, where μ and σ represent mean and standard deviation, respectively. 200, 300, and 400 represent the number of mobile devices. We will also introduce the parameters used in our experiments. The shape of mobile device activity area A is set to square and the ranges of x-axis and y-axis of this area are both in $[0\text{m}, 180\text{m}]$. The locations of mobile device are randomly generated and distributed in A. The numbers of the mobile devices are moderate values of $N \in \{200, 300, 400\}$. We set the density threshold σ for cloudlet placement judgment as 30, the coverage radius for cloudlet as 30m, and the time of period of experiments t as 30 minutes. We change the number of APs M from 1 to 7. This way,

Table 3: The parameters of Gaussian distribution.

Devices	200 (μ, σ)	300 (μ, σ)	400 (μ, σ)
Fixed devices	(30,25),(50,16), (110,18),(150,19)	(30,25),(50,16), (110,18),(150,19)	(30,25),(50,16), (110,18),(150,19)
Moving devices of t_0	(60,19),(80,13), (150,25),(140,30)	(60,22),(65,20), (130,30),(140,22)	(55,35),(40,40), (110,15),(130,30)
Moving devices of t_1	(55,28),(60,15), (150,25),(120,30)	(60,22),(65,20), (130,30),(140,22)	(55,35),(40,40), (110,40),(130,30)
Moving devices of t_2	(60,19),(80,15), (150,25),(150,30)	(60,22),(65,20), (130,30),(140,22)	(55,35),(40,40), (110,40),(130,30)

Input: G
Output: The moving trace of all cloudlets
Begin
(1) **for** $t = 1$ to $t_{\max}$
(2) **do**
(3) $CCP \longleftarrow$ **Algorithm** 2 (***MDP***, ***V***)
(4) $CP \longleftarrow$ filteringCCP through ***Confirmation of center positions operation***
(5) $(ocl_i \in OCL,\ ocl_i$ selected $cps) \longleftarrow$ ***Locations selection mechanism***
(6) **for** $i = 1$ to $|OCL|$
(7) **do**
(8) **if** l_i have selected a cp **then**
(9) shortest trace tra_i from ocl_i to its selected $cp \longleftarrow$ Dijkstra algorithm (G, l_i)
(10) transmit tra_i as the moving description to l_i
(11) **end if**
(12) **end for**
(13) **end for**
End

Algorithm 3: Enhanced adaptive cloudlets placement (G).

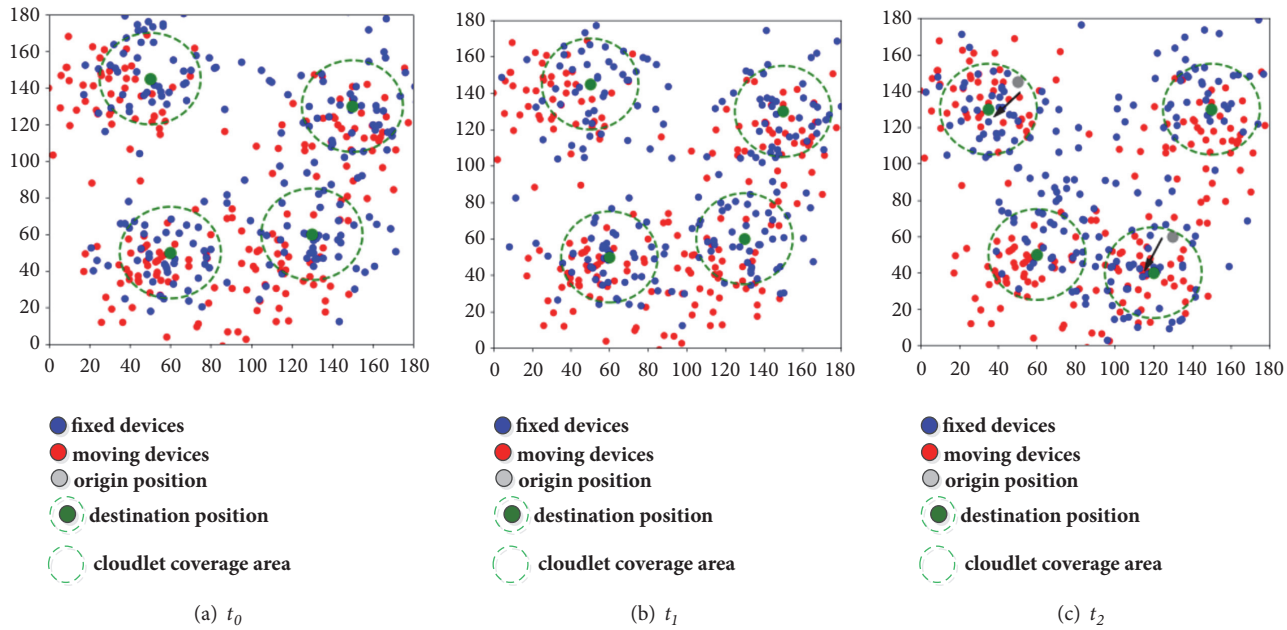
we comprehensively evaluate EACP-CA's ability to handle datasets with different characteristics in various application scenarios.

5.2. Performance Evaluation and Comparison. In this section, we will evaluate the performance of our proposed approach and compare it with k-means based approach through the number of movable cloudlets covered mobile devices.

In order to intuitively display the distributions of mobile devices and cloudlets in A, we show three sequential record instances, i.e., t_0, t_1, and t_2 in our experiments. The different times distributions of mobile devices and cloudlets are illustrated in Figure 4. Figure 4(a) describes the distributions at time t_0. And at time t_1, since the positions of the mobile devices are not changed significantly, there is no corresponding change in the locations of the mobile cloudlets, as we can see in Figure 4(b). But at time t_2, the distribution of mobile devices has changed greatly and we can also find that the two cloudlets move to new locations illustrated in Figure 4(c).

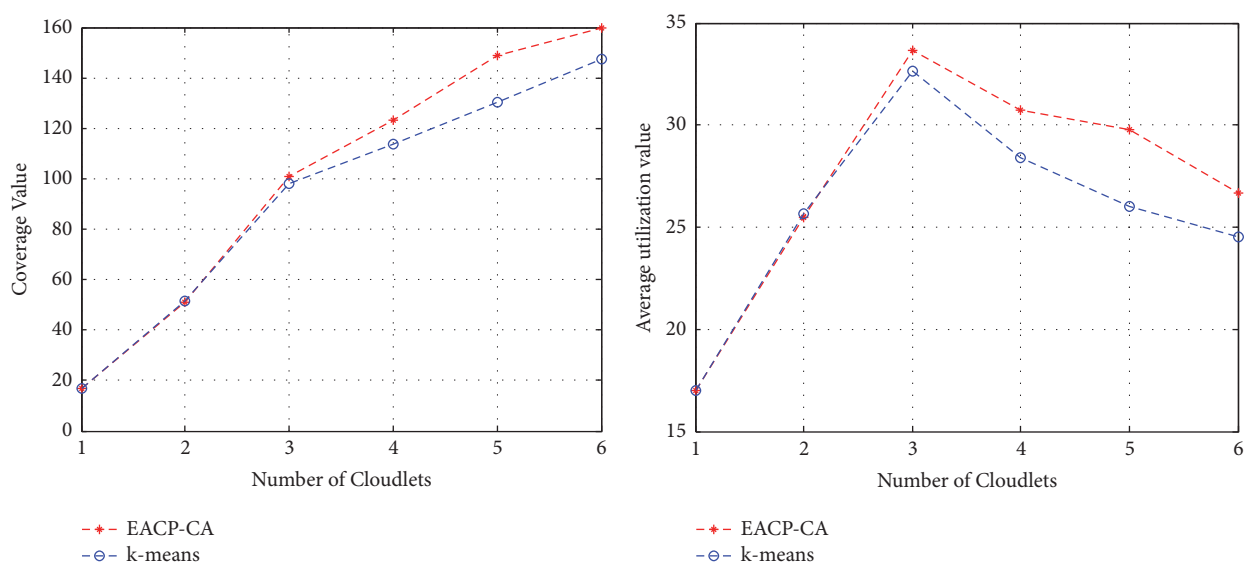
We investigate the performance of our approach and k-means based approach with different cloudlet numbers. We simulate to generate three datasets, which include 200, 300, and 400 mobile devices. The experimental results are shown in Figures 5, 6, and 7 demonstrating that our approach outperforms k-means based approach. Figure 5 presents the results for the dataset with 200 mobile devices. Figure 5(a) shows the coverage value by all cloudlets through executing our EACP-CA approach and k-means based approach, while Figure 5(b) shows the average utilization value of each cloudlet in L. And the experimental results for datasets with 300 mobile devices and 400 mobile devices which are shown in Figures 6 and 7, respectively, are the same as the experimental results of dataset with 200 mobile devices. Figures 6(a) and 7(a) show the coverage value by all cloudlets through executing our EACP-CA approach and k-means based approach for the datasets with 300 mobile devices and 400 mobile devices, respectively, while Figures 6(b) and 7(b) show the average utilization value of each cloudlet in L.

The experimental results illustrated in Figures 5, 6, and 7 show that EACP-CA is better than k-means based approach because both the coverage value and average utilization of each cloudlet within different cloudlet numbers obtained by EACP-CA are higher than k-means based approach. The experimental results show that, with the rising number of cloudlets, the coverage value obtained by EACP-CA and k-means based approach also increases with the fixed number of mobile devices but becomes less significant. This is caused by the fixed number of mobile devices and the increased number of cloudlets, because, through performing the clustering algorithm, the dense mobile devices are covered by



(a) t_0 (b) t_1 (c) t_2

Figure 4: Distributions of mobile devices and cloudlets at t_0, t_1, and t_2.



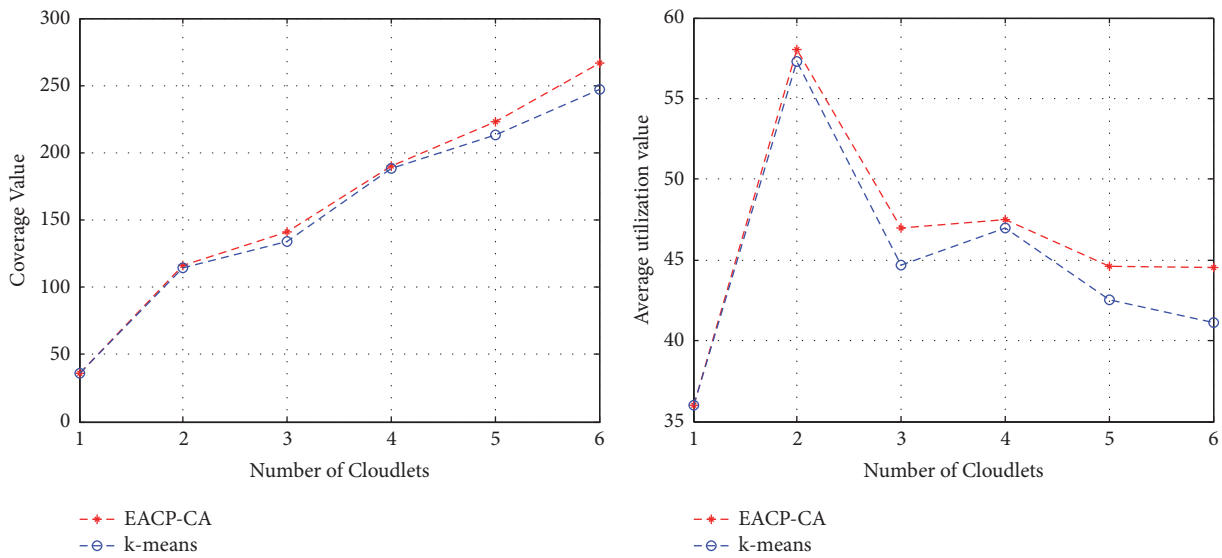
(a) Coverage value comparison (b) Average utilization value comparison

Figure 5: Comparison of performance with EACP-CA and k-means (number of mobile devices $N = 200$).

many cloudlets and the remained discrete mobile devices are also covered by another cloudlet which leads decreasing of the average utilization of each cloudlet. Therefore, we can find that the coverage value is impacted by the number of cloudlets while the number of mobile devices is fixed. With the growing number of cloudlets, the average utilization of each cloudlet is increasing at the beginning and then begins to decrease again. Simultaneously, we can also apply the average utilization value of cloudlets to select a suitable number of cloudlets. We can obtain the appropriate number of cloudlets when the average utilization value of cloudlets and the number of users covered become lager.

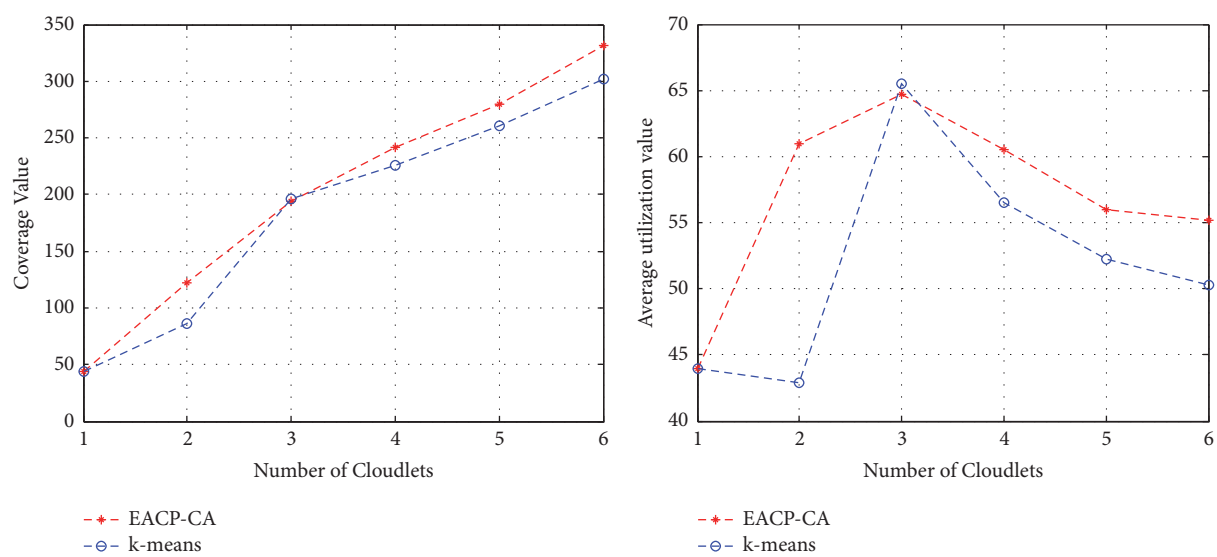
6. Conclusion and Future Work

In this paper, we propose EACP-CA, a novel covering-based approach for enhanced adaptive cloudlets placement. Given a known mobile devices activity area, according to the characteristics of the data and being independent of the initial centers, EACP-CA first employs CA with "blind" features to adaptively partition mobile devices into clusters based on the physical distances between their positions at time t. Then we adjust the central positions obtained by clustering algorithm through the graph path. Next, we execute confirmation of the center positions operation and get the desired



(a) Coverage value comparison (b) Average utilization value comparison

Figure 6: Comparison of performance with EACP-CA and k-means (number of mobile devices $N = 300$).



(a) Coverage value comparison (b) Average utilization value comparison

Figure 7: Comparison of performance with EACP-CA and k-means (number of mobile devices $N = 400$).

cloudlets central positions according to the distribution of the mobile devices. We also need to obtain the pair of the original mobile cloudlet positions and the positions. Finally, we conduct the enhanced adaptive cloudlets placement to finish the mobile cloudlets placement. We parallelize the portioning operation of EACP-CA on Spark to increase the efficiency of EACP-CA in processing big data. The results of the experiments on the simulated datasets demonstrate that EACP-CA significantly outperforms the k-means based approach, which also demonstrate that EACP-CA is more efficient in clustering and improving the utilities of mobile cloudlets.

In our future work, we will enhance EACP-CA to solve big data and consider the assignment of mobile users to the placed cloudlets.

Data Availability

The dataset used to support the findings of this study have been deposited in the repository and can be accessed via the next link: http://bigdata.ahu.edu.cn/paper.

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 National Science Foundation of China (no. 61872002) and the Natural Science Foundation of Anhui Province of China (no. 1808085MF197).

References

[1] P. Asrani, "Mobile cloud computing," *International Journal of Engineering and Advanced Technology*, vol. 2, no. 4, pp. 606–609, 2013.

[2] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: can offloading computation save energy?" *The Computer Journal*, vol. 43, no. 4, Article ID 5445167, pp. 51–56, 2010.

[3] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 725–737, 2017.

[4] M. Chen, Y. Zhang, Y. Li, S. Mao, and V. C. M. Leung, "EMC: emotion-aware mobile cloud computing in 5G," *IEEE Network*, vol. 29, no. 2, pp. 32–38, 2015.

[5] M. Chen, Y. Zhang, L. Hu, T. Taleb, and Z. Sheng, "Cloud-based wireless network: virtualized, reconfigurable, smart wireless network to enable 5G technologies," *Mobile Networks and Applications*, vol. 20, no. 6, pp. 704–712, 2015.

[6] L. Qi, X. Zhang, W. Dou, and Q. Ni, "A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2616–2624, 2017.

[7] H. Xiang, X. Xu, H. Zheng et al., "An adaptive cloudlet placement method for mobile applications over GPS big data," in *Proceedings of the 59th IEEE Global Communications Conference, GLOBECOM 2016*, USA, December 2016.

[8] D. Niyato, P. Wang, P. C. H. Joo, Z. Han, and D. I. Kim, "Optimal energy management policy of a mobile cloudlet with wireless energy charging," in *Proceedings of the 2014 IEEE International Conference on Smart Grid Communications, SmartGridComm 2014*, pp. 728–733, Italy, November 2014.

[9] C. You and K. Huang, "Multiuser resource allocation for mobile-edge computation offloading," in *Proceedings of the 59th IEEE Global Communications Conference, GLOBECOM 2016*, USA, December 2016.

[10] Y. Zhang, D. Niyato, and P. Wang, "Offloading in Mobile Cloudlet Systems with Intermittent Connectivity," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2516–2529, 2015.

[11] E. Ahmed and M. H. Rehmani, "Mobile Edge Computing: Opportunities, solutions, and challenges," *Future Generation Computer Systems*, vol. 70, pp. 59–63, 2017.

[12] Z. Xu, W. Liang, W. Xu, M. Jia, and S. Guo, "Efficient Algorithms for Capacitated Cloudlet Placements," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 10, pp. 2866–2880, 2016.

[13] L. Ma, J. Wu, L. Chen, and Z. Liu, "Fast algorithms for capacitated cloudlet placements," in *Proceedings of the 21st IEEE International Conference on Computer Supported Cooperative Work in Design, CSCWD 2017*, pp. 439–444, New Zealand, April 2017.

[14] H. Yao, C. Bai, M. Xiong, D. Zeng, and Z. Fu, "Heterogeneous cloudlet deployment and user-cloudlet association toward cost effective fog computing," *Concurrency Computation*, vol. 29, no. 16, 2017.

[15] L. Bottou and Y. Bengio, "Convergence properties of the k-means algorithms," in *Proceedings of the Advances in neural information processing systems*, pp. 585–592, 1995.

[16] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," *Wireless Communications and Mobile Computing*, vol. 13, no. 18, pp. 1587–1611, 2013.

[17] 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.

[18] L. Yang, Z. Han, Z. Huang, and J. Ma, "A remotely keyed file encryption scheme under mobile cloud computing," *Journal of Network and Computer Applications*, vol. 106, pp. 90–99, 2018.

[19] J. Li, X. Chen, C. Jia, and W. Lou, "Identity-based encryption with outsourced revocation in cloud computing," *IEEE Transactions on Computers*, 2013.

[20] J. Li, L. Huang, Y. Zhou, S. He, and Z. Ming, "Computation partitioning for mobile cloud computing in a big data environment," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2009–2018, 2017.

[21] K. Gai, M. Qiu, H. Zhao, L. Tao, and Z. Zong, "Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing," *Journal of Network and Computer Applications*, vol. 59, pp. 46–54, 2016.

[22] Z. Cai, H. Yan, P. Li, Z.-A. Huang, and C. Gao, "Towards secure and flexible EHR sharing in mobile health cloud under static assumptions," *Cluster Computing*, vol. 20, no. 3, pp. 2415–2422, 2017.

[23] Y. Zhang, Y. Zhou, X. Guo et al., "Self-Adaptive K-means based on Covering Algorithm," *Complexity*, Article ID 7698274, 2018.

[24] Y.-W. Zhang, Y.-Y. Zhou, F.-T. Wang, Z. Sun, and Q. He, "Service recommendation based on quotient space granularity analysis and covering algorithm on Spark," *Knowledge-Based Systems*, vol. 147, pp. 25–35, 2018.

[25] R. H. Jhaveri, N. M. Patel, Y. Zhong, and A. K. Sangaiah, "Sensitivity Analysis of an Attack-Pattern Discovery Based Trusted Routing Scheme for Mobile Ad-Hoc Networks in Industrial IoT," *IEEE Access*, vol. 6, pp. 20085–20103, 2018.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 2870475, 9 pages
https://doi.org/10.1155/2018/2870475

Research Article

Towards Optimized DFA Attacks on AES under Multibyte Random Fault Model

Ruyan Wang, Xiaohan Meng, Yang Li, and Jian Wang

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, China

Correspondence should be addressed to Yang Li; liyang@uec.ac.jp

Received 10 May 2018; Revised 24 June 2018; Accepted 5 July 2018; Published 13 August 2018

Academic Editor: Xuyun Zhang

Differential Fault Analysis (DFA) is one of the most practical methods to recover the secret keys from real cryptographic devices. In particular, DFA on Advanced Encryption Standard (AES) has been massively researched for many years for both single-byte and multibyte fault model. For AES, the first proposed DFA attack requires 6 pairs of ciphertexts to identify the secret key under multibyte fault model. Until now, the most efficient DFA under multibyte fault model proposed in 2017 can complete most of the attacks within 3 pairs of ciphertexts. However, we note that the attack is not fully optimized since no clear optimization goal was set. In this work, we introduce two optimization goals as the fewest ciphertext pairs and the least computational complexity. For these goals, we manage to figure out the corresponding optimized key recovery strategies, which further increase the efficiency of DFA attacks on AES. A more accurate security assessment of AES can be completed based on our study of DFA attacks on AES. Considering the variations of fault distribution, the improvement to the attack has been analyzed and verified.

1. Introduction

In the age of IoT, IoT technologies can widely perceive our physical world and generate sensing data for further research. There are lots of scenarios in IoT where people have to collaborate through devices to complete tasks; for example, a device sends data to other devices [1], or one user shares EHR in mobile cloud computing [2], and these transmitted data are often the privacy data of users. At the same time, in the big data environment [3, 4], many enterprises need to constantly assimilate big data knowledge and private knowledge by multiple knowledge transfers to maintain their competitive advantage [5]. Thus, the protection of data is especially important during the transmission and encryption of data. However, in recent years, attackers increasingly have access to various cryptographic algorithms. In most cases, attackers develop fault attacks [6] on cryptographic devices and then the private information is leaked. Thus, a lot of sensitive data suffer from severe security and privacy threats.

In general, security and privacy protection are crucial in the field of cloud, fog, or IoT [7–9]. The basis of the security mechanism is the implementation of the cryptosystem. It should be pointed out that the security of cryptosystem includes not only design security but also implementation security. In several ways to assess the implementation security, fault attack is a common method. By studying fault attacks, researchers can evaluate the security of cryptographic algorithms and provide ideas for strengthening protection of sensitive data. This work focuses on the security assessment of AES in fault attacks, which is the most common algorithm in a block cipher system. Among numerous fault attacks, DFA is one of the most practical methods to retrieve the secret key and has become a wide research topic in many fields. Although DFA attacks have been successfully applied to AES, the attack process requires a certain number of faulty ciphertexts or a large key search space. How to reduce the number of faulty ciphertexts required or the search space of keys for attack is a hot research topic.

In this paper, we propose two optimization goals and corresponding strategies. One goal is completing a DFA attack on AES with the fewest ciphertext pairs, and the other is completing a DFA attack on AES with the least computational complexity. The DFA attacks using our strategies can realize the goal of the fewest ciphertext pairs or the least computational complexity, respectively. The optimized DFA attacks in this work take fewer resources and can be completed

faster, achieving higher efficiency. As a result, a more accurate security assessment of AES can be completed based on our work. An earlier version of this paper was presented at the International Conference on Cloud Computing and Security (ICCCS 2018).

The rest of this paper is organized as follows: In Section 2, we introduce the related work proposed by predecessors. Section 3 explains a classical DFA on AES and Liao's method in [10]. Two strategies applied to DFA attacks on AES we propose are introduced in Section 4. The theoretical analysis of our method is given in Section 5 and we conclude in Section 6.

2. Related Work

The concept of DFA was first introduced in [11] in 1996. The principle of DFA is to induce faults (unexpected environmental conditions) into cryptographic implementations to reveal their internal states. In 2003, Gilles Piret and JeanJacques Quisquater described a DFA attack technique [12] and could break the AES-128 with only 2 faulty ciphertexts, assuming the fault is in MixColumns operation of the eighth or ninth round. In 2004, Christophe Giraud proposed two different DFA attacks on AES [13]. The first one induces a fault to only one bit of an intermediate result and the key can be obtained with 50 faulty ciphertexts for AES-128. While the second one induces a fault to a whole byte and less than 250 faulty ciphertexts are needed for key recovery for AES-128. In [14] in 2011, Tunstall, Mukhopadhyay, and Ali proposed a two-stage algorithm of DFA that could recover the AES 128-bit key using one fault injection. However, without plaintext-ciphertext exhaustive search, the most efficient DFA key recovery on AES-128 with a single-byte fault requires 2 pairs of ciphertexts [15]. In terms of DFA attacks on AES with a random multibyte fault, the existing literature shows that 6 pairs of ciphertexts are required to develop the attack [16]. In particular, in extreme cases that the injected faults are four-byte ones, attackers need 1500 pairs of ciphertexts for key recovery.

In 2017, Nan Liao et al. [10] proposed an improved DFA attack method on AES with unknown and random faults. They focused on multibyte faults whose locations and values are unknown to the attackers. The fault model in their work combined the single-byte fault model and multibyte fault model and took both accuracy and efficiency into considerations. Their experimental results showed that most of the attacks could be completed within 3 pairs of ciphertexts. After that, a hybrid model was proposed in [17] to improve availability of ciphertext for DFA against AES and 6 pairs of correct and faulty ciphertexts could recover the secret key of AES-128. In [17], the attack models available for analysis include single-byte random faults in encryption process, multibyte random faults in encryption process, and single-byte faults in key schedule. In addition, one improved DFA attack using all-fault ciphertexts on AES was proposed in [18]. The all-fault ciphertexts were used to optimize the selection of the brute-force space, helping to recover the secret key quickly and improve the analysis efficiency. Their experiment result demonstrated that the time consumed on the brute-force attack could be reduced 60.81% on average.

3. DFA on AES

3.1. Generic Fault Model. Two kinds of fault models are widely used in most of DFA attacks on AES, which are single-byte fault model and multibyte fault model. In this paper, multibyte fault model assumes that the size of the injected fault ranges from one byte to three bytes in one column of AES state. The four-byte faults are not discussed in this work since they are not as useful as others in the key recovery, also they can be omitted in practical fault injections. When some techniques like laser beam [19] are used to induce faults, the fault can be fixed to single byte and the specific location of the fault can even be selected. However, when other techniques are used, such as supply voltage variation [20] and clock glitch injection [21], the size of the fault may be more than one byte, and attackers cannot control the location. It should be noted that though the fault injection techniques like laser beam enable attackers to control the characteristics of the fault, they are sophisticated and high-cost. On the contrary, fault injection techniques such as supply voltage variation and clock glitch injection are noninvasive and they need lower cost, which are more practical.

Therefore, this research focuses on the more general fault model, which is multibyte fault model since the methods to induce multibyte faults are more practical. In addition, it is necessary to introduce one kind of fault model that combines the advantages of the two fault models.

3.2. Basic Key Recovery in DFA on AES. For AES that consists of 10 round functions, DFA attacks usually target the last two rounds. When the fault is injected to the last two rounds, the fault only affects some bytes of the ciphertext. Therefore, it is feasible to retrieve the key by analyzing the differential value of the corresponding parts of the correct and faulty ciphertext.

Assume that a single-byte fault is injected to the first column of the state after `ShiftRows` operation of R_9. After R_9, the fault spreads to the entire column. After R_{10}, which omits `MixColumns` operation, the fault affects specific four bytes of the output. In other words, only four bytes of the ciphertext will be affected by the injected fault. Also, the locations of four bytes are determined by the location of the initial fault. Attackers can make assumptions about the four-byte round-key of R_{10} in affected locations and verify whether the fault information derived is consistent with the fault model.

The specific key recovery process is as follows: assuming key values and calculating (1)-(3), attackers can achieve two internal states after `ShiftRows` operation of R_9, respectively, from the ciphertext pair. Calculating their difference and comparing the information with the fault model, those incorrect key assumptions are eliminated. In (1)-(3), δ is the difference of the correct and faulty ciphertexts; K_9, K_{10} denote the round-keys of R_9, R_{10}; S_9, S_{10} denote the internal states after `AddRoundKeys` operation of R_9, R_{10}; $\tilde{S}_9$, $\tilde{S}_{10}$ denote the corresponding states in faulty encryptions. *InvMC*, *InvSB,* and *InvSR* are the inverse processes for `MixColumns`, `SubBytes`, and `ShiftRows` operations. We can have (1) based on the characteristics of the encryption functions of

Table 1: Some notations and their meanings.

Notations	Meaning
$P_{candidates}$	The proportion of the number of candidate keys to the number of all possible keys
N_{can_key}	The number of candidate keys after the first analysis in DFA process
P_{can_1b}	$P_{candidates}$ for the single-byte fault model
P_{can_2b}	$P_{candidates}$ for the two-byte fault model
P_{can_3b}	$P_{candidates}$ for the three-byte fault model

AES. Since `MixColumns` is not included in R_{10}, we can get S_9 and $\tilde{S}_9$ as shown in (2)-(3).

$$\begin{aligned}\delta &= InvMC\left(K_9 \oplus S_9\right) \oplus InvMC\left(K_9 \oplus \tilde{S_9}\right)\\ &= InvMC\left(S_9 \oplus \tilde{S_9}\right),\end{aligned} \tag{1}$$

$$S_9 = InvSB\left(InvSR\left(K_{10} \oplus S_{10}\right)\right), \tag{2}$$

$$\tilde{S_9} = InvSB\left(InvSR\left(K_{10} \oplus \tilde{S_{10}}\right)\right). \tag{3}$$

If the injected fault is multibyte, the circumstance is almost the same. Though the outputs of two fault diffusion processes are identical, the numbers of ciphertext pairs required for key recovery under two kinds of models differ. In the case of single-byte faults, 2 pairs of ciphertexts are enough to retrieve four bytes of the round-key [15]. However, in the case of multibyte faults, 6 pairs of ciphertexts are required [16].

3.2.1. DFA Method Proposed by Nan Liao et al. In 2017, Nan Liao et al. proposed improved DFA attacks on AES with multibyte faults [10]. Since our method is based on their contributions, their method is introduced first. They classified faults into four types according to the number of faulty bytes. In their attack, four-byte faults are not under discussion since four-byte faults hardly appear in real attacks. The occurrence rate of the fault type is denoted as P_t, t denotes the number of faulty bytes, and $t \in [1, 3]$. The notations used are provided in Table 1. $P_{candidates}$ denotes the proportion of the number of candidate keys to the number of all possible keys, which is approximate to the proportion of the number of covered faults to the number of all possible faults. N_{can_key} is defined as multiplying $P_{candidates}$ and the number of all possible keys N_{all}, which is shown in

$$N_{can\_key} = N_{all} \times P_{candidates}. \tag{4}$$

The amount of all possible round-keys is always 2^{32} for the first analysis in DFA process, corresponding to analyzing the first ciphertext pair in DFA process. Every time an analysis in DFA process is completed, the amount of candidate keys is decreased.

For the single-byte fault model, the proportion of the number of covered faults to the number of all possible faults is

$$\frac{C_4^1 \times 255^1}{2^{32} - 1} = 2.37 \times 10^{-7}. \tag{5}$$

As mentioned above, P_{can_1b} is defined as

$$P_{can\_1b} \approx 2.37 \times 10^{-7}. \tag{6}$$

Similarly, P_{can_2b} and P_{can_3b} are defined as follows:

$$P_{can\_2b} \approx \frac{C_4^2 \times 255^2}{2^{32} - 1} = 9.08 \times 10^{-5}, \tag{7}$$

$$P_{can\_3b} \approx \frac{C_4^3 \times 255^3}{2^{32} - 1} = 0.0154. \tag{8}$$

P_{can_1b} is so small that only 2 pairs of ciphertexts are enough to retrieve the round-key. Similarly, 3 pairs of ciphertexts are required for two-byte faults and 6 pairs of ciphertexts are required for three-byte faults. The theoretical candidate numbers in three fault models after each analysis are, respectively, given in Table 2.

It is claimed in [10] that $P_{candidates}$ decides the number of ciphertext pairs required in the DFA attack. If small $P_{candidates}$ like P_{can_1b} is used in the attack, the number of ciphertext pairs required will be greatly reduced and the efficiency of the attack will be increased.

It can be found that if the fault type is known to attackers, they are able to use the consistent fault model to complete the DFA attack, leading to fewer ciphertexts required. Especially when single-byte faults occur frequently, fewer ciphertext pairs are needed. However, in the practical environment, attackers have no idea about the fault type. They can only assume the fault type and verify the correctness.

Similar to the analysis process under multibyte model in [16], Nan Liao et al. considered three fault types without four-byte faults. The biggest difference between their methods is that Liao's method divided faults into three types in each analysis and calculated, respectively. Nan Liao et al. refined the object of each analysis and obtained more detailed information after each analysis. They found that many fault type series could be analyzed with 2-5 pairs of ciphertexts. Thus, it is suggested in Liao's method to give priority to assuming and verifying the fault type series that need less ciphertext pairs. Only one candidate left after DFA process means the remaining candidate is the correct round-key. No candidate left means the assumed fault type series is wrong.

Figure 1 shows all possibilities of fault type series that need less than 6 pairs of ciphertexts. The line connecting two oval frames is defined as a path since it represents one possible situation of fault type series. All figures in the frames represent the number of key candidates after the last analyses including three fault situations. Take 1024615 in the oval frame in the left column for example. After the second analysis under three-byte fault model, the number of key candidates is $66142496 \times P_{can\_3b} + 389983 \times P_{can\_2b} + 1020 \times P_{can\_1b} \approx 1024615$. Thus, the figure in the oval frame is the sum of the results under three situations. For the dotted line path, the number of key candidates after analysis is more than 1, which means analysis needs to be continued to recover the key. For the red solid line path, the number of key candidates after analysis is close to 0. For example, the rightmost red path represents $1020 \times P_{can\_1b} = 0.00024$. If the path is consistent with the real fault type series at this point, then only the

Table 2: The theoretical candidate numbers after each analysis in three fault models.

Analyses completed	1	2	3	4	5	6
Single-byte fault model	1020	0.00024	0	0	0	0
Two-byte fault model	389983	35	0.003	0	0	0
Three-byte fault model	66142496	1018594	15686	241	3.72	0.06

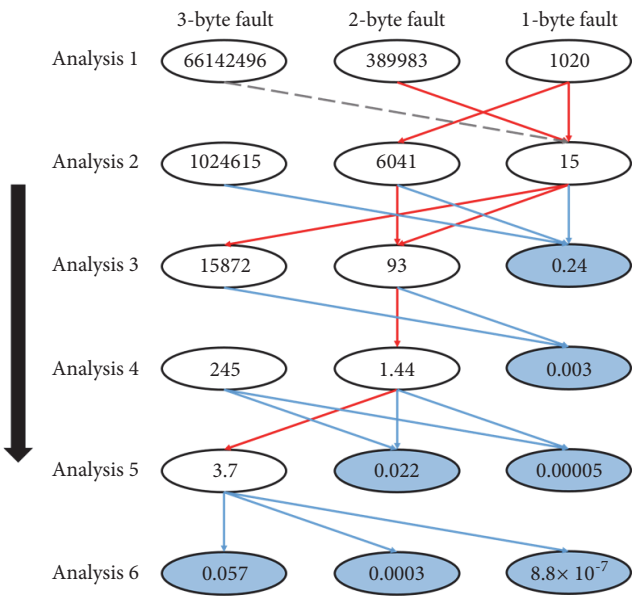


Figure 1: The possible situations that can identify the key within 6 pairs of ciphertexts [10].

correct round-key survives. For the blue oval frames, the figures in these frames are already close to 0. In Liao's method, they defined the paths that can retrieve the round-key within 5 pairs of ciphertexts as exploitable paths and give priority to assuming and verifying exploitable paths. When the real fault type series are consistent with these paths, the remaining key candidate is the correct round-key and the attack can be completed quickly.

Nan Liao et al. verified the effectiveness of their method by experiments. They collected 12000 ciphertexts and faults are injected in the same column in the same round for these ciphertexts. The 12000 ciphertexts are divided into 2000 groups and each group consists of 6 ciphertexts. If multibyte fault model is used, 6 ciphertext pairs in each group will all be exploited to recover the round-key. If Liao's method is used, there is a big probability that less than 6 ciphertext pairs are enough for the round-key recovery.

4. Proposed DFA Method and Application on AES

4.1. Inspirations from [10]. In [10], the attack successfully recovers the correct round-key with 2-5 pairs of ciphertexts in most cases. However, Liao's method is not perfectly reasonable, especially the method of choosing exploitable paths. It should be noted that, in Figure 1, the number in the oval frame is the cumulative sum of the previous analysis results of three fault situations. Thus, it is incomplete to determine exploitable paths through the number in the oval frame. Some paths will be missed if the cumulative sum is considered barely. For instance, fault type series "223⋯" only needs 3 pairs of ciphertexts to be verified, but it is not included in exploitable paths in Liao's method. The reason for missing paths is that they add up the number of key candidates from three situations and make an analysis on the sum, leading to neglect of the number of key candidates after each analysis for a single path. As a result, some paths that can retrieve the round-key within 5 pairs of ciphertexts are eliminated and more calculations are required. In order to avoid such omissions, we further refine the analysis process and discuss one fault type at a time. That is to say, one path is taken as the unit of analysis instead of three fault types being discussed at the same time.

In addition, the goal in [10] is not clear enough. The authors stressed that their method could retrieve the round-key with fewer ciphertext pairs and the least computation. They mixed required ciphertext pairs and computation together for discussion, which made the goal ambiguous.

Generally, we find that the goal in [10] is not clear and their strategy is optimizable. We first set the optimization goal and then develop the method to find the optimized key recovery strategy. Our improvement will be introduced in the following section. Note that although this work mainly focuses on AES, our work can be easily adapted to DFA attacks on other ciphertexts.

4.2. Improved DFA Attack on AES under Multibyte Random Fault Model. The following content is our improved DFA attack on AES, showing great advantages compared with the previous DFA attacks.

We denote m_i as the amount of key candidates after i^{th} analysis in DFA process. P_{can_tb} is the proportion of the key candidates after one analysis in DFA process under t-byte fault model. The theoretical m_i value is calculated as

$$m_i = m_{i-1} \times P_{can\_tb}, \quad i \in [1,6], \; t \in [1,3]. \tag{9}$$

The amount of key candidates decreases after each analysis. When $i = 0$, $m_0 = (2^8)^4 = 2^{32}$, which means the initial amount of key candidates before the first analysis in DFA process is always 2^{32}. When $i = 6$, m_6 must be zero based on the fact that the key can be determined with 6 pairs of ciphertexts under multibyte fault model.

The procedure of our attack method is as follows:

(1) Obtain the correct ciphertext and several faulty ciphertexts.

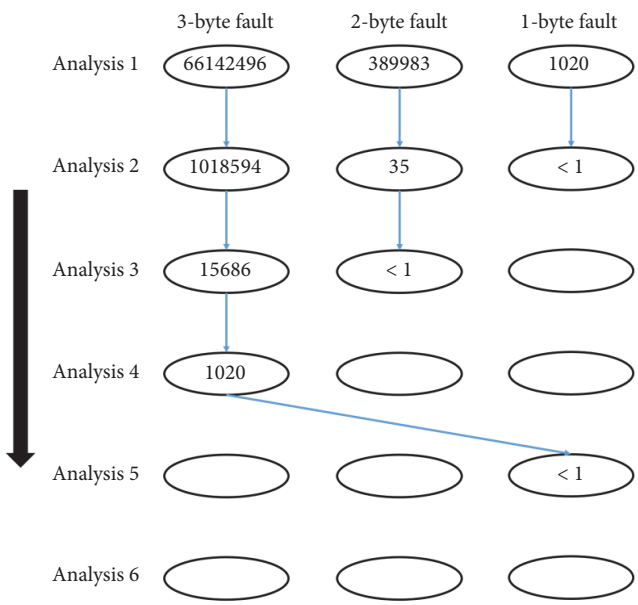


Figure 2: The DFA process of some paths.

(2) Choose one path assumption, analyze the ciphertext pair, and verify the assumption.

(3) After i^{th} analysis, if $m_i = 1$, then the key candidate is the correct round-key.

(4) After i^{th} analysis, if $m_i = 0$, then repeat (2) until the correct round-key is recovered.

According to three fault types and the maximum ciphertext pairs described earlier, there are 3^6 possible paths for AES totally. We review each path and calculate the amount of key candidates after each analysis in DFA process to determine the fewest ciphertext pairs required. A figure for better understanding is shown as Figure 2. It shows all intermediate results of analyses in DFA process for three paths: "33331⋯"; "22⋯"; and "11⋯." The number in the oval frame stands for the amount of remaining key candidates after the last analysis for the current path. Different from Figure 1, the number in the oval frame is related to only one path, which is the current path being verified. In our method, it is intuitive and accurate to see the fewest ciphertext pairs each path required.

When guessing and verifying the paths, different assuming orders lead to different computational complexity and different numbers of ciphertext pairs required. Given a fault model and a specific goal, it is possible to determine an optimized strategy for 3^6 paths. In this paper, we consider two specific goals to be optimized, which are recovering the round-key with the fewest ciphertext pairs and the least computational complexity.

Key Recovery with Fewest Ciphertext Pairs. If the goal is to recover the round-key with the fewest ciphertext pairs, the strategy of assuming paths is as follows. First, list all paths that require 2, 3, 4, 5, and 6 pairs of ciphertexts, respectively. Afterwards, start from assuming and verifying the paths that require 2 ciphertext pairs; if only one key candidate is left after DFA process, this candidate is viewed as the correct round-key. Otherwise, we keep verifying paths until we retrieve the round-key, with the order of **2 ciphertexts ⟶ 3 ciphertexts ⟶ 4 ciphertexts ⟶ 5 ciphertexts ⟶ 6 ciphertexts**. When there is more than one path that can be analyzed with the same number of ciphertext pairs, we preferentially verify the path whose occurrence rate is higher. In other words, the first priority is less ciphertext pairs required, and the second priority is higher occurrence rate. Finally, when round-key is recovered, the ciphertext pairs used are consistent with the actual ciphertext pairs the path requires. The process of sorting all 3^6 paths is shown in Algorithm 1. After Algorithm 1, φ is a set of 3^6 paths in an order of less ciphertext pairs required to more ciphertext pairs required.

For the first strategy, the overview of the DFA attack is shown in Figure 3.

Key Recovery with Least Computational Complexity. If the goal is to recover the round-key with the least computational complexity, we need to take P_1, P_2, and P_3 into consideration and reaffirm the order of assuming paths. For the purpose of reducing computational complexity, one needs to eliminate duplicated calculations by saving the internal values and key candidates that survive the analysis. When later analysis requires the same data, the information saved can be directly exploited without recalculations. Furthermore, we should consider the occurrence rate of each path to determine the order of assuming paths. However, it is not sufficient to make decisions based on the occurrence rates of the paths. The calculations cost for different fault types in each analysis are also an important part. Hence, we introduce the concept of cost-efficiency, which is the determinant for each analysis. We define ce as follows:

$$ce = \frac{P_t}{P_{can\_tb}}, \tag{10}$$

where t is the type of fault. It contains two variables that are the occurrence rate of some fault type and $P_{candidates}$ corresponding to that fault type. When the occurrence rate of some fault type is higher, this fault type appears more in actual situations. That means it is more likely to find the correct path quickly if we first assume and verify this fault type. Thus, the higher the occurrence rate of the fault type is, the higher the priority should be. When $P_{candidates}$ for some fault type is smaller, it means the analysis corresponding to the fault type can pick out fewer key candidates, leading to less computation. Thus, the smaller $P_{candidates}$ for the fault type is, the higher the priority should be. Therefore, we consider two variables in total that are closely related to the second goal and it is reasonable to determine the order of assuming paths relying on the variable ce. Before the first analysis in DFA process, we calculate ce for all fault types and determine the order of assuming fault types based on ce. The greater ce is, the more the fault type is worth being verified preferentially.

The specific approach is as follows. Before the first analysis in DFA process, ce for three fault types are calculated and the order of assuming fault types is determined. In each analysis, we first verify the fault type whose ce is the greatest and next the second great and last the left. When one analysis in DFA process is finished and the amount of key candidates is far greater than 1, we continue assuming and verifying fault

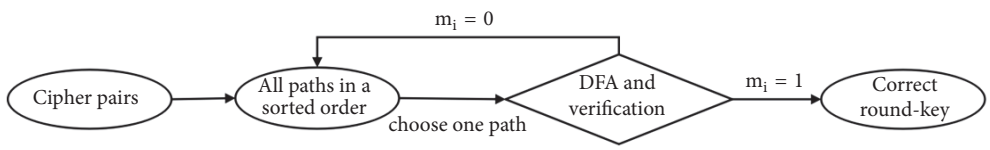


Figure 3: The attack method.

Input: φ: 3^6 paths of disorder.
Output: φ: ordered 3^6 paths.
(1) **for** each $j \in [2, 6]$ **do**
(2) initialize $\psi_j = \Phi$
(3) **end for**
(4) **for** each $p \in \varphi$ **do**
(5) $j \longleftarrow$ the number of ciphertext pairs p requires
(6) $\psi_j = p \cup \psi_j$
(7) **end for**
(8) **for** each $j \in [2, 6]$ **do**
(9) sort paths in ψ_j in order of occurrence rate from low to high
(10) **end for**
(11) **return** $\varphi \longleftarrow \psi_2 \parallel \psi_3 \parallel \psi_4 \parallel \psi_5 \parallel \psi_6$

Algorithm 1: The process of sorting all 3^6 paths.

type. After several analyses, if the amount of key candidates is 1, the current path matches the real situation and the left candidate is the round-key. If the amount of key candidates is 0, it means the current path assumed is wrong. At this point, we have to turn back to the previous analysis and assume the next fault type. Similar to the first strategy, the analysis above can be viewed as assuming ordered paths from the macroscopic angle. However, the order of assuming paths here is connected with *ce* for three fault types. For better understanding, a detailed example of the strategy is given. Suppose that the situation of three fault types in practical environment is $P_1 > P_2 > P_3$, and the fault type series is "131221." First, *ce* for three fault types are calculated in advance. The values of *ce* for three fault types are denoted as ce_t, and t denotes the fault type.

$$ce_1 = \frac{P_1}{2.37 \times 10^{-7}},$$
$$ce_2 = \frac{P_2}{9.08 \times 10^{-5}}, \tag{11}$$
$$ce_3 = \frac{P_3}{0.0154}.$$

It is known that $P_1 > P_2 > P_3$, so clearly $ce_1 > ce_2 > ce_3$. According to the second strategy, we first assume single-byte faults and then two-byte faults and three-byte faults at last. After the first analysis, the amount of key candidates decreases from 2^{32} to 1020. Since the amount is not 1, we continue assuming that the fault is single-byte. After the second analysis, the amount of key candidates is 0, which means the current path "11$\cdots$" is wrong. So we turn back to the second analysis and assume that the fault is two-byte one. There are 0 key candidates after the second analysis, which means the current path "12$\cdots$" is also wrong. Similarly, we turn back and assume that the fault is three-byte. This time, 15 key candidates are left after the second analysis. We continue assuming that the fault is single-byte. After the third analysis, only one key candidate survives. Obviously, the key candidate left is the round-key and the fault type series is "131$\cdots$." The computation in the analyses above is significantly less than that in 6 analyses under multibyte fault model.

5. Results and Discussion

This section analyzes the two strategies proposed and they are more efficient than the previous attacks. In the following, the first strategy is called data-complexity priority strategy and the second strategy is called computation-complexity priority strategy.

5.1. Data-Complexity Priority Strategy. As mentioned earlier, 6 pairs of ciphertexts are required in DFA attacks under multibyte fault model. However, in our method using data-complexity priority strategy, most of the DFA attacks can be completed within 5 pairs of ciphertexts.

Here the comparison of the expected amount of used ciphertext pairs between our method and Liao's method is also given. For all 3^6 paths, we denote the expected amount of used ciphertext pairs as *E(r)*. It is the accumulation of the product of the occurrence rate and used faulty ciphertext pairs for all paths. In the case of known fault type,

$$E(r) = \sum P_{path} \times r, \tag{12}$$

in which r denotes the actual ciphertext pairs that *path* requires ($2 \le r \le 6$), and P_{path} denotes the occurrence rate of

path. Take path "11⋯" as an example; its occurrence rate is $P_{11\ldots} = P_1 \times P_1$.

In Liao's method, the number of used ciphertext pairs is more than the actual required ciphertext pairs for *path*. Take fault type series "331⋯" as an example; it only needs 3 pairs of ciphertexts to be verified. But, in Liao's method, it will not be verified until all exploitable paths have been verified since it is not included in exploitable paths. In this example, they will use 6 pairs of ciphertexts to complete the attack. So the expected amount of used ciphertext pairs in Liao's method is greater than $E(r)$.

In our method using data-complexity priority strategy, the number of used ciphertext pairs always equals to the actual required ciphertext pairs for *path*. That is to say, the expected amount of used ciphertext pairs in our method equals to $E(r)$. According to the expected amount of used ciphertext pairs, we can clearly see that our method using data-complexity priority strategy is better than Liao's method.

In [10], authors give the probability of attacks that succeed within 3 pairs of ciphertexts. In Liao's method, the fault type series that can be successfully verified with 2 pairs of ciphertexts are "11⋯", "12⋯", "21⋯"; the fault type series that can be successfully verified with 3 pairs of ciphertexts are "311⋯", "312⋯", "313⋯", "321⋯", "331⋯", "221⋯", "222⋯", "231⋯". Hence, the probability of attacks that succeed within 3 pairs of ciphertexts are denoted as follows.

(1) The probability of attacks that succeed with 2 pairs of ciphertexts is

$$P_{2ciphertexts} = P_1^2 + 2 \bullet P_1 \bullet P_2. \tag{13}$$

(2) The probability of attacks that succeed with 3 pairs of ciphertexts is

$$\begin{aligned} P_{3ciphertexts} = {} & P_1 \bullet P_3 \bullet (P_3 + 1) + P_2^2 \bullet (P_1 + P_2) \\ & + 2 \bullet P_3 \bullet P_2 \bullet P_1. \end{aligned} \tag{14}$$

(3) Thus the probability of attacks that succeed within 3 pairs of ciphertexts is

$$P_{2ciphertexts+3ciphertexts} = P_{2ciphertexts} + P_{3ciphertexts}. \tag{15}$$

In our method using data-complexity priority strategy, the fault type series that can be successfully verified with 2 pairs of ciphertexts are the same as those in Liao's method. Besides those in Liao's method, we can verify more fault type series within 3 pairs of ciphertexts: "131⋯"; "132⋯"; "133⋯"; "223⋯"; "232⋯"; "322⋯." Hence, the probability of attacks that succeed within 3 pairs of ciphertexts in our method is

$$\begin{aligned} P_{3ciphertexts} = {} & P_1 \bullet P_3 \bullet (2 + P_3 + 2 \bullet P_2) \\ & + P_2^2 \bullet (1 + 2 \bullet P_3). \end{aligned} \tag{16}$$

Suppose that the probabilities of occurrence are $P_1 = 0.89$, $P_2 = 0.1$, and $P_3 = 0.01$, respectively. In Liao's method, $P_{2ciphertexts+3ciphertexts} = 99.15\%$. In our method using data-complexity priority strategy, $P_{2ciphertexts+3ciphertexts} = 99.997\%$. In this case, it can be seen that the probability of attacks using our strategy that succeed within 3 pairs of ciphertexts is slightly greater than theirs: 99.997% > 99.15%. Then suppose that $P_1 = 0.4$, $P_2 = 0.3$, and $P_3 = 0.3$, respectively. In Liao's method, the probability of attacks that succeed within 3 pairs of ciphertexts is 72.7%. In our method using data-complexity priority strategy, the probability of attacks that succeed within 3 pairs of ciphertexts is 92.8%. The probability of attacks that succeed within 3 pairs of ciphertexts using our strategy is greater than theirs by 20.1%. Generally, it is obvious that our data-complexity priority strategy is better than Liao's method.

5.2. Computation-Complexity Priority Strategy. The following gives the analysis of the computational complexity in DFA attacks using computation-complexity priority strategy.

In this work, for each possible key value, we calculate the same basic key recovery in DFA (see (1)-(3)), so the computation in one analysis in DFA process is fixed. Therefore, the computation in a DFA attack actually is equivalent to the number of times of calculating (1)-(3), which is the sum of the amount of key candidates in each analysis. Hence, we can use the cumulative sum of the amount of key candidates per step to represent the computation.

For the multibyte fault model in [16], the times of calculating (1)-(3) are

$$\begin{aligned} & 2^{32} + 66571993 + 1031865 + 15993 + 247 + 3.84 \\ & \quad \approx 2^{32} + 2^{26.01}. \end{aligned} \tag{17}$$

So the computation in DFA attacks under multibyte fault model is fixed, which can be represented as $2^{32}+2^{26.01}$ for comparison with computation in other DFA attacks.

For computation-complexity priority strategy, the computation in a DFA attack is closely related to the occurrence rates of three fault types. In order to compare them in a more intuitive way, we give 3 possible situations about the occurrence rates of three fault types. In each situation, we calculate the product of the occurrence rate of each path and the total computation used for the path, and then we add up the product values of all paths. The corresponding average computation is shown in Table 3.

As we can see in Table 3, the average computation for three situations of three fault types is all less than the fixed value for the general multibyte fault model. It can be found from the table that, in the case of P_1 being much greater than P_2 and P_3, the computation in DFA attacks is the least and the computation decreases a lot compared to the fixed value. At this time, DFA attacks are the most successful. In practical attack scenarios, this strategy can be exploited when single-byte faults appear frequently. The computation-complexity priority strategy can help attackers to reduce the computation in DFA attacks to a certain extent.

6. Conclusion

This work contributes to a more accurate security evaluation for the DFA attack under the multibyte random fault model. Previous work proposed the idea to take advantages of

Table 3: Average computation for different situations of three fault types.

Occurrence rate	P_1	P_2	P_3	P_1	P_2	P_3	P_1	P_2	P_3
	0.7	0.2	0.1	0.2	0.7	0.1	0.1	0.2	0.7
Average computation	$2^{32} + 2^{22.68}$			$2^{32} + 2^{22.72}$			$2^{32} + 2^{25.48}$		

exploitable faults for better key recovery efficiency. To define better efficiency of DFA attack, this work introduces two optimization goals as the fewest ciphertext pairs and the least computational complexity. With different optimization order of these two goals, we propose the corresponding optimized key recovery strategies. Using AES as the analysis target, the improvement compared with previous work has been verified theoretically and with examples. We focus on the security assessment of AES in fault attacks and thus provide reference for the protection of private data. In future work, researchers may apply the approach of optimization to DFA attacks on other ciphers and investigate the attack process and efficiency in detail. In addition, the computation-complexity priority strategy may be further optimized to achieve higher efficiency of the attack.

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 research was supported by National Natural Science Foundation of China (no. 61602239) and Jiangsu Province Natural Science Foundation (no. BK20160808).

References

[1] C. Wang, J. Shen, Q. Liu, Y. Ren, and T. Li, "A Novel Security Scheme Based on Instant Encrypted Transmission for Internet of Things," *Security and Communication Networks*, vol. 2018, Article ID 3680851, 7 pages, 2018.

[2] Z. Cai, H. Yan, P. Li, Z.-A. Huang, and C. Gao, "Towards secure and flexible EHR sharing in mobile health cloud under static assumptions," *Cluster Computing*, vol. 20, no. 3, pp. 2415–2422, 2017.

[3] Y. Wen, J. Liu, W. Dou, X. Xu, B. Cao, and J. Chen, "Scheduling workflows with privacy protection constraints for big data applications on cloud," *Future Generation Computer Systems*, 2018.

[4] X. Xu, X. Zhao, F. Ruan et al., "Data Placement for Privacy-Aware Applications over Big Data in Hybrid Clouds," *Security and Communication Networks*, vol. 2017, Article ID 2376484, 15 pages, 2017.

[5] C. Wu, "Time optimization of multiple knowledge transfers in the big data environment," *Computers, Materials and Continua*, vol. 03, 2018.

[6] B. Dan, R. A. Demillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *International Conference on Theory and Application of Cryptographic Techniques*, pp. 37–51, 1997.

[7] Y. Cao, Z. Zhou, X. Sun, and C. Gao, "Coverless information hiding based on the molecular structure images of material," *Computers, Materials and Continua*, vol. 54, no. 2, pp. 197–207, 2018.

[8] 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.

[9] 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.

[10] N. Liao, X. Cui, K. Liao, T. Wang, D. Yu, and X. Cui, "Improving DFA attacks on AES with unknown and random faults," *Science China Information Sciences*, vol. 60, no. 4, 2017.

[11] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Proceedings of the International Cryptology Conference*, pp. 513–525, 1997.

[12] G. Piret and J. J. Quisquater, "A differential fault attack technique against spn structures, with application to the aes and khazad," in *Proceedings of the Cryptographic Hardware and Embedded Systems - CHES 2003, International Workshop*, pp. 77–88, Cologne, Germany, 2003.

[13] C. Giraud, "Dfa on aes," Cryptology ePrint Archive 2003/008, 2003, http://eprint.iacr.org/.

[14] M. Tunstall, D. Mukhopadhyay, and S. Ali, "Differential fault analysis of the advanced encryption standard using a single fault," in *Proceedings of the WISTP*, pp. 224–233, 2011.

[15] D. Mukhopadhyay, "An improved fault based attack of the advanced encryption standard," in *Proceedings of the AFRICACRYPT*, pp. 421–434, 2009.

[16] A. Moradi, M. T. M. Shalmani, and M. Salmasizadeh, "A generalized method of differential fault attack against aes cryptosystem," in *Proceedings of the CHES*, pp. 91–100, 2006.

[17] Y. Liu, X. Cui, J. Cao, and X. Zhang, "A hybrid fault model for differential fault attack on AES," in *Proceedings of the 2017 IEEE 12th International Conference on ASIC (ASICON)*, pp. 784–787, Guiyang, October 2017.

[18] Y. Ni, X. Cui, T. Wang et al., "Improving DFA on AES using all-fault ciphertexts," in *Proceedings of the 12th IEEE International Conference on Advanced Semiconductor Integrated Circuits, ASICON 2017*, pp. 283–286, October 2017.

[19] M. Agoyan, J. Dutertre, A. Mirbaha, D. Naccache, A. Ribotta, and A. Tria, "Single-bit DFA using multiple-byte laser fault injection," in *Proceedings of the 2010 IEEE International Conference on Technologies for Homeland Security (HST 2010)*, pp. 113–119, November 2010.

[20] A. Barenghi, G. M. Bertoni, L. Breveglieri, M. Pellicioli, and G. Pelosi, "Low voltage fault attacks to AES," in *Proceedings of the 2010 IEEE International Symposium on Hardware-Oriented*

Security and Trust (HOST), pp. 7–12, Anaheim, CA, USA, June 2010.

[21] N. Selmane, S. Guilley, and J.-L. Danger, "Practical setup time violation attacks on AES," in *Proceedings of the 7th European Dependable Computing Conference, EDCC-7*, pp. 91–98, May 2008.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 9856537, 8 pages
https://doi.org/10.1155/2018/9856537

Research Article

A Security Sandbox Approach of Android Based on Hook Mechanism

Xin Jiang,[1] Mingzhe Liu,[1] Kun Yang,[1] Yanhua Liu,[1] and Ruili Wang[2]

[1]*State Key Laboratory of Geohazard Prevention and Geoenvironment Protection, Chengdu University of Technology, Sichuan, China*
[2]*Institute of Natural and Mathematical Sciences, Massey University, Auckland, New Zealand*

Correspondence should be addressed to Mingzhe Liu; liumz@cdut.edu.cn

Received 13 May 2018; Accepted 28 June 2018; Published 19 July 2018

Academic Editor: Xuyun Zhang

As the most widely applied mobile operating system for smartphones, Android is challenged by fast growing security problems, which are caused by malicious applications. Behaviors of malicious applications have become more and more inconspicuous, which largely increase the difficulties of security detection. This paper provides a new security sandbox approach of Android based on hook mechanism, to further enrich Android malware detection technologies. This new sandbox monitors the behaviors of target application by using a process hook-based dynamic tracking method during its running period. Compared to existing techniques, (1) this approach can create a virtual space where apk can be installed, run, and uninstalled, and it is isolated from the outside and (2) a risk assessment approach based on behavior analysis is given so that users can obtain an explicit risk prognosis for an application to improve their safety. Tests on malware and normal application samples verify this new security sandbox.

1. Introduction

The security of Android is highly valued by and of wide concern to the industry [1] since a huge number of mobile applications are developed and used based on Android system. Various security issues of Android apps are continually being discovered and discussed, ranging from sensitive data leakage [2–4] to privilege escalation [5–7]. The proliferation of malicious apps on Android devices and the theft of user privacy data have become major problems for the development of the Android biosphere [8].

Previous methods have normally used Android internal sandbox to protect Android system in the public network [9, 10]. However, the internal sandbox hardly analyzes the risk behaviors of Android applications. Most of the prior studies for risk behaviors of Android applications rely on static analysis and dynamic analysis. Research on static analysis of Android application behavior, including MDroid [11] and MHealth Apps [12] static analysis tools has achieved good results in the application of behavior. On the other hand, static analyses may have their limitations, for instance, detecting sensitive data leakage [13–16] and analyzing capability leakage [5, 17]. In addition, the effectiveness of static analysis is restricted due to the distinctive features of Android's programming paradigm. Android apps are based on the Android OS, which can be regarded as a giant set of libraries containing both Java code and native code (so far, Android has consisted of more than 13 million lines of code [18]). Reference [19] proposed a verifiable diversity ranking search scheme over encrypted outsourced data while preserving privacy in cloud computing, which also supports Android data.

Contrary to static analysis, dynamic analysis executes selected program paths and thus can precisely identify property violations [20, 21]. Nonetheless, Android apps are tightly coupled with the Android OS which consists of a set of libraries containing both Java and native code and complex interprocess communications. To address this, Qi [22] proposed the taint propagation information tracking system based on TaintDroid, by modifying the Android system source code. Reference [23] implemented a tool named DroidInjector, which reproduced the malicious behavior of the application through the simulation of the application behavior. Gharib [24] put forward a system named ProfileDroid, which can depict the application behavior from 2 aspects of semantics and behavior and improve the accuracy

of behavior analysis. Ardeshiricham [25] then designed a more detailed system named DroidScope. Reference [26] proposed an Activity call graph analysis method to generate UI interaction script automatically for Android applications. It can not only depict the application behavior semantically but also analyze the behavior characteristics of the application components. Lin has designed and implemented a tool named sandbox [27]. It can analyze sensitive actions from 2 aspects of general behavior and internal behavior. Reference [28] presented a scheme named SecDisplay for trusted display service; it protects sensitive data displayed from being stolen or tampered surreptitiously by a compromised OS. To detect DDoS attack, Cheng [29] proposed an abnormal network flow feature sequence prediction approach which could fit to be used as a DDoS attack detector in the big data environment and solve the aforementioned problems. And that method can also be used for detecting risk behaviors of Android app. Liu [30] designed a more practical privacy protection data aggregation protocol based on a new trusted model to avoid attack from DDoS.

In general, dynamic analysis on the application behavior can accurately judge the malicious behavior; however it cannot guarantee that all the codes for the implementation of application coverage, so that there might be a high missing rate. Static analysis can ensure the code coverage, but the accuracy of static analysis is easily affected by code obfuscation technology. For this reason, the paper begins with the application of Android risk behavior perspective, dynamic analysis based on the idea of the design and implementation with hook mechanism. It can automatically install, start, and uninstall the application and can simulate user actions in the application after the start, at the same time to monitor and record the behavior of the application. On this way, we put forward an evaluation method based on risk behavior. A security risk for the application of behavior is analyzed, so that users about risk related applications can have a clear anticipation.

2. Preliminary

2.1. Information Entropy. Information entropy, also known as Shannon entropy, is proposed by Shannon to solve the problem of quantitative measurement of information. In information theory, entropy is used to measure the expected value of a random variable. It represents the amount of information lost in the process of information transmission before being accepted and is also the entropy of information. In thermodynamics, the definition of entropy is the logarithm of the number of possible states of a system; its physical meaning is a measure of the degree of disorder in the system. Entropy is the measure of uncertainty of random variables in information theory. Information entropy considers that the magnitude of information of a message is directly related to its uncertainty. Named after Boltzmann's H-theorem, Shannon denoted the entropy H of a discrete random variable X with possible values $x_1, \ldots, x_n$ as

$$H(X) = E(I(X)) \tag{1}$$

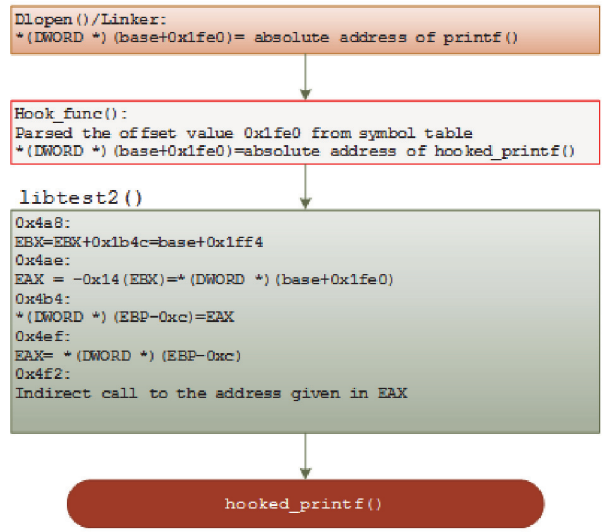


Figure 1: Example of how the hook function intercepts the call to printf() and reroutes the call to hooked_printf().

Here E is the expected value, and I is the information content of X. $I(X)$ is itself a random variable. If p denotes the probability mass function of X then the entropy can explicitly be written as

$$\begin{aligned} P(X) &= \sum_{i=1}^{n} p(x_i) I(x_i) = \sum_{i=1}^{n} p(x_i) \log_b \frac{1}{p(x_i)} \\ &= -\sum_{i=1}^{n} p(x_i) \log_b p(x_i) \end{aligned} \tag{2}$$

2.2. A Hook Example. As printf function, for example, if the hook module wants to intercept the calls to printf() and redirect to another function, it should write the redirected function address to the offset addresses of the symbol printf defined in the relocation sections, after the linker loaded the dynamic library into memory.

To replace the call of the printf() function with the call of the redirected hooked_printf() function, as shown in the software flow diagram in Figure 1, a hook function should be implemented between the dlopen() and libtest() calls. The hook function will first get the offset address of symbol printf, which is 0x1fe0 from the relocation section named .rel.dyn. The hook function then writes the absolute address of hooked_printf() function to the offset address. After that, when the code in libtest2() calls into the printf(), it will enter the hooked_printf() instead.

3. Methods

The goal of sandbox is to avoid both apps and Android system code modifications. The design of sandbox is to directly modify the apps virtual-memory tampering with ART internals representation of Java classes and methods. ART-sandbox consists of two components. The first component is the core engine written in C and the other one is the Java side that is

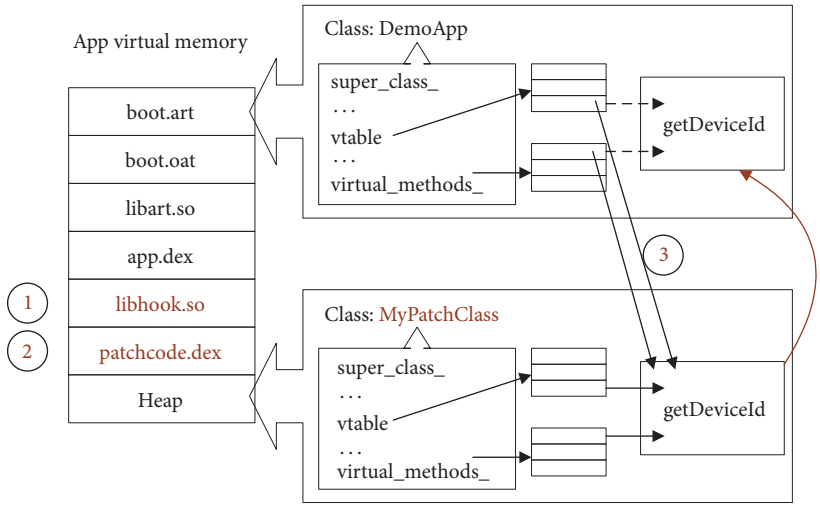


Figure 2: App virtual-memory layout.

a bridge for calling from user-defined Java code to sandbox's core. The core engine aims to find target methods in virtual memory, load user-supplied DEX files, hijack the vtable, and set native hooks. Moreover, it registers the native methods callable from the Java side. Sandbox is configured by reading a user-supplied JSON formatted configuration file containing the target methods list.

Figure 2 represents the apps memory layout while sandbox hooking library is enabled. The hooking library is loaded inside the apps virtual memory (step 1), and then sandbox loads the user-defined patch code by DexClassLoaders methods (step 2). After this, sandbox uses its internal functions to retrieve target methods reference. It can hook these methods by both vtable and virtual methods hijacking (step 3).

To get the target methods reference, sandbox uses the JNI function FindMethodID. Sandbox overwrites the target methods entry within both the vtable and virtual methods array by writing the address of the methods patch code. The original methods reference is not modified by sandbox and its address is stored inside the internal data structures. This address will be used to call the original method implementation.

3.1. Hook Virtual Space. When the Android application starts the Activity, the ActivityManager.start-Activity() method will eventually be invoked no matter what API call is passed. This call is a remote Binder service (speeding up the call), and the Android application will first look up the Binder service cache in the local process. Virtual app(VA) intervened in the invocation process by the following way.

(1) Replace the local ActivityManagerServise(AMS) Binder service for the proxy object that is constructed for the VA to take over the call. This step is realized by Java reflection technology.

(2) After taking over the AMS, when the startActivity is invoked to open more applications, the Activity in the VA modification Intent is the occupied Activity that has been

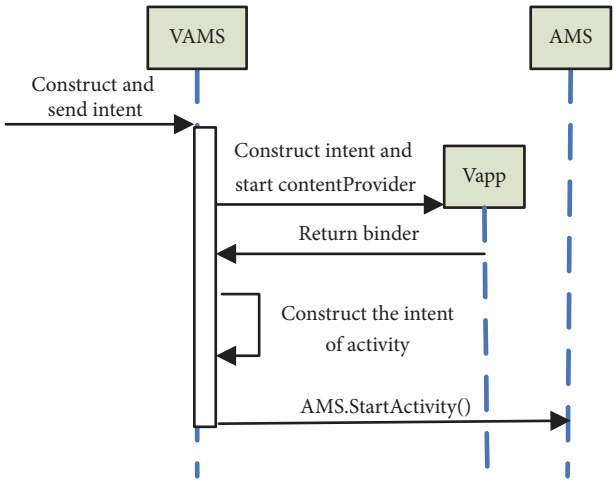


Figure 3: Diagram of starting Activity.

declared in VA. The goal of this step is to directly start Activity without AndroidManifest.xml.

(3) When the multiple application process is started, the message processing callback is increased by the ActivityThread.mH.mCallback. This step takes over the message callback of more than one application master threads.

On the basis of the above modification, the multiple application Activity startup process can be divided into the following two steps: start Activity (shown as in Figure 3) and resume Activity (shown as in Figure 4).

3.2. APP Risk Behavior. An application invokes an API that truly reflects the behavior of the application in the Android system. For example, an application that generates network behavior will certainly invoke the API associated with the network communication. An application that generates file behavior will certainly call the file-related APIs. Hence, to

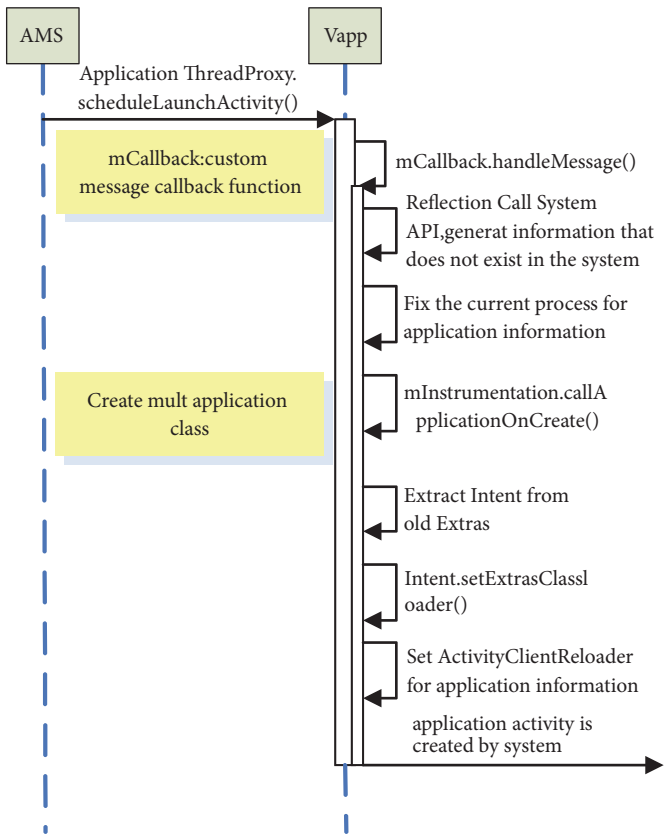


Figure 4: mCallback resume Activity information from Intent.

describe an application's behavior, one can use it as a standard for invoking the API. Once users have installed malicious applications in Android, these apps typically have some of the following common malicious behaviors in Android:

(1) The backstage sending the charge message and calling the toll telephone.

(2) Theft of user information (including mobile phone messages, call records, mobile phone IMEI, IMSI number, and user-used operators);

(3) Access to the user's location information, to open the mic recording and camera in the backstage.

(4) Backstage networking, transmission of user information, and consumption of user network traffic.

(5) Camouflage process, in the backstage to kill other mobile phone processes (such as Alipay application process) and then camouflage another process to cheat.

According to the description above, if an application invokes one or more APIs required to implement the above behavior, there is a certain degree of risk that the user can install the application. Table 1 shows some of the APIs and their behavior levels.

When the risk is large to a certain extent, users should be informed. The level of risk represented by different APIs is not the same for risk APIs. In general, the user's economic interests as a direct risk measurement criteria:

(1) APIs that may directly cause loss of property to the user, with the highest degree of risk

(2) The API that can obtain or disclose user's privacy, its risk degree being secondary

(3) To modify the system settings and User Configuration, damage to the system environment of the API, the degree of risk being relatively low.

Based on the theory of information entropy, this paper proposes a new approach to evaluate the risk behavior of Android using information entropy. One installs the application into the sandbox and runs and simulates user actions for a fixed number of times, such as 500. Suppose that, in this process, all the sensitive APIs invoked are $k1, k2, \ldots, kn$, and set the information entropy used to evaluate the risk behavior of the application to δ, and set s as the total API numbers that has occurred for all behavior:

$$s = \sum_{i=1}^{n} k_i \tag{3}$$

$$\delta = -\sum_{i=1}^{n} \frac{k_i}{s} \times \log_b \frac{k_i}{s} \tag{4}$$

Because malicious applications generally focus on risk behavior, they frequently invoke sensitive APIs. This paper uses a set of sensitive APIs to describe and characterize an application. If δ is greater, its entropy will be more than the normal application of information entropy. By calculating an

Table 1: APIs and risk levels defined by authors.

Evaluation Project	Danger level	Evaluation Project	Danger level
Virus scanning	high	Apply data to any backup	medium
Sensitive word Information	medium	Apply Signature Not verified	medium
Advertising SDK Detection	low	Sensitive function calls	medium
Third-party SDK detection	low	Java Layer Dynamic debugging	low
Java Code decompile	high	Load Dex from SDcard	low
So file crack	high	Implicit invocation of intent components	low
Tampering and two-time packaging	high	WebView Remote Code	high
Dynamic injection attack	high	Database injection	high
Interface Hijacking	high	ContentProvider Data Disclosure	high
Input listening	high	Encryption method not safe to use	high
HTTP Transport data	high	HTTPS not verified	medium
WebView PlainText Store password	high	Download any apk	medium
PlainText digital certificate	high	Global writable Internal files	medium
Debug Log functions	high	DDoS	medium
Resource File Disclosure	medium	Residual test information	low
Dynamic Debug Attacks	medium	WebView Bypass Certificate validation	low
Activity Component Export	medium	Unsafe use of random numbers	low
Service component Export	medium	Intent Scheme URL	low
Broadcast receiver Component Export	medium	Fragment injection attack	low
Content Provider Component Export	medium		

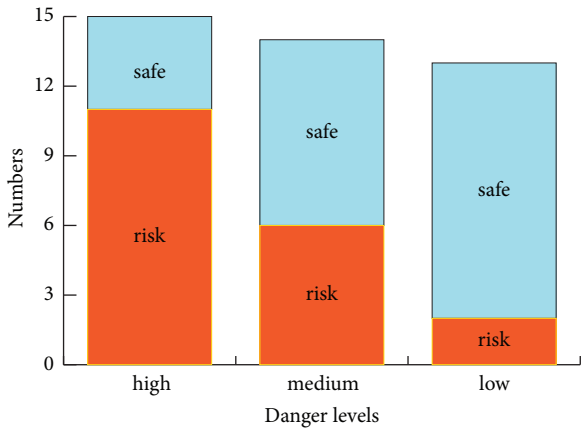
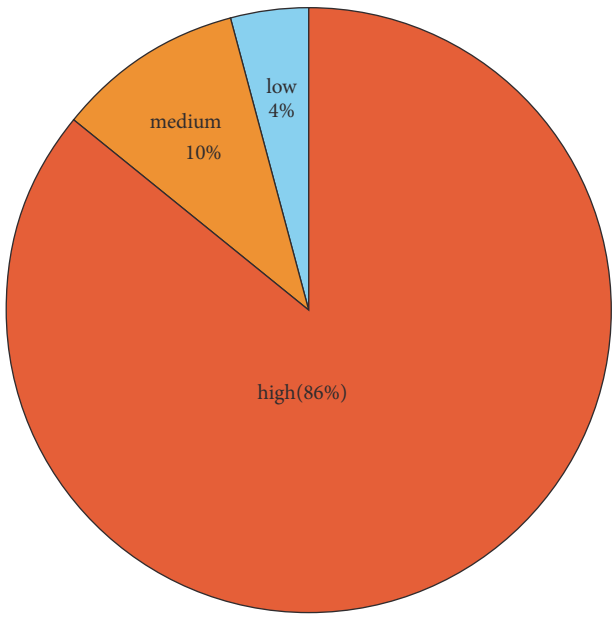


Figure 5: Test results for authclient.apk.



In total

Figure 6: Found problems for authclient.apk.

application on the sensitive API set of information entropy δ, one can judge the level of risk.

4. Experiments and Results

We firstly test one app which is named authclient.apk downloaded from Google. Figures 5 and 6 show the test results.

This paper collects 1200 malicious Android apps that have been identified by each major mobile phone's security platform as a sample of malicious applications and crawls the top-ranked downloads in Google's official App Store, a total of 400 applications, as a normal application sample. In general, Google's official App Store is generally considered to be a more secure app store, and the application of the top-ranked Google stores has withstood the test of countless users and security vendors. It can be considered a safe and normal application. Based on the above 2 sample sets, the sandbox is used to monitor and record its API calls, and the results are statistically categorized as shown in Figures 7 and 8.

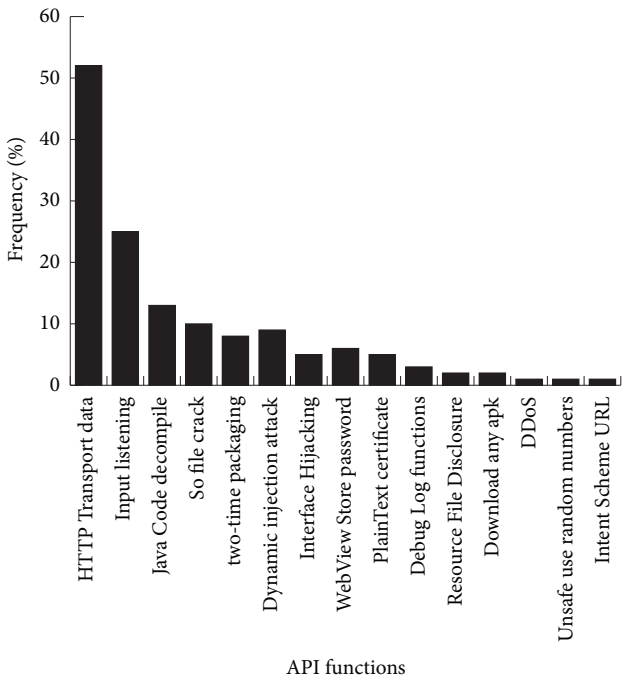


Figure 7: The information entropy of malicious apps.

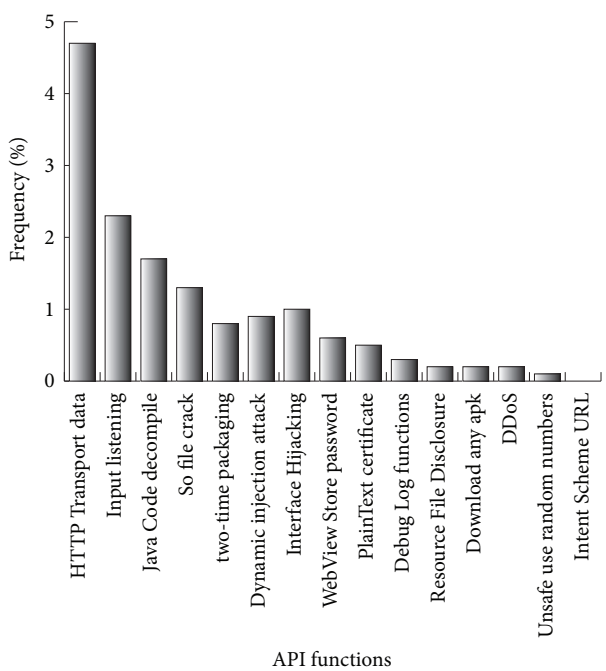


Figure 8: The information entropy of normal apps.

From the above results, network API calls occupy the main position for both malicious and normal application, and that is a trend in the future development of Android applications. In addition to the network API, there are malicious applications to other sensitive API calls and normal application of the obvious difference. As can be seen from Figures 6 and 7, malicious applications have significantly more frequent calls to sensitive APIs than normal applications, and these APIs can capture the user's IMEI code, IMSI code, phone number, SIM card sequence code, etc., so these APIs are obviously more risky than network APIs much higher.

5. Discussion and Conclusion

With the rapid development of mobile Internet, the Android applications have become an indispensable tool for people's daily life, and understanding the risks of Android applications is very important. This article focuses on the analysis and evaluation of the Android application risk behavior, monitors and records the behavior of Android applications through the Android sandbox, and uses the information entropy theory to analyze and evaluate the risk behavior of Android applications. This method can provide reference for application store review and also make Android users have a clear assessment of the risks associated with the application. Based on the above methods, this paper collects more than 1200 malicious applications and 400 normal applications, calculates the information entropy of both, compares them, and verifies the validity of the method described in this paper.

However, before being widely deployed in practical applications, this proposed sandbox has to tackle the privacy and efficiency challenges in sharing schemes [31]. We need to send users' data to clouds and guarantee the security of users data while ensuring rapid transmission of instant data [32, 33] and the security of public storage clouds [34, 35]. Note that mining data from multiple data sources to extract useful information [36, 37] for better understanding of security risk evaluation should be considered in the future study.

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 National Natural Foundation of Science, China (41274109), the Innovative Team Project of Sichuan Province (2015TD0020), and the New Zealand Marsden Fund.

References

[1] S. Blackshear, A. Gendreau, and B. Y. E. Chang, "Droidel: a general approach to android framework modeling," in *Proceedings of the ACM Sigplan International Workshop on State of the Art in Program Analysis*, pp. 19–25, 2015.

[2] A. Feizollah, N. B. Anuar, R. Salleh, and F. Amalina, *Comparative Evaluation of Ensemble Learning and Supervised Learning in Android Malwares Using Network-Based Analysis*, Springer International Publishing, 2015.

[3] Y. Zhou, X. Zhang, X. Jiang, and V. W. Freeh, "Taming information-stealing smartphone applications (on android)," in *Proceedings of the International Conference on Trust and Trustworthy Computing*, pp. 93–107, 2011.

[4] M. Grace, W. Zhou, X. Jiang, and A.-R. Sadeghi, "Unsafe exposure analysis of mobile in-app advertisements," in *Proceedings of the 5th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '12)*, pp. 101–112, April 2012.

[5] L. Lu, Z. Li, Z. Wu, W. Lee, and G. Jiang, "CHEX: statically vetting android apps for component hijacking vulnerabilities," in *Proceedings of the ACM Conference on Computer and Communications Security (CCS '12)*, pp. 229–240, October 2012.

[6] P. Pearce, A. P. Felt, G. Nunez, and D. Wagner, "AdDroid: privilege separation for applications and advertisers in Android," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security (ASIACCS '12)*, pp. 71-72, Seoul, Republic of Korea, May 2012.

[7] S. Bugiel, L. Davi, A. Dmitrienko, T. Fischer, A. R. Sadeghi, and B. Shastry, "Towards taming privilege-escalation attacks on android," in *Proceedings of the Annual Network and Distributed System Security Symposium*, vol. 130, pp. 346–360, 2012.

[8] P. Zhao, K. Bian, T. Zhao et al., "Understanding smartphone sensor and app data for enhancing the security of secret questions," *IEEE Transactions on Mobile Computing*, vol. 16, no. 2, pp. 552–565, 2017.

[9] Z.-X. Shen, C.-W. Hsu, and S. W. Shieh, "Security semantics modeling with progressive distillation," *IEEE Transactions on Mobile Computing*, vol. 16, no. 11, pp. 3196–3208, 2017.

[10] F. Roesner, T. O. Kohno, and D. Molnar, "Security and privacy for augmented reality systems," *Communications of the ACM*, vol. 57, no. 4, pp. 88–96, 2014.

[11] K. Moran, M. Tufano, C. Bernal-Cárdenas et al., "Mdroid+: a mutation testing framework for android," in *Proceedings of the the 40th International Conference*, pp. 33–36, Gothenburg, Sweden, May 2018.

[12] M. Hussain, A. Al-Haiqi, A. Zaidan et al., "A security framework for mHealth apps on Android platform 1," *Computers & Security*, vol. 75, pp. 191–217, 2018.

[13] V. Rastogi, Y. Chen, and W. Enck, "AppsPlayground: automatic security analysis of smartphone applications," in *Proceedings of the 3rd ACM Conference on Data and Application Security and Privacy (CODASPY '13)*, pp. 209–220, ACM, February 2013.

[14] S. Arzt, S. Rasthofer, C. Fritz et al., "FLOWDROID: precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for Android apps," *ACM Sigplan Notices*, vol. 49, no. 6, pp. 259–269, 2014.

[15] C. Gibler, J. Crussell, J. Erickson, and H. Chen, "Androidleaks: automatically detecting potential privacy leaks in android applications on a large scale," in *Proceedings of the International Conference on Trust and Trustworthy Computing*, pp. 291–307, 2012.

[16] A. P. Fuchs, A. Chaudhuri, and J. S. Foster, "Scandroid: Automated security certification of android applications," in *Proceedings of the 31st IEEE Symposium on Security*, 2009.

[17] Y. Zhou and X. Jiang, "Systematic detection of capability leaks in stock android smartphones," in *Proceedings of the Nineteenth Annual Network and Distributed System Security Symposium (NDSSS '12)*, 2012.

[18] D. Katz and A. Nagy, "VuFind: solr power in the library," in *Open Source Technology: Concepts, Methodologies, Tools, and Applications*, 2015.

[19] H. P. Yuling Liu and J. Wang, "Verifiable diversity ranking search over encrypted outsourced data," *CMC, Computers, Materials & Continua*, vol. 55, no. 1, pp. 037–057, 2018.

[20] T. Azim and I. Neamtiu, "Targeted and depth-first exploration for systematic testing of android apps," in *Proceedings of the 28th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA '13)*, pp. 641–660, October 2013.

[21] S. Hao, B. Liu, S. Nath, W. G. Halfond, and R. Govindan, "PUMA: programmable UI-automation for large-scale dynamic analysis of mobile apps," in *Proceedings of the International Conference on Mobile Systems, Applications, and Services ACM*, pp. 204–217, 2014.

[22] W. Qi, W. Ding, X. Wang et al., "Construction and mitigation of user-behavior-based covert channels on smartphones," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, pp. 44–57, 2018.

[23] W. Fan, Y. Sang, D. Zhang, R. Sun, and Y. Liu, "DroidInjector: a process injection-based dynamic tracking system for runtime behaviors of android applications," *Computers & Security*, vol. 70, pp. 224–237, 2017.

[24] A. Gharib and A. Ghorbani, "DNA-Droid: a real-time android ransomware detection framework," in *Proceedings of the International Conference on Network and System Security*, pp. 184–198, 2017.

[25] A. Ardeshiricham, W. Hu, J. Marxen, and R. Kastner, "Register transfer level information flow tracking for provably secure hardware design," in *Proceedings of the 20th Design, Automation and Test in Europe Conference and Exhibition (DATE '17)*, pp. 1691–1696, March 2017.

[26] Y. Liu, S. Wang, Y. Yang, Y. Chen, and H. Sun, "An automatic UI interaction script generator for android applications using activity call graph analysis," *EURASIA Journal of Mathematics, Science and Technology Education*, vol. 14, no. 7, pp. 3159–3179, 2018.

[27] L. Xin, "Malware detection technology research of android platform based on sandbox," *Electronic Design Engineering*, vol. 24, no. 12, pp. 48–50, 2016.

[28] J. Cui, Y. Zhang, Z. Cai, A. Liu, and Y. Li, "Securing display path for security-sensitive applications on mobile devices," *CMC: Computers, Materials & Continua*, vol. 55, no. 1, pp. 017–035, 2018.

[29] J. Cheng, R. Xu, X. Tang, V. S. Sheng, and C. Cai, "An abnormal network flow feature sequence prediction approach for ddos attacks detection in big data environment," *CMC: Computers, Materials & Continua*, vol. 55, no. 1, pp. 095–119, 2018.

[30] Y. Liu, W. Guo, C. Fan, L. Chang, and C. Cheng, "A practical privacy-preserving data aggregation (3PDA) scheme for smart grid," *IEEE Transactions on Industrial Informatics*, 2018.

[31] Z. Cai, H. Yan, P. Li, Z.-A. Huang, and C. Gao, "Towards secure and flexible EHR sharing in mobile health cloud under static assumptions," *Cluster Computing*, vol. 20, no. 3, pp. 2415–2422, 2017.

[32] C. Wang, J. Shen, Q Liu I, Y. Ren, and T. Li, "A novel security scheme based on instant encrypted transmission for internet of things," *Security and Communication Networks*, vol. 2018, pp. 1–7, 2018.

[33] J. Shen, C. Wang, T. Li, X. Chen, X. Huang, and Z.-H. Zhan, "Secure data uploading scheme for a smart home system," *Information Sciences*, vol. 453, pp. 186–197, 2018.

[34] L. Yang, Z. Han, Z. Huang, and J. Ma, "A remotely keyed file encryption scheme under mobile cloud computing," *Journal of Network and Computer Applications*, vol. 106, pp. 90–99, 2018.

[35] L. Jin, L. Yatkit, C. Xiaofeng, L. Patrick, and L. Wenjing, "A hybrid cloud approach for secure authorized deduplication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 5, pp. 90–99, 2015.

[36] R. Wang, W. Ji, M. Liu et al., "Review on mining data from multiple data sources," *Pattern Recognition Letters*, vol. 109, pp. 120–128, 2018.

[37] J. Wu, J. Long, and M. Liu, "Evolving RBF neural networks for rainfall prediction using hybrid particle swarm optimization and genetic algorithm," *Neurocomputing*, vol. 148, pp. 136–142, 2015.

Hindawi
Security and Communication Networks
Volume 2018, Article ID 2507293, 12 pages
https://doi.org/10.1155/2018/2507293

Research Article

Street-Level Landmark Evaluation Based on Nearest Routers

Ruixiang Li, Yuchen Sun, Jianwei Hu, Te Ma, and Xiangyang Luo

State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China

Correspondence should be addressed to Xiangyang Luo; luoxy_ieu@sina.com

Received 10 May 2018; Accepted 5 July 2018; Published 18 July 2018

Academic Editor: Lianyong Qi

High reliable street-level landmarks are the basis of IP geolocation, but landmark evaluation methods having been proposed cannot evaluate the street-level landmarks. Therefore, in this paper, a street-level landmark evaluation method based on nearest router is proposed. The location organization declared is regarded as an area not a point. Firstly, the declared location of preevaluated landmark is verified by IP location databases. Secondly, the preevaluated landmarks are grouped according to their nearest router. Then, the distance constraint is obtained using delay value between landmark and its nearest router by delay-distance correlation. And relation model is established among distance constraint, organization's region radius, and distance between two landmarks. Finally, the reliability value of landmarks is calculated in each group based on relational model and binomial distribution. Landmarks evaluation experiment is taken based on 7082 preevaluated landmarks, and the results show that geolocation errors decrease obviously using evaluated landmarks. The mean error of 100 targets in Shanghai is reduced from 7.832km to 2.185km.

1. Introduction

The Internet of Things is an important part of a new generation of information technology, and it is an extended network based on the Internet. The core supporting technology of the Internet of Things is cloud computing. The cloud computing model enables real-time dynamic management and intelligent analysis of millions of items in the Internet of Things. IP geolocation is a technology to determine the geographical location of IP network entities. It has wide application prospects in the field of Internet of Things, such as ensuring that data flow within the privacy protection in cloud data sharing [1–3], improving the secret security in mobile cloud when secret keys distributed [4, 5], supporting location-based cloud services[6–8], and targeting covert communication subjects [9–11]. Landmark-based IP geolocation is an effective means to determining the region of the Internet entity. High reliable street-level landmarks are the basis of IP geolocation. Existing evaluation methods of landmarks are divided into evaluation of web-based landmarks and evaluation of the IP location databases.

A method, mining landmarks from web, has been proposed [12]. In this method, IP.cn database was used to evaluate the city location of the landmarks, and major Chinese ISP databases were used to assess the province location. On the basis of [12], Jiang H et al. [13] exclude the cloud services landmarks according to cloud service providers' IP address (such as Amazon AWS, Rackspace Hosting, and other top cloud providers), exclude hosting landmarks whose organizations in Whois do not contain specific keywords such as "university", "academy", or "institution", and use MaxMind database to verify landmarks' city location. E-GeoTrack algorithm [14] based on voting strategy and implicit information in nearest router was proposed to evaluate Internet forum landmarks obtained from a web page. Above methods can only evaluate the accuracy of city location, and there are large errors in fine-grained geolocation using evaluated landmarks. Wang Y et al. [15] obtain the web landmarks on the basis of paper [12] and verify the landmarks as follows: (1) if the zip code of landmark is inconsistent with the zip code entered during the query, the landmark will be deleted, (2) visit the website by IP address and domain name respectively, and if the contents, or heads (distinguished by <head> and </head>), or titles (distinguished by <title> and </title>) returned are different, the web site is considered to be hosted on a shared host or

Table 1: Calculating *LWV* of a DNS name.

url \ location	Loc_a	Loc_b	Loc_c	Loc_d
dns_a/url1	0.64	-	0.96	0.89
dns_a/url2	0.64	-	0.95	0.89
dns_a/url3	-	0.57	0.95	0.86
LWV of dns_a	0.43	0.19	0.95	0.88

on CDN ("Content Delivery Network"), and the landmark is deleted, and (3) delete the landmarks whose domains are the same but zip codes are different. This method deletes some hosting, shared hosting, and CDN landmarks. But to street-level geolocation, it is also necessary for evaluation further on this basis.

Different IPs in the same IP block may correspond to widely distributed geographical locations; thus, the location information in the foreign commercial IP location databases is very rough [16, 17]. Based on [16], Backstrom et al. [18] proposed to determine user's location based on their social network and use this location to improve IP location database. Poese et al. [19] studied IP location databases such as MaxMind, IP2LOCATION, IPInfoDB, and HostIP, found that the vast majority of IP data in United States were inconsistent with IP initial allocation data, and pointed that the city-level location of IP in these databases was inaccurate. In [20, 21], the authors analysed the distribution characteristics of IP address blocks of different geographic granularity in mainland China, compared the data consistency between multiple IP location database, verified the accuracy of the IP location database using a small number of existing landmarks, and improved the accuracy of the database by clustering IP blocks at different locations. For improving the accuracy of database, location data collected from search engine logs [22], IP allocation strategy of ISPs, urban areas, and population [23] were introduced into the database evaluation methods. Using those methods improve the accuracy at city level, but there are inadequacies in evaluating fine-grained (e.g., street level) locations.

In the absence of reliable street-level landmarks, there are significant errors in the evaluation of street-level landmarks based on street-level geolocation methods. And there are no other effective street-level landmark evaluation methods currently. In view of this reality, a street-level landmark evaluation method based on the nearest router is proposed. In this paper, the location of the organization is a region, and the radius of region is the maximum value of the distance calculated by latitude and longitude between the organizations' declared location and the location of network entity.

The method includes three steps. Firstly, multiple databases are used to verify the city-level location of the original candidate landmarks obtaining from web pages based on voting strategy. If the zip code information is obtained, it will be further used to verify. Then, route paths to candidate landmarks are obtained by using traceroute commend. Candidate landmarks are grouped based on the nearest router. And the relationship model between radius of organization, geographical distance, and network distance constraints is established within two candidate landmarks in the same group. Combining the binomial distribution, the reliability value of each landmark is calculated by relationship model.

The rest of this paper is organized as follows. The related work is introduced in Section 2. The principles and steps of street-level landmark evaluation method based on nearest routers are elaborated in detail in Section 3. Section 4 briefly analyses the street-level landmark evaluation method. The experimental results are given in Section 5. Finally, this paper is concluded in Section 6.

2. Related Work

Structon [12] is a webpage-based landmark mining method and the key idea is that web pages are embedded with rich geographic information (such as province/state, city, and zip code). The geographic information extracted from web page can be mapped onto the IP address of the web server. As a result, landmarks (network entities whose IP and geographic address have been known) are obtained. The main steps of Structon are as follows.

Firstly, according to HTML tags, each HTML file is parsed into multiple blocks, and each block is treated as a string roughly. For each block, a regular expression is used to extract geographical location information. Web servers for the same domain name are collected, and the location weight vector (*LWV*) for the domain name is calculated as Table 1.

Based on the weights of different urls of dns_a in different locations, the *LWV* values of dns_a in different location are calculated.

Then, the IP address and its corresponding location weight vector are taken as input. The multistage inference algorithm is used to increase the coverage and accuracy of the IP location database. The multistage inference algorithm includes three parts as follows.

Part 1. Location calculation of /24 IP segment: the location probability distribution function (*PDF*) is calculated by the *LWV* of each IP in /24 IP segment. The highest probability location is regarded as the geographical location of all IP in /24 IP segment. An example is shown in Table 2.

As shown in Table 2, the Loc_b is regarded as the IP segment geographical location.

Part 2. Error correction based on majority voting. If most of the IP subsegments have the same geographic location

Table 2: The *PDF* of the segment.

IP \ location	Loc_a	Loc_b	Loc_c	Loc_d	Loc_e
61.155.111.42	0.003	0.004	0.003	0.24	-
61.155.111.44	-	0.02	-	-	-
61.155.111.70	-	0.77	-	-	0.13
Location *PDF*	0.26%	68%	0.26%	20.5%	11%

and the remaining IP subsegments are at other locations, the entire IP segment is assumed to be in the same location.

Part 3. Inference based on AS and BGP information: the BGP routing table shows that some ASs only contain a small number of IP addresses. Therefore, these ASs are small ISPs, and these small ISPs are likely to be located in the same province or city. In these small ASs, if some IP are in the same location L, it can be well inferred that the entire ISP is also located in L.

Finally, IP address location tables of a major Chinese ISP and ip.cn database are used to verify the accuracy of inferred results.

Structon is a method using web page information to obtain landmarks. After location inference, a large amount of landmarks can be obtained. However, the landmarks verification strategies are rough, which only verify the provincial and city-level locations and cannot satisfy the requirements of street-level landmarks. At the same time, the database (such as ip.cn) is not completely accurate. The reliability of the verification results cannot be guaranteed if only using single database. After Structon, the researchers use multiple database to verify the provincial and city-level locations and use zip codes for further verification [13–15]. Those strategies improved the accuracy of landmarks obviously but did not reach the requirement of street-level landmarks evaluation.

Based on [12], a street-level landmarks evaluation method is proposed in this paper. Landmarks are collected from web, and IP location databases were used to verify the data consistency between declared location and databases' results. If the zip codes about original candidate landmarks were got, that would be used to verify the location further. Landmarks after location verification are named candidate landmarks. Then, the route paths from probe to candidate landmarks are got, and in the router path, the minimum single-hop delay from the nearest router to the candidate landmark is obtained. According to the same nearest router, candidate landmarks are divided into some group which can be considered as a set. Finally, in each set, we get distance constraint by delay-distance correlation and delay from nearest router to candidate landmark and calculate the distance between any two candidate landmarks by their latitude and longitude. According to the relationships among distance constraint, distance, and organization's region radius, all subsets satisfying the relationships can be found, and the reliable values of candidate landmarks in each subset can be calculated based on binomial distribution and initial probability. All groups are evaluated, and all landmarks whose reliable value is greater than reliability threshold can be selected out.

3. Methods

Aiming at the deficiencies of existing landmark evaluation methods in evaluating street-level landmarks, a street-level landmark evaluation method based on the nearest router is proposed. The precondition of this method is that if two terminal landmarks have the same nearest router, they are close in geographical location.

The method flowchart is shown in Figure 1. This method is mainly divided into three parts: location verification, landmarks grouping, and group landmarks evaluation. In location verification, candidate landmarks whose locations in multiple IP location databases are consistent with declared city were selected from original candidate landmarks. If zip code of original candidate landmarks is obtained, zip code needs to be used for verification further. In landmarks grouping, route paths to candidate landmarks were measuring many times, and final route path for one candidate landmark was got by merging route paths. The candidate landmarks with the same nearest router are grouped. In each group, if there are at least two landmarks for one institution, the one whose delay from nearest router to candidate landmark is minimal will be reserved. In group landmark evaluation, relationship model among distance constraint, region radius, and distance is established, and all subsets in which any two elements satisfy the relationship model will be found. In each subset, in the basis of initial probability and binomial distribution, the reliability values of candidate landmarks are calculated.

The main steps of the method are as follows:

Input: original candidate landmark and multiple IP location databases

Output: reliable landmarks with reliability value

Step 1 (original candidate landmarks acquisition). According to the landmark acquisition method mentioned in [12], information such as address, server domain, and zip code of companies, universities, and governments is obtained from web pages. Public mail server will be excluded, and server domain and address are converted to IP (IP_i) and latitude (lat_i) and longitude (lng_i), respectively. Original candidate landmark is marked as data pair ($< IP_i, lat_i, lng_i >$). If a domain name corresponds to n IPs, n data pairs will be marked whose IPs are different only. That means an organization may correspond to multiple candidate landmarks.

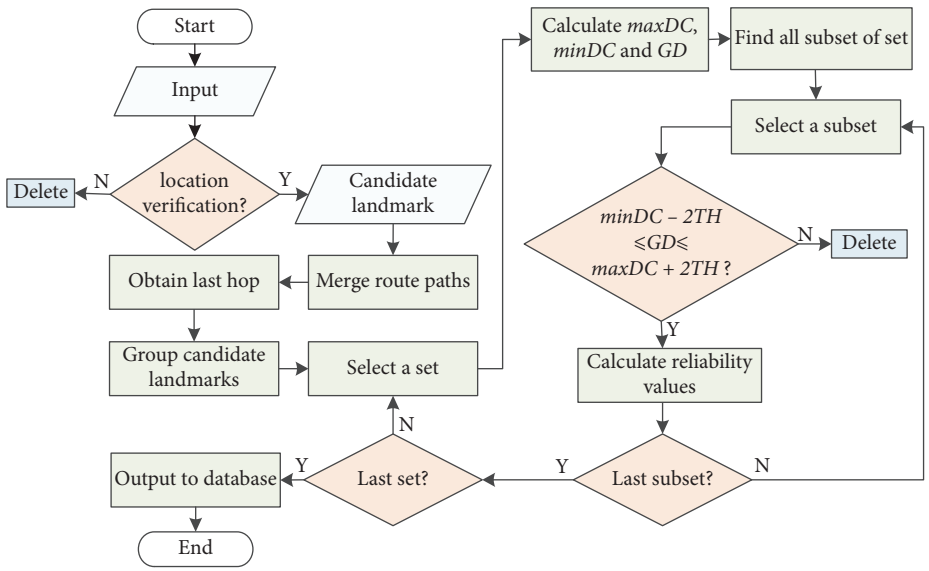


Figure 1: Method flowchart.

Step 2 (location verification). City address of original candidate landmark is searched from IP location databases such as IPIP, IP.cn, and Baidu. If the query results of databases and declared address are the same, the original candidate landmark will be retained. If zip code of original candidate landmark has been obtained, that will be used to verify landmark address further, and original candidate landmark (called "candidate landmark") whose claimed location belongs to the zip code area will be retained.

Step 3 (routing information acquisition). Use traceroute command to get route path from probe to candidate landmarks during small network delay fluctuation period repeatedly, and merge paths to obtain more detailed routing information and the minimum single-hop delay.

Step 4 (candidate landmark groupings). The candidate landmarks were grouped by the nearest common routers. Each group is called an evaluation landmark set. In a set, if multiple candidate landmarks belong to same organization, the one whose single-hop delay to nearest router will be retained, and other candidate landmarks will not participate in the evaluation process further, but the reliability value is the same as the retained one. If single-hop delay value to nearest router is greater than 1ms, the value may be regarded as inaccurate measurement one and the candidate landmark will not be evaluated further.

Step 5 (candidate landmark group evaluation). In an evaluation landmark set, the relation model among distance constraint, distance, and radius of region was built. All subsets satisfying relation model need to be found. This issue can be converted to solve complete subgraph problem in an undirected graph. The construction method of graph is as follows: the elements in the set are mapped to the vertices in the undirected graph. And if two elements in set satisfy the relation model, there is edge between the two vertices. The subset consists of all vertices in one complete subgraph. Obtaining all subsets is our goal. According to the IP allocation strategy of ISPs, the number of vertices of the graph is less than 64 generally. So, the issue can be solved in acceptable time.

In a subset, the reliability (recorded as "p_{re}") of evaluated landmarks is calculated based on initial reliability and binomial distribution.

$$p_{re} = 1 - \prod_{i=1}^{k} (1 - p_i) \tag{1}$$

where k is the number of elements in subset and p_i is the initial reliability value of ith element.

Step 6 (reliable landmark storage). Repeat Step 5 to evaluate all groups. When a candidate landmark appears in multiple subsets and may have multiple reliability values, the final reliability of the landmark evaluated is the maximum value. The reliability threshold is set as α, which means evaluated landmark is reliable when its reliability value is not less than α. And store the reliable landmark into result database.

Then, the process of merging the route paths and building the relation model are explained in detail.

Merging the Route Paths. Use traceroute command to get route path during small network delay fluctuation period repeatedly, and merge route paths. Figure 2 is an example of route paths merging.

The route path from probe (S) to candidate landmark (L) contains four routers (R1, R2, R3, and R4). In the first measurement, the IP addresses of R2, R3, and R4 are

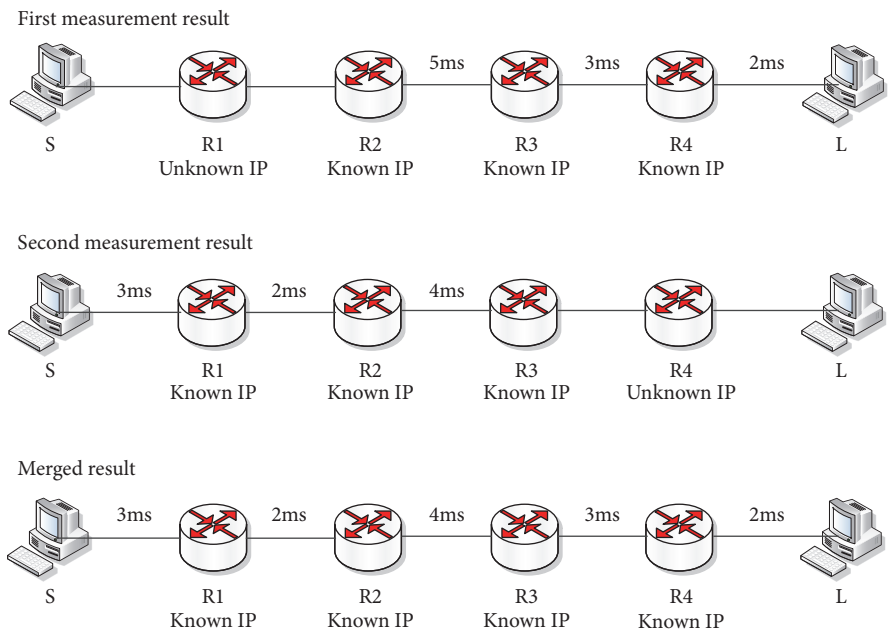


Figure 2: Example of route paths merging.

obtained, and the delay between R2 and R3 is 5ms. And, in the second measurement, the IP addresses of R1, R2, and R3 are obtained, and the delay between R2 and R3 is 4ms. Finally, the merged results are that the IP addresses of four routers are obtained and the delay of R2 and R3 is 4ms.

Building the relation model: for a subset, the geographical distance (GD) for two candidate landmarks L_i and L_j is calculated by latitude and longitude. According to the single-hop delay (t) from nearest router to candidate landmark, the distance constraint (DC) between candidate landmark and nearest router can be got by

$$DC = v * t \tag{2}$$

where v is the propagation velocity of the electromagnetic wave ($v = 20km/ms$) and t is the single-hop delay. Then, the minimum distance constraint (recorded as "$\min DC$") and the maximum distance constraint (recorded as "$maxDC$") between two candidate landmarks can be obtained.

$$\min DC = v * |t_1 - t_2| = |DC1 - DC2| \tag{3}$$

$$\max DC = v * (t_1 + t_2) = DC1 + DC2 \tag{4}$$

The trilateral relation among GD, $\min DC$, and $\max DC$ is established as inequality (5).

$$\begin{aligned} \min DC &\le GD \\ GD &\le \max DC \end{aligned} \tag{5}$$

Adding the radius of region (recorded as "TH"), relation model among distance constraint, region radius, and distance between L_i and L_j is established as inequality (6).

$$\begin{aligned} \min DC - 2TH &\le GD \\ GD &\le \max DC + 2TH \end{aligned} \tag{6}$$

4. Analysis of Method

In this section, the relation model among distance constraint, distance and region radius, and reliability calculation strategy will be discussed.

4.1. Relation Model. Organization location is a region, but online maps return the latitude and longitude which is a point when you translate organization address. And the given point deviates from the location of IP entity. When establishing the relationship between distance and distance constraint, we should consider the radius of organization region, as shown in Figure 3.

S is the nearest common router of L_1 and L_2, and P_1 is the latitude and longitude location given by online map. TH is the radius of organization region which IP entity of L_1 locates in. The distance constraint between S and L_1 is named $DC1$. Analysis of L_2 is the same as L_1. The distance of P_1 and P_2 is named GD. Therefore, the minimum value of the distance between IP entity of L_1 and IP entity of L_2 is $GD - 2TH$ (from P_1'' to P_2'), and the maximum value is $GD + 2TH$ (from P_1' to P_2''). According to (1) and (2), the values of $\max DC$ and $\min DC$ can be calculated.

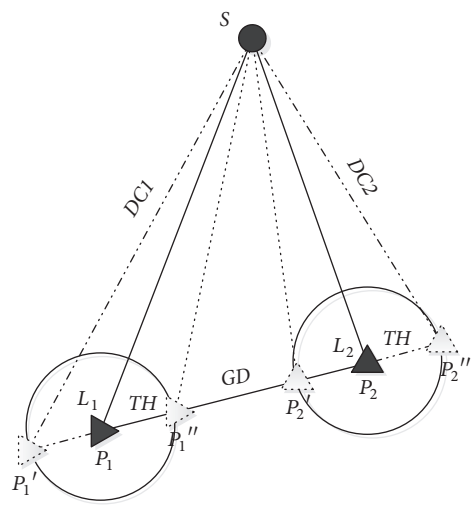


Figure 3: Relationship among distance, distance constraint, and radius.

Based on the triangular trilateral relationship, there are

$$\begin{aligned} \min DC &\le GD - 2TH \\ GD - 2TH &\le \max DC \end{aligned} \tag{7}$$

and

$$\begin{aligned} \min DC &\le GD + 2TH \\ GD + 2TH &\le \max DC \end{aligned} \tag{8}$$

That is equivalent to

$$\begin{aligned} \min DC + 2TH &\le GD \\ GD &\le \max DC + 2TH \end{aligned} \tag{9}$$

and

$$\begin{aligned} \min DC - 2TH &\le GD \\ GD &\le \max DC - 2TH \end{aligned} \tag{10}$$

Combining inequality (9) and inequality (10), we obtain the range of GD, which is the relation model (inequality (6)).

$$GD \in [\min DC - 2TH, \max DC + 2TH] \tag{11}$$

According to (11), for L_1 and L_2 after evaluation, the smaller the TH is, the closer the locations are.

4.2. Reliability Calculation Strategy. The information in the web page lacking effective verification deceives information receivers. Based on this reality, if the organization corresponds to multiple IP addresses in the same set, only the candidate landmark with the smallest single-hop delay value to the nearest router is retained. Therefore, the landmarks may be considered that are not related to each other.

For any candidate landmark L_i in a subset (marked as "C") of evaluation landmark set, the probability that L_i locates in organization region is p_i denoted as

$$P(L = True) = p_i, \quad p_i \in [0, 1] \tag{12}$$

Then, the probability that L_i does not locate in organization region is $1 - p_i$ denoted as

$$P(L_i = False) = 1 - P(L_i = True) = 1 - p_i \tag{13}$$

When there are n elements in C, the probability that all elements are not in the their region is denoted as P_c.

$$P_c = \prod_{i=1}^{n} P(L_i = False) = \prod_{i=1}^{n} (1 - p_i) \tag{14}$$

Since $p_i \in [0, 1]$, there is

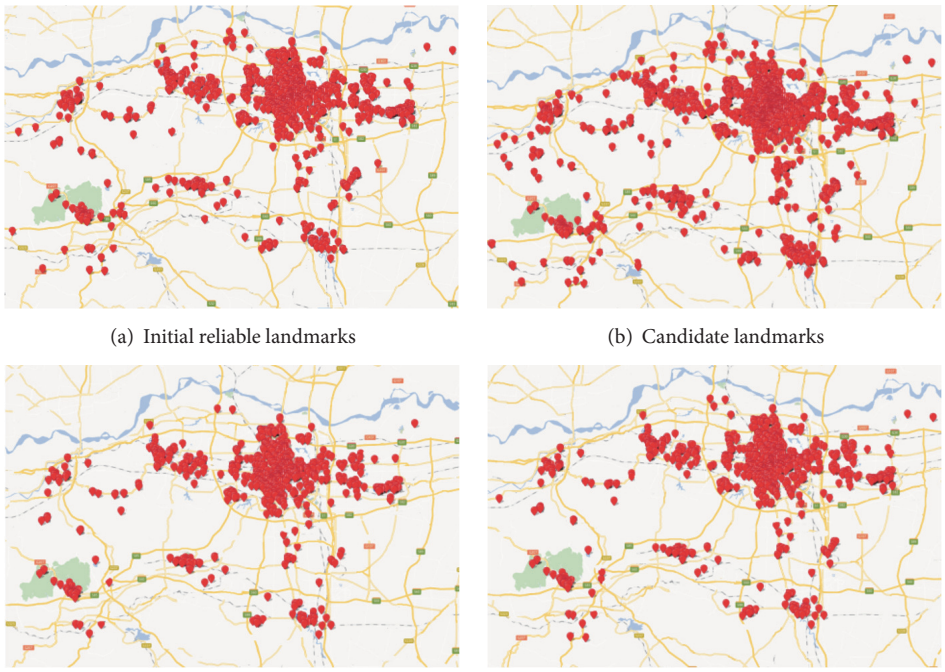
$$P_c \le 1 - p_j \tag{15}$$

where $p_j = P(L_j = True)$ and $p_j \in C$. Only when $\forall L_k \in C, k \ne j, p_k = P(L_k = True) = 0$, "=" is taken.

In each set, one organization only retains one candidate landmark. So, it can be considered that the elements in the set are not related to each other. When any two elements in subset satisfy inequality (6), the probability that our method mistakes the candidate landmarks as reliable landmarks is P_c, which is smaller than (or equal to) single candidate landmark misjudged as reliable. And the more elements in the subset, the higher reliability of evaluated landmarks.

5. Experimental Results

In order to verify the validity of our method, the feasibility verification experiment and the street-level landmark evaluation experiment were carried out.



(a) Initial reliable landmarks

(b) Candidate landmarks

(c) Evaluated Landmarks (TH=3)

(d) Evaluated Landmarks (TH=5)

Figure 4: Landmarks distribution.

Before conduction the experiments, with 7 days as a cycle and 30 minutes as a time period, we make a statistical analysis of the network stability in each time period within four cycles. Finally, the results show that, in China, the period (called "suitable period") from 22:30 to 06:30 (next day) is suitable for network measurement.

5.1. Feasibility Verification Experiment. 500 reliable street-level landmarks in Zhengzhou (called "initial reliable landmarks") were selected, and the distribution of initial reliable landmarks is shown in Figure 4(a). According to the distribution characteristics of initial reliable landmarks in geographical space, 500 street-level different locations were generated using a random function around the initial reliable landmark locations. 500 online IPs were selected which relate to Zhengzhou city in Baidu, IPIP, and IP.cn database. We generated 500 landmarks based on 500 online IPs and 500 street-level locations. The generated landmarks may be considered unreliable. In this experiment, the dataset (called "candidate landmarks") containing 500 reliable landmarks and 500 generated landmarks is shown in Figure 4(b), and the initial reliability values of all landmarks in dataset are set to 50%.

Performing 50 times routing measurements for each candidate landmark in suitable period, and merging the route paths, the delay values from nearest router to candidate landmarks were obtained. According to nearest router, candidate landmarks will be grouped. In this experiment, the reliability threshold was set to 75% and TH value was set to 3 and 5, respectively. When TH is 3, 416 evaluated landmarks (shown in Figure 4(c)) are evaluated. The ratio of evaluated landmarks to initial reliable landmarks is 83.2%. And when TH=5, 435 evaluated landmarks (shown in Figure 4(d)) are evaluated and the ratio is 87%.

Comparing Figures 4(a), 4(c), and 4(d), the evaluated landmarks are more concentrated in geospatial distribution than initial reliable landmarks. And compared with TH=3, the difference in geospatial distribution of evaluated landmarks (TH=5) is not obvious.

The experimental results show that street-level landmark evaluation method can obtain reliable landmarks from candidate landmarks effectively. The reasons that the method fail to obtain all initial reliable landmarks from candidate landmarks may be as follows.

(1) The delay measurement results are inaccurate. Relation model is established based on delay. For two candidate landmarks, if one's single-hop delay value increases, the value of min DC increases that may lead to $GD < \min DC$.

(2) The distribution of initial reliable landmarks is dispersed. If the distribution of initial reliable landmarks is dispersed, GD increases for two candidate landmarks that may lead to $GD > \max DC$.

5.2. Street-level Landmark Evaluation Experiment. The data are collected from Internet web pages, including organization name, location, web home page, and zip code. During the collecting process, the IPs with stable features, such as time servers, mail servers, and ftp servers, may be got more attention. The IP address is obtained based on DNS services, and latitude and longitude are obtained by online map

Table 3: The number of landmarks.

City	Before location verification	After location verification	After evaluation (TH=3)	After evaluation (TH=5)
Beijing	4658	1072	392	456
Shanghai	8966	3289	1227	1341
Shenzhen	6605	1857	783	869
Xiamen	3702	864	236	301

services. According to the famous mail server providers' IP addresses, part of obtained data is deleted. Eventually, the numbers of original candidate landmarks in Beijing, Shanghai, Shenzhen, and Xiamen are 4658, 8966, 6605, and 3702, respectively. Baidu, IPIP, IP.cn databases, and zip codes are used for location verification. For privacy protection, the obtained data only retain the IP address and information of latitude and longitude. After location verification, the numbers of candidate landmarks in Beijing, Shanghai, Shenzhen, and Xiamen are 1072, 3289, 1857, and 864, respectively. In addition, data set in this experiment includes 400 reliable street-level landmarks also (100 landmarks in each city). Reliable landmarks are used to verify the evaluated landmarks and do not participate in the evaluation process.

The initial reliability of all candidate landmarks is set to 50%, and reliability threshold is set to 75%. 50 times routing measurements are performed for each candidate landmark in suitable period. Candidate landmarks are evaluated when TH=3 and TH=5, respectively. The number of landmarks in each city is shown in Table 3.

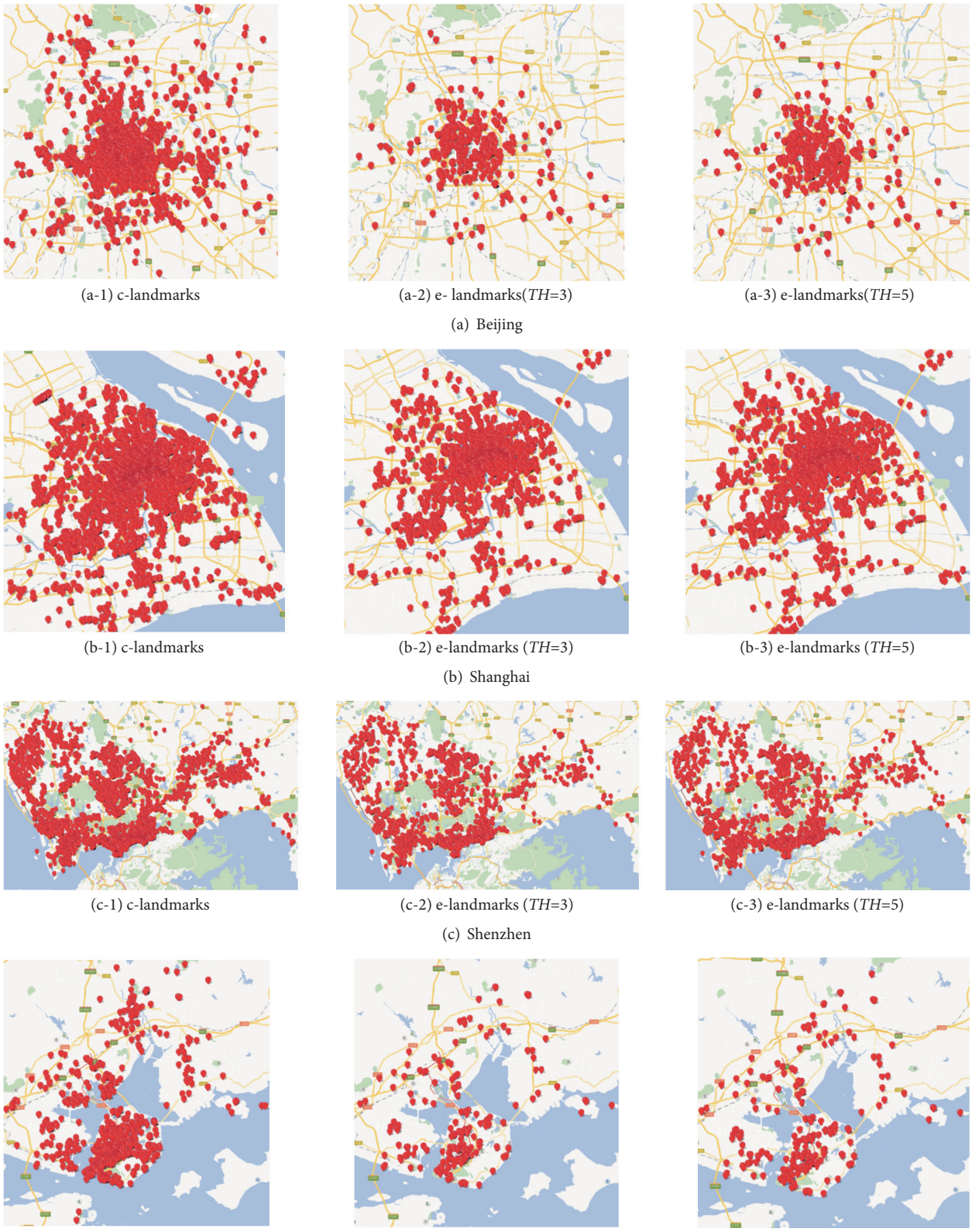
The number of candidate landmarks has been greatly reduced after location verification because of servers hosting. There are two reasons that the number of landmarks decreased after evaluation. One reason is that some of the landmarks are not reliable and do not satisfy the relation model. Another reason is that the method proposed in this paper cannot evaluate the landmarks whose nearest router connect with a landmark only, although this landmark may be reliable. According to inequality (6), the larger the value of TH is, the more landmarks rest after evaluation. When the value of TH increases, the range of GD is expanded, meaning that the allowance error distance from the organization center to server location and allowance error delay from nearest router to landmark increase. Therefore, the larger the TH values are, the more landmarks are obtained according to relation model. When TH=3, the numbers of evaluated landmarks in Beijing, Shanghai, Shenzhen, and Xiamen are 392, 1227, 783, and 236, respectively, accounting for 36.57%, 35.15%, 42.16%, and 27.31% of candidate landmarks and 8.42%, 13.69%, 11.85%, and 6.37% of original candidate landmarks. When TH=5, the numbers are 456, 1341, 869, and 301, respectively, accounting for 42.54%, 40.77%, 46.8%, and 34.84% of candidate landmarks and 9.79%, 14.96%, 13.16%, and 8.13% of original candidate landmarks. The distribution of candidate landmarks and evaluated landmarks of each city is shown in Figure 5 (candidate landmarks marked as "c-landmarks" and evaluated landmarks marked as "e-landmarks").

In each city, candidate landmarks (obtaining method is similar to Structon [12], called "before evaluation" landmarks) and evaluated landmarks are used to locate 100 reliable street-level landmarks using SLG algorithm [15], respectively. The relationship between the mean error and the number of landmarks is shown in Figure 6.

In Figure 6, the mean error using evaluated landmarks to locate 100 reliable street-level landmarks is smaller than using the same number of candidate landmarks in the same city. The mean error of geolocation using landmarks evaluated by TH=3 is slightly less than the value caused by the landmarks evaluated by TH=5. The geolocation accuracy is affected by the accuracy of landmarks on the one hand. The base of street-level geolocation is street-level landmarks. On the other hand, the geolocation accuracy is affected by the number of reliable landmarks. The accuracy of geolocation increases when the number of landmarks increases. In Figure 6, when the number of reliable landmarks increases from 0 to 300, the geolocation error decreases rapidly. When the number of reliable landmarks is greater than 300, the number of landmarks increases, and the speed of the geolocation error decreases gradually. When all candidate landmarks are used for geolocation, the mean error in Beijing, Shanghai, Shenzhen, and Xiamen is 9.624 km, 7.832 km, 7.634 km, and 8.994 km, respectively. But the value is 3.98km, 2.185km, 2.234km, and 5.237km respectively, when TH=3, and the value is 3.943 km. 2.198 km, 2.241 km, and 4.473 km, respectively, when TH=5.

In Beijing, Shanghai, Shenzhen, and Xiamen city, when all candidate landmarks and evaluated landmarks are used to locate 100 reliable street-level landmarks using SLG algorithm. The relationship between geolocation error and cumulative probability is shown in Figure 7.

Figure 7 shows that the location accuracy of using evaluated landmarks is significantly higher than the value of using candidate, and the difference in location accuracy between TH=3 and TH=5 is not obvious. In Beijing, Shanghai, and Shenzhen, comparing with pre-landmarks, the probability of geolocation error less than 5km increases by more than 35%, and the probability of geolocation error less than 10km increases by 20%, when evaluated landmarks are used. In Xiamen, the probability of geolocation error less than 5km and 10km increases by 20% and 15%, respectively. The main



(a-1) c-landmarks (a-2) e- landmarks(TH=3) (a-3) e-landmarks(TH=5)

(a) Beijing

(b-1) c-landmarks (b-2) e-landmarks (TH=3) (b-3) e-landmarks (TH=5)

(b) Shanghai

(c-1) c-landmarks (c-2) e-landmarks (TH=3) (c-3) e-landmarks (TH=5)

(c) Shenzhen

(d-1) c-landmarks (d-2) e-landmarks (TH=3) (d-3) e-landmarks (TH=5)

(d) Xiamen

Figure 5: Landmarks distribution of each city.

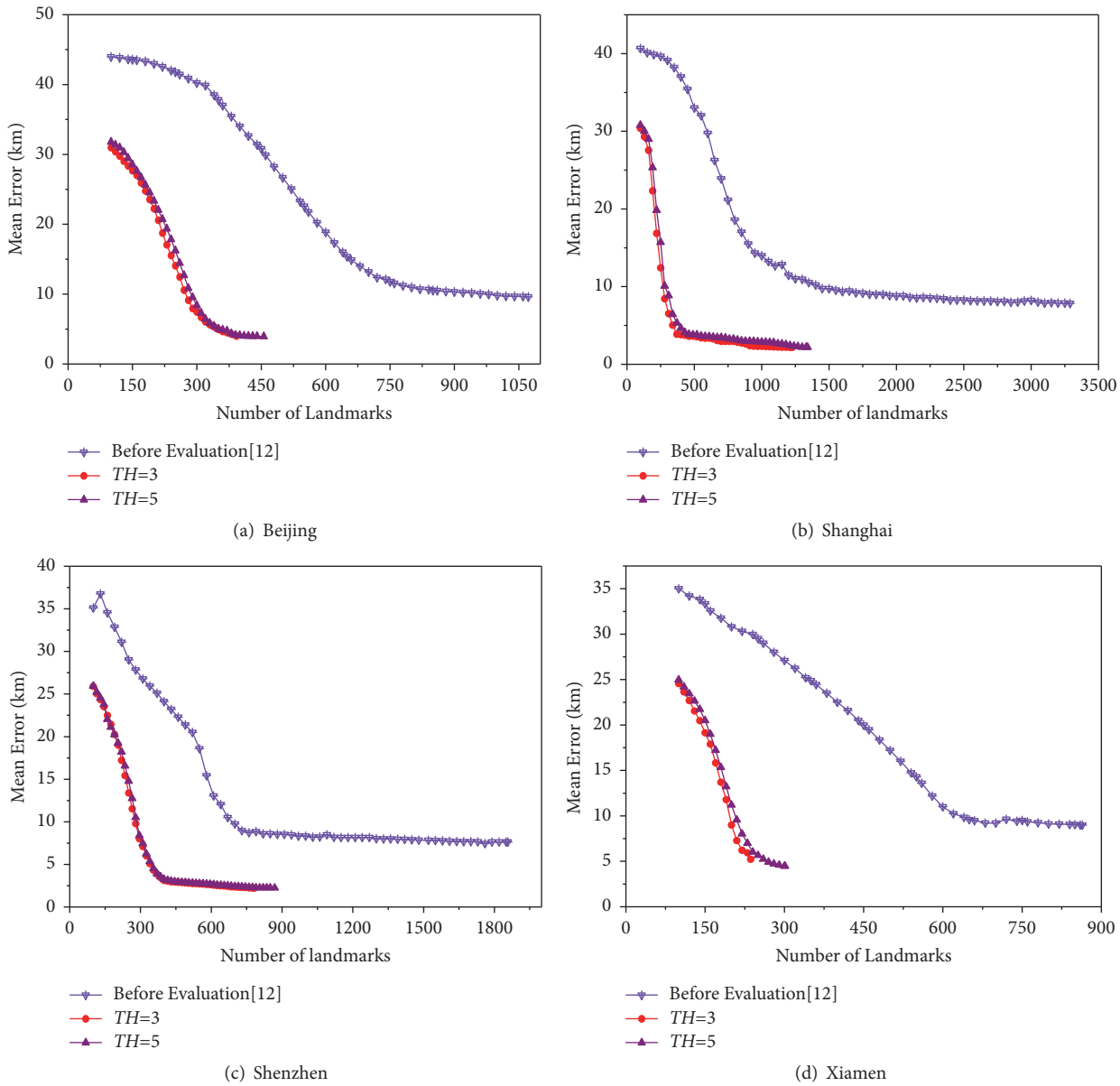


Figure 6: Relationship between the mean error and the number of landmarks.

reason for the low increasing percentage is that the number of evaluated landmarks is less than former three cities.

These experimental results show that comparing with candidate landmarks, using landmarks evaluated by our method for street-level geolocation can improve the geolocation accuracy.

6. Conclusions

Landmark-based street-level IP geolocation has a high application prospect in the field of Internet of Things. In view of the deficiencies of current methods in street-level landmark evaluation, a street-level landmark evaluation method based on the nearest router is proposed. Using the fact that "locations of terminal landmarks with the same nearest router are close to each other in geographical location", the relation model among distance constraint, distance, and region radius. In the basis of the relation model, the goal of evaluating street-level landmark is achieved. Meanwhile, the reliability values of evaluated street-level landmarks are calculated by initial reliability and binomial distribution. The experimental results show that the landmarks evaluated by this method increase the geolocation accuracy. Effectiveness of this method is affected by accuracy of delay value, landmarks distribution, and anonymous nearest router. In the future, the delay value acquisition method and street-level landmark evaluation method when nearest router is anonymous are the focus of work.

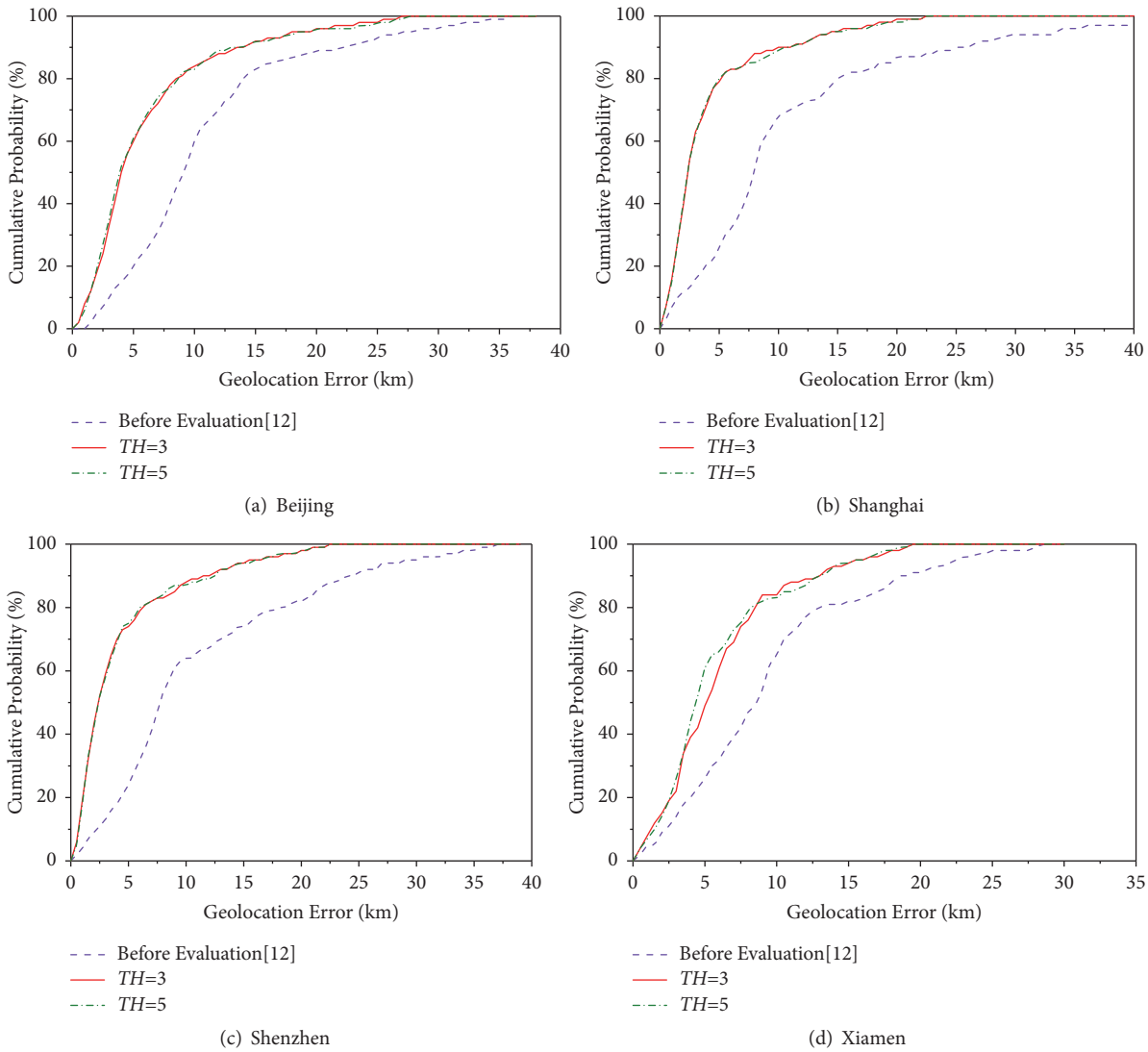


Figure 7: Relationship between geolocation error and cumulative probability.

Data Availability

The data in this article is mainly got from website pages (such as "http://tool.bridgat.com", "http://www.71ab.com/"). The DNS service is used to convert the domain name into IP, and the Baidu map API ("http://lbsyun.baidu.com/index.php?title=webapi/guide/webservicegeocoding") is used to convert the address information to latitude and longitude. Baidu ("http://lbsyun.baidu.com/index.php?title=webapi/ip-api"), IPIP ("https://www.ipip.net/"), and IP.cn ("https://ip.cn/") databases are used to verify candidate landmarks' city location.

Disclosure

Part of the paper was represented in the following conference: http://www.icccsconf.org/%E6%8E%A8%E8%8D%90SCI%E6%9C%9F%E5%88%8A%E8%AE%BA%E6%96%87%E5%88%97%E8%A1%A8.pdf.

Conflicts of Interest

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

Acknowledgments

The work presented in this paper is supported by the National Key R&D Program of China [nos. 2016YFB0801303 and 2016QY01W0105], the National Natural Science Foundation of China [nos. U1636219, 61602508, 61772549, U1736214, and 61572052], Plan for Scientific Innovation Talent of Henan Province [no. 2018JR0018], and the Key Technologies R&D Program of Henan Province [no. 162102210032].

References

[1] M. Gondree and Z. N. J. Peterson, "Geolocation of data in the cloud," in *Proceedings of the 3rd ACM Conference on Data and*

Application Security and Privacy, CODASPY 2013, pp. 25–36, New York, NY, USA, February 2013.

[2] E. Schmieders, A. Metzger, and K. Pohl, "A Runtime Model Approach for Data Geo-location Checks of Cloud Services," in *Service Oriented Computing and Applications*, vol. 8831 of *Lecture Notes in Computer Science*, pp. 306–320, Springer, Berlin, Germany, 2014.

[3] Z. Cai, H. Yan, P. Li, Z.-A. Huang, and C. Gao, "Towards secure and flexible EHR sharing in mobile health cloud under static assumptions," *Cluster Computing*, vol. 20, no. 3, pp. 2415–2422, 2017.

[4] L. Yang, Z. Han, Z. Huang, and J. Ma, "A remotely keyed file encryption scheme under mobile cloud computing," *Journal of Network and Computer Applications*, vol. 106, pp. 90–99, 2018.

[5] 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.

[6] X. Xu, W. Dou, X. Zhang, C. Hu, and J. Chen, "A traffic hotline discovery method over cloud of things using big taxi GPS data," *Software: Practice and Experience*, vol. 47, no. 3, pp. 361–377, 2017.

[7] Z. Zhou, W. Dou, G. Jia et al., "A method for real-time trajectory monitoring to improve taxi service using GPS big data," *Information and Management*, vol. 53, no. 8, pp. 964–977, 2016.

[8] X. Xu and W. Dou, "An assistant decision-supporting method for urban transportation planning over big traffic data," in *Proceedings of the International Conference on Human Centered Computing*, pp. 251–264, Phnom Penh, Cambodia, 2014.

[9] Y. Ma, X. Luo, X. Li, Z. Bao, and Y. Zhang, "Selection of rich model steganalysis features based on decision rough set α-positive region reduction," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1-1, 2018.

[10] Y. Zhang, C. Qin, W. Zhang, F. Liu, and X. Luo, "On the fault-tolerant performance for a class of robust image steganography," *Signal Processing*, vol. 146, pp. 99–111, 2018.

[11] X. Luo, X. Song, X. Li et al., "Steganalysis of HUGO steganography based on parameter recognition of syndrome-trellis-codes," *Multimedia Tools and Applications*, vol. 75, no. 21, pp. 13557–13583, 2016.

[12] C. Guo, Y. Liu, W. Shen, H. J. Wang, Q. Yu, and Y. Zhang, "Mining the Web and the Internet for Accurate IP Address Geolocations," in *Proceedings of the 28th Conference on Computer Communications (INFOCOM '09)*, pp. 2841–2845, IEEE, Rio de Janeiro, Brazil, April 2009.

[13] H. Jiang, Y. Liu, and J. N. Matthews, "IP geolocation estimation using neural networks with stable landmarks," in *Proceedings of the IEEE International Conference on Computer Communications Workshops*, pp. 170–175, San Francisco, Calif, USA, 2016.

[14] G. Zhu, X. Luo, F. Liu, and J. Chen, "An Algorithm of City-Level Landmark Mining Based on Internet Forum," in *Proceedings of the IEEE International Conference on Network-Based Information Systems*, pp. 294–301, Taipei, Taiwan, September 2015.

[15] Y. Wang, D. Burgener, M. Flores et al., "Towards street-level client-independent IP geolocation," in *Proceedings of the USENIX Conference on Networked Systems Design and Implementation*, pp. 365–379, Boston, Mass, USA, 2011.

[16] S. S. Siwpersad, B. Gueye, and S. Uhlig, "Assessing the geographic resolution of exhaustive tabulation for geolocating internet hosts," in *Proceeding of the Springer-Verlag International Conference on Passive and Active Network Measurement*, pp. 11–20, Cleveland, Ohio, USA, 2008.

[17] Y. Shavitt and N. Zilberman, "A geolocation databases study," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 10, pp. 2044–2056, 2011.

[18] L. Backstrom, E. Sun, and C. Marlow, "Find me if you can: Improving Geographical Prediction with Social and Spatial Proximity," in *Proceedings of the ACM International Conference on World Wide Web*, pp. 61–70, ACM, Raleigh, NC, USA, April 2010.

[19] I. Poese, S. Uhlig, M. A. Kaafar et al., "IP geolocation databases: unreliable?" *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 2, pp. 53–56, 2011.

[20] H. Li, P. Zhang, Z. Wang et al., "Changing IP geolocation from arbitrary database query towards multi-databases fusion," in *Proceedings of IEEE Symposium on Computers and Communications*, pp. 1150–1157, Heraklion, Greece, July 2017.

[21] H. Li, Y. He, R. Xi et al., "A Complete evaluation of the chinese IP geolocation databases," in *Proceedings of the IEEE International Conference on Intelligent Computation Technology and Automation*, pp. 13–17, Nanchang, China, June 2015.

[22] O. Dan, V. Parikh, and B. D. Davison, "Improving IP geolocation using query logs," in *Proceedings of the 9th ACM International Conference on Web Search and Data Mining, WSDM 2016*, pp. 347–356, San Francisco, Calif, USA, February 2016.

[23] D. Komosny, M. Voznak, S. U. Rehman et al., "Location accuracy of commercial IP address geolocation databases," *Information Technology and Control*, vol. 46, no. 3, pp. 333–344, 2017.